WANG

VS

File Management and Application Development Utilities

VS

File Management and Application Development Utilities

1st Edition — November 1988

715-1715

WANG

WANG LABORATORIES, INC.
ONE INDUSTRIAL AVENUE, LOWELL, MA 01851 TEL. (508) 459-5000, TELEX 172108

Disclaimer of Warranties and Limitation of Liabilities

The staff of Wang Laboratories, Inc., has taken due care in preparing this manual. However, nothing contained herein modifies or alters in any way the standard terms and conditions of the Wang purchase, lease, or license agreement by which the product was acquired, nor increases in any way Wang's liability to the customer. In no event shall Wang or its subsidiaries be liable for incidental or consequential damages in connection with or arising from the use of the product, the accompanying manual, or any related materials.

Software Notice

All Wang Program Products (software) are licensed to customers in accordance with the terms and conditions of the Wang Standard Software License. No title or ownership of Wang software is transferred, and any use of the software beyond the terms of the aforesaid license, without the written authorization of Wang, is prohibited.

Warning

This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device, pursuant to Subpart J of Part 15 of FCC rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

CONTENTS

CONTENTS (continued)

PART II GENERAL FILE MANAGEMENT UTILITIES

CHAPTER 8 COMPARE

CHAPTER 9 CONDENSE

CHAPTER 10 CREATE

CHAPTER 13 SELPRINT

CHAPTER 14 SORT

CHAPTER 15 SORTINT

CHAPTER 16 TABLEDIT

CHAPTER 17 TRANSL

PART III APPLICATION DEVELOPMENT UTILITIES

CHAPTER 18 COBGEN

FIGURES

FIGURES (continued)

FIGURES (continued)

FIGURES (continued)

TABLES

TABLES (continued)

PREFACE

INTRODUCTION

Wang Laboratories, Inc., provides a large number of utility programs designed to extend the support which the VS Operating System provides for the VS programming and application development environment. These utilities can be run by programmers and nonprogrammers alike, although a knowledge of at least one programming language is recommended. At a minimum, the user should be familiar with the VS system as described in the *VS System User's Introduction*.

Most VS utilities are documented in a set of three reference manuals, each of which deals with a different aspect of data processing operations; the rest are either described in such manuals as the *VS System Administrator's Reference* and *VS System Operator's Guide*, or have reference manuals of their own. A complete list appears below.

For Release 7 of the VS Operating System, several new utilities have been added, and a new organization of the documentation has been adopted. Users who are familiar with the previous organization will find two major differences:

- The former *VS System Utilities Reference* is replaced by the *VS Administration and Analysis Utilities Reference*, the *VS Media, Transfer, and Device Utilities Reference*, and in part by this manual.

- This manual replaces the former *VS File Management Utilities Reference*. It includes all the programs formerly documented in that manual, in addition to five that were formerly documented in the *VS System Utilities Reference* and nine that are new additions to the set of Wang-supported VS utilities.

ORGANIZATION OF THIS MANUAL

This manual is organized in three parts, each of which describes a distinct set of programs:

- Part I, *Data File Management Utilities*, describes five programs, formerly called simply File Management Utilities, that are used to create and maintain data files and to extract information from them.

- Part II, *General File Management Utilities*, describes ten programs used in various ways to manage files (locating, displaying, comparing, sorting, and similar operations). This group includes CONDENSE, formerly part of the set of utilities documented in Part I.

- Part III, *Application Development Utilities*, describes five programs of use mainly to programmers: they generate source code, aid program compilation, and so on.

CONDENSE has been removed from the set now called the Data File Management Utilities, mostly on the grounds that it is peripheral to the kind of fundamental data processing activity for which these programs are principally used. The kind of input file on which it operates cannot be created or used by the Data File Management Utilities, although CONDENSE does provide a way for these utilities to access the data in such files. Nevertheless, moving CONDENSE into Part II has the advantage of permitting the chapters in Part I to be arranged in an order that is both logical and alphabetical.

VS SYSTEM UTILITY DOCUMENTATION

The following list shows the VS system utilities with the manuals in which they are documented.

VS Administration and Analysis Utilities Reference (715-1717)

FASTLINK	LISTVTOC	SHRSTAT
IOELOG	PATCH	SSL
IOTRACE	POOLSTAT	VERIFY

VS Media, Transfer, and Device Utilities Reference (715-1716)

COPY	FONTCNTL	PSPRINT
COPYOIS	FORMCNTL	TAPECOPY
COPY2200	IBMCOPY	TAPEDUMP
DISKINIT	OISCART	TAPEINIT
FLOPYDUP	PERSON	

VS File Management and Application Development Utilities Reference (715-1715)

COBGEN	DISPLAY	RPTGEN
COMPARE	EZFORMAT	SELPRINT
COMPILE	EZPRINT	SORT
CONDENSE	FILEDISP	SORTINT
CONTROL	INQUIRY	TABLEDIT
CREATE	PROCGEN	TRANSL
DATENTRY	REPORT	

VS Operator's Guide (715-0418)

BACKUP	CIP	VOLCOPY

VS System Security Reference (715-1736) [SECURITY]

VS GENEDIT Utility Reference (715-1511)

VS Editor Reference (715-1143)

VS Linker Reference (715-1145)

VS Symbolic Debugger Reference (715-1144)

VS DMS Reference (800-1124)

UTILITIES IN THE VS ENVIRONMENT

All VS File Management and Application Development Utilities reside in the @SYSTEM@ library on the system volume. The name of each utility serves as the file name; thus, DATENTRY is located in the file DATENTRY in the library @SYSTEM@ on the system volume.

You can run utilities either directly, through the VS Command Processor, or under procedure control. (Some utilities can be run from the internal menus of other utilities; for example, DATENTRY, EZFORMAT, INQUIRY, and REPORT can all be called from within CONTROL.) When you run a utility by pressing PF1 from the Command Processor menu, you need enter only the utility's name in the File field, omitting library and volume (or leaving the default names unchanged). The VS Operating System always searches the @SYSTEM@ library on the system volume if it cannot locate the specified file in the indicated library and volume location. (The default library name should be deleted, however, if it contains a file with the same name as the utility.)

The VS utilities are designed so that workstation interaction can be controlled or minimized through the VS Procedure Language. Most requests for information are processed through GETPARMs, which are the standard interface between VS Procedure Language and a program. Appendix A contains a list and brief descriptions of the GETPARMs for each utility.

In this manual, sample procedures are provided for several utilities whose nature, being less interactive than most, makes them likely candidates for operation through a procedure.

SOURCES OF ADDITIONAL INFORMATION

The manuals listed below may be helpful when used in conjunction with this manual:

VS Assembly Language Reference (800-1200)
VS BASIC Language Reference (800-1202)
VS COBOL 85 Language Reference (715-0499)
VS Programmer's Guide to COBOL 85 (800-1214)
VS FORTRAN 77 Language Reference (800-1208)
VS System Administrator's Reference (715-0420)
VS Operating System Services Reference (715-0423)
VS Principles of Operation (715-0422)
VS Procedure Language Reference (800-1205)
VS RPG II Language Reference (800-1203)
VS System Operator's Guide (715-0418)
VS System User's Introduction (715-0417)
VS VSSUBS Reference (715-0599)

Part I

DATA FILE MANAGEMENT UTILITIES

CHAPTER 1
THE DATA FILE MANAGEMENT UTILITIES

1.1 OVERVIEW

The Wang VS System provides a set of system utility programs that enables a user with relatively limited data processing experience, and without the necessity of writing program code, to design, build, access, inspect, and report on DMS data files. These utilities are known collectively as the Data File Management Utilities[1]. Their functions are summarized in the following list:

CONTROL -- This utility creates a control file that contains all the specifications for the field, record, and file attributes of a specific data file. Once these are recorded in the control file, you can easily create, build, and maintain the data file itself through DATENTRY, or use EZFORMAT to create a data entry program specifically designed for this file. Chapter 2 describes the operation of the CONTROL utility.

DATENTRY -- Using the specifications in a control file, DATENTRY enables you to create a data file and enter or maintain records in it. The records in a file can also be listed on the screen or printed out for easy reference. Chapter 3 describes the operation of the DATENTRY utility.

EZFORMAT -- By generating a customized data entry and maintenance program based on a user-designed screen format and a control file, EZFORMAT provides an alternative to DATENTRY for existing DMS files. It can also aid the application developer by generating code through which a given screen format can be reproduced in a program. Chapter 4 describes the operation of the EZFORMAT utility.

[1] The Data File Management Utilities were formerly called the File Management Utilities. See the Preface for an explanation of this change.

INQUIRY -- This utility interrogates and tests a data file using criteria that you enter in a convenient, English-like syntax. The results can be displayed or printed. Chapter 5 describes the operation of the INQUIRY utility.

REPORT -- From the information in a control file and additional specifications you enter, REPORT defines the parameters and format of a printed report that uses the contents of one or two data files as input. You can use the Report Definition File in which this information is stored as often as you want.

1.2 DATA PROCESSING WITH THE DATA FILE MANAGEMENT UTILITIES

In processing files through the Data File Management Utilities, you work with sometimes one and sometimes two data files at a time. The data file is first defined, then created and filled with data records. Afterwards it can be interrogated or reported on. Figure 1-1 represents the relationship among the Data File Management Utilities in data processing.

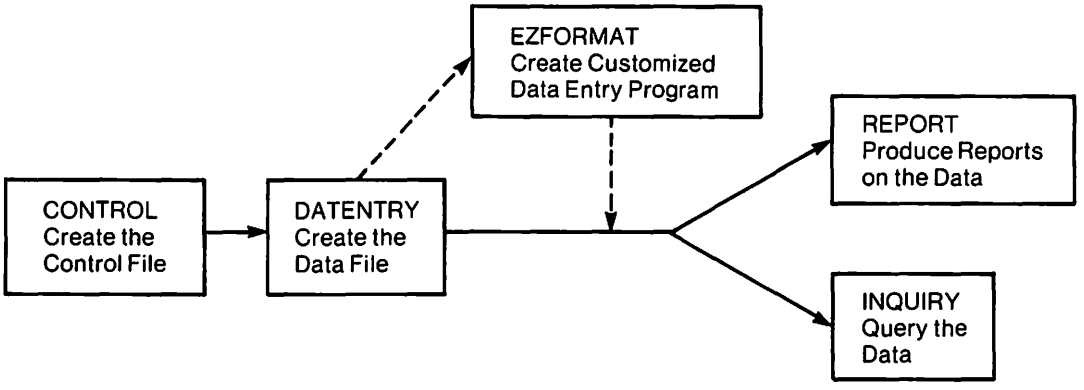


Figure 1-1. Data Processing With the Data File Management Utilities

The creation of a data file begins with the CONTROL utility. When you supply information to CONTROL, in response to the requests issued by the program, you define the physical and logical structure of the data file (which does not yet exist). The information you supply includes general information about the file, such as file organization, and about the data records it contains, such as record length. Through CONTROL you also specify the characteristics of each field in the record, such as its name, length, and internal format. You can also specify criteria for validating the data entered in a field. All the file, record, and field specifications are stored in a control file. (A single control file defines only one type of record, and data files created through the Data File Management Utilities contain only one type.) Once the characteristics of the data file are defined in the control file, you can go on to create it.

The actual creation of the data file -- and then the entry of data, which is a separate process -- is done with the DATENTRY utility. A control file is needed to provide structure and format: it determines what fields DATENTRY will display and what information will be accepted in them. When data is accepted, it is written into the newly created file. You can also use DATENTRY to add, delete, or modify data once a file has been created. The control file is needed for any of these operations.

If you would rather use a customized screen format for data entry instead of the default screen supplied by DATENTRY, you can use the EZFORMAT utility to create a special data entry program. EZFORMAT enables you to define a data entry screen for a specific data file described in a control file. This screen is incorporated into a program whose functions are similar to those of DATENTRY -- although it cannot create a data file, it can be used for entering, modifying, or deleting records.

The INQUIRY utility can retrieve selected portions of the data file by means of queries formulated in a syntax quite close to that for conversational English. You can display the retrieved data on the workstation screen or store it in an output file for later use. A control file is needed to provide the necessary information about the structure and layout of the data file.

To produce a printed report on one or two data files, you can use the REPORT utility. By specifying a control file and supplying additional specifications to REPORT, you can define the contents and format of the report. You can even define new fields (as long as their values can be derived from existing fields) and specify the format and kind of summary data to be included. The report definition is permanently saved in a Report Definition file (RDF), so that, in addition to generating routine reports, it can be periodically reviewed and revised according to changing needs. Reports can be either printed or displayed on the screen.

1.3 THE CONDENSE UTILITY

Although it is not included among the Data File Management Utilities in Part I of this manual, the CONDENSE utility is related to them insofar as it relies on control files in order to function[1]. CONDENSE is used to process an existing data file containing more than one type of record. A file of this kind cannot be created through DATENTRY, interrogated through INQUIRY, or reported on through REPORT. However, if each record type in the file is described by a control file, you can use CONDENSE to extract a new data file containing only one type of record; this record can, according to your specifications, include fields taken from records of different types in the input file. Chapter 9 describes the operation of this utility.

[1] CONDENSE was included in the document that previously described these utilities. See the Preface for more information.

CHAPTER 2
CONTROL

2.1 INTRODUCTION

The CONTROL utility is used to define the attributes of a data file. By supplying information to CONTROL, you determine the characteristics of the data file and the records it will contain. The utility creates a control file to contain these specifications, and the control file thereafter serves as a guide for other utilities as they are used to construct the data file, modify it, or retrieve information from it.

Any data file that is created or modified through the DATENTRY utility, accessed by the REPORT or INQUIRY utilities, or has an EZFORMAT-generated data entry program created for it must have an associated control file.

2.1.1 Control Files

The control file consists of one *file descriptor record*, a number of *field descriptor records* (one for each field in the data record this control file defines), and, optionally, one to three *comment records*. For alternate indexed data files there are also one or two *alternate key records*.

- The file descriptor record consists of fields that define the data file at the file level (for example, file organization) and the record level (for example, record length).
- The field descriptor records consist of fields that define the data record at the field level (for example, name, length, internal format, and validation criteria).
- Comment records are used as internal documentation; they appear only when the CONTROL utility is used to display them.
- Alternate key records keep track of the keys by which an alternate indexed file can be sorted or accessed, as through DATENTRY. There can be as many as 16.

Besides determining the design of a data file, the control file provides a basis for data control. You can use the CONTROL utility to place certain constraints on the entire data file, all fields of the record, or only particular fields of a record. By limiting the access to the data file or the types of data that can be entered, these constraints operate to insure accuracy in the input, update, and output of data. The DATENTRY, INQUIRY, and REPORT utilities, which are used for these operations, interact with the control file in such a way that the constraints are observed.

Note: *The control file provides a complete description or VIEW of a data file. If the control file is used in conjunction with DATENTRY to create the data file, the control file will accurately reflect the data file format. However, if the control file is modified in a way that changes the data file description, this will not be reflected in the data file, because control files are only a VIEW and cannot be used to actually alter a data file's format.*

If you anticipate a future expansion or change to the data file, it can be useful to reserve extra space in the record when the control file is originally created. This is done by setting the record length longer than the sum of the fields you are defining. At any time following, the control file can be modified, and descriptors defining fields at the end of the record can be added.

When you run DATENTRY after changing a control file, the new VIEW is used, and any new fields are available for input in the proper data format. Previously created records will contain spaces in this area. This is fine if the new field is of character format; however, if this area has been defined to have a numeric format, this area will contain invalid numeric data until modified and corrected.

Unless appropriate space has been provided, control file changes that affect the length of the data record require the creation of a new data file. A new file must also be created when the internal format of any field is changed in such a way as to affect its length. Using the CREATE utility described in Chapter 10 of this manual, or a user-written program, you can reconstruct an existing data file and produce a new file that contains all the original data, but in a new structure that accommodates the changes in the control file.

2.1.2 Running CONTROL

To run the CONTROL utility, press PF1 (RUN Program or Procedure) from the Command Processor menu and specify CONTROL in the Program field.

The following sections describe CONTROL processing:

Section	Process
2.2	Specifying the Control File
2.2.1	Listing and Selecting Control Files
2.3	Creating a Control File
2.3.1	Defining the File Header
2.3.2	Specifying the Fields in the Data Record
2.3.3	Providing Validation for Data Entered in the Fields
2.3.4	Using Repeated Fields To Make Tables
2.4	Maintaining a Control File
2.4.1	Adding Fields to a Control File (PF3)
2.4.2	Modifying Fields (PF4)
2.4.3	Modifying the File Header (PF↑4)
2.4.4	Deleting Fields (PF5)
2.4.5	Listing File Header and Field Information (PF6)
2.4.6	Creating and Modifying Validation Tables (PF7)
2.4.7	Modifying the Field Update Sequence for DATENTRY (PF12)
2.5	User Exit Subroutines
2.5.1	Writing a User Exit Subroutine
2.5.2	Linking DATENTRY to the User Exit Subroutine
2.5.3	Running the Linked Data Entry Program
2.6	Creating Source Code From a Control File
2.7	Running Other Utilities From Within Control

2.2 SPECIFYING THE CONTROL FILE

When CONTROL processing begins, the CONTROL File Specification screen (Figure 2-1) appears, prompting you for the name and location of a control file.

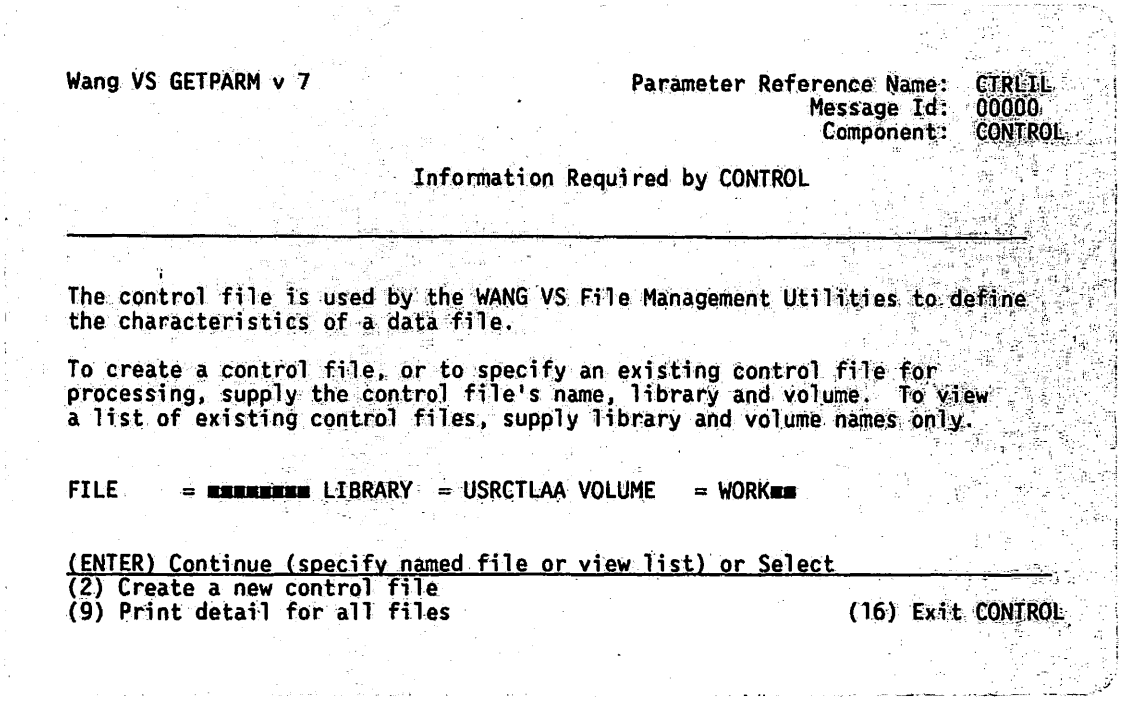


Figure 2-1. CONTROL File Specification Screen

You can specify a particular control file by supplying the file name, library, and volume and pressing ENTER. CONTROL also gives you the option of scanning a list of the files in your control file library and selecting a control file from the list. (For more information, see Section 2.2.1.) The following list describes the fields of the CONTROL File Specification screen and the way to enter file information in them.

FILE -- To specify a particular control file, either an existing file to be modified or examined or a new file to be created, enter the name in this field. The name you specify is the control file name. It must be from 1 to 8 characters in length and may not contain embedded blanks. Valid characters are A through Z, 0 through 9, @, #, and $.

To see a list of the files in your control file library, leave this field blank and specify only the library and volume names.

LIBRARY -- Enter the name of the library in which you want the control file stored. A default library name for the control file appears in the field. CONTROL creates it by concatenating your user ID with the letters CTL; for example, if your user ID is USR, the default library name is USRCTL.

If you want CONTROL to display a list of your control files, be sure this field contains the name of the library in which they reside.

VOLUME -- Enter the name of the volume on which you want the control file stored. If there is a default value in this field, it is taken from the INVOL field of your usage constants. (If none has been set, there is no default.) Usage constants are set through the SET Usage Constants function (PF2) of the Command Processor menu.

If you want CONTROL to display a list of your control files, make sure this field contains the name of the volume where they are stored.

When you supply all the requested information, including the file name, in the fields of the CONTROL File Specification screen, you have the following options:

- To create a new control file, press PF2. (Section 2.3 describes the creation process.)
- To view, modify, or maintain an existing control file, press ENTER. The CONTROL central menu offers several options, which are described in Section 2.4.
- To exit from CONTROL without further processing, press PF16.

If you supply the names of your control file library and volume, but leave the File field blank and press ENTER, CONTROL lists the contents of the library. You can select a control file from the list as though you had specified the actual file name. The next section describes this process.

2.2.1 Listing and Selecting Control Files

If you enter the name and volume of your control file library in the Library and Volume fields of the CONTROL File Specification screen, but leave the File field blank, the first CONTROL File List screen appears when you press ENTER. (See Figure 2-2 for an example of this screen.)

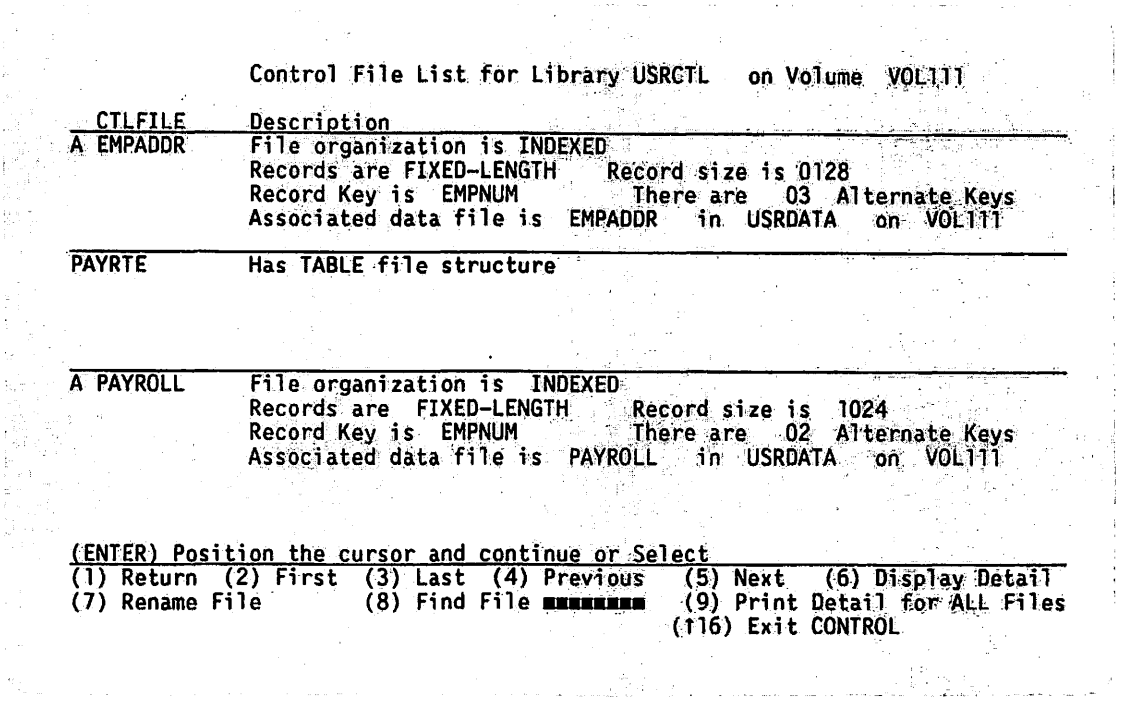


Figure 2-2. A Sample CONTROL File List Screen

The files in the library are listed alphabetically, three at a time. CONTROL can display as many CONTROL File List screens as needed for this purpose. The screens show valid control files with a pseudoblank at the left of the file name, and the description column reports the following:

- Type of file organization
- Record length
- Name of the key field (for all indexed files)
- Number of alternate keys (for alternate indexed files)
- Name and location of the associated data file, if one exists

The table files which CONTROL, at your option, creates for the purpose of data checking are housed in the same library as the control files. They are identified in the description column.

All the files in the library should be either control or table files, but if other kinds are present they are simply reported as having no header (i.e., no header of the kind that would identify them as control or table files).

Only the control files are listed with a pseudoblank to the left of each file name. You can specify any of these control files by placing the cursor on the pseudoblank and pressing ENTER. The result is the same as if you had specified that file name from the Control File Specification screen: the CONTROL central menu (Figure 2-9) appears.

Note: *To move to the next screen, press PF5, not ENTER. If you press ENTER while the cursor is on a pseudoblank, you specify the indicated file, and you must press PF16 from the Central menu to return and change the specification. If you press ENTER while the cursor is not on a pseudoblank, CONTROL displays an error message.*

The CONTROL File List screen offers other options as well. PF keys 2 through 5 let you move back and forth in the list to see all the files in the library:

PF2	Move back to the first screen
PF3	Move forward to the last screen
PF4	Move back to the previous screen
PF5	Move forward to the next screen

To locate a particular file quickly, you can use PF8 as follows:

1. Using the TAB key, move the cursor to the field labeled "(8) Find File" on the last line of the screen.

2. Enter the file name and press PF8. The screen that lists the file you specified appears immediately. (If the file is not found in the library, CONTROL puts the cursor on the file whose name comes next in ASCII order after the one you entered.)

You also have these options from the CONTROL File List screen:

PF Key	Option and Description
1	**Return** -- Press PF1 to return to the CONTROL File Specification screen (Figure 2-1).
6	**Display Detail** -- Press PF6 to see information from the file header and field records for the control file the cursor is on. This option is identical to the "List header and fields" option available from the Central menu (PF6). See Section 2.4.5 for a description of that option.
7	**Rename** -- This option is available only for control files. Place the cursor on the pseudoblank to the left of the file name you want to change, and press PF7. A screen appears that displays the original name, followed by a blank field for the new name. Supply the new name and press ENTER.
9	**Print Detail for ALL Files** -- This option is equivalent to pressing PF6 for every control file in the library, and directing the output to the printer instead of the workstation screen. Information is printed for control files only.
↑16 (32)	Terminate CONTROL processing.

2.3 CREATING A CONTROL FILE

To create a new control file, enter the file name, library, and volume in the fields of the CONTROL File Specification screen (Figure 2-1) and press PF2. If you have named a file that already exists in that location, CONTROL displays an error message asking you to press PF3 if you want to delete the existing file. If you choose not to do this, you must supply a new file name before you can proceed.

In creating a control file, you supply two kinds of information:

- Specifications for the whole data file and the records it will contain. (Remember that a control file can describe only one record type.) This information is contained in the file header.
- Specifications for each field in the record type this control file describes.

Note: *Appendix B describes the formats of the records in which the control file stores this information.*

2.3.1 Defining the File Header

When you press PF2 from the Central menu, the CONTROL File Header Definition screen (Figure 2-3) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  HEADER
                                                          Message Id:  0001
                                                           Component:  CONTROL

                        Information Required by CONTROL

Creation of file header information.  Enter parameters for the data file.

RECLEN   = ■■■■■        (2024 var/comp; 2040 fixed/relative; 2048 consec)
FILETYPE =  F           (F-Fixed, V-Variable, C-Compressed records)
FILEORG  =  I           (C-Consecutive, I-Indexed, R-Relative)
KEYFIELD = ■■■■■■■■     (specify if FILEORG is Indexed)
USEREXIT = ■■■■■■■■     (user supplied, if any: "USER1" -- "USER10")
UPDATE   =  0           (0-record update allowed, 1-not allowed)
DELETE   =  0           (0-record deletions allowed, 1-not allowed)
REPORT   =  0           (0-report allowed, 1-not allowed)
OPENSHAR =  Y            (Shared mode for DATENTRY, Y-Yes, N-No)
TIMEOUT  =  000         (Timeout for DATENTRY, 000 - 255 seconds)
DEFAULT DATA   FILE     = ■■■■■■■■ LIBRARY = ■■■■■■■■ VOLUME   =   ■■■■■■
COMMENT1 = ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
COMMENT2 = ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
COMMENT3 = ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

(ENTER) Continue header creation OR          (16) Return to file specification
```

Figure 2-3. CONTROL File Header Definition Screen

The values you enter from the CONTROL File Header Definition screen are used to construct a file descriptor record, or header, for the control file. The following list describes the fields of the screen. Specify the appropriate values for the data file you want this control file to describe. When you finish making specifications, press ENTER. (For more information about data files and records and the ways they are organized, see the *VS DMS Reference*.)

RECLEN (Record Length) -- Enter the length, in bytes, of the data record. The maximum length varies depending on the file type and organization. If the file is to contain variable-length records, specify the maximum record length you want to allow. Table 2-1 shows maximum record lengths possible for fixed, variable, and compressed files with consecutive, indexed, and relative organization.

Table 2-1. Maximum Record Lengths by File Type and Organization

File Organization	File Type		
	Fixed	Variable	Compressed
Consecutive	2048	2024	2024
Indexed	2040	2024	2024
Relative	2040	2040	----[a]

[a] Records in relative files cannot be compressed.

FILETYPE (File Type) -- Specify either F (the default), V, or C to indicate the structure of the data records in the file. These entries correspond to fixed-length records (F), variable-length records (V), and variable-length compressed records (C).

The DATENTRY utility cannot create records whose lengths are truly variable. When a control file specifies variable-length records, DATENTRY creates records with variable-length organization, but their length does not vary; all have the length specified as the maximum in the RECLEN field. You should therefore specify variable-length or compressed records only when you plan to enter data by some means other than the DATENTRY utility.

FILEORG (File Organization) -- Specify either C, I, or R to indicate whether the file is to be organized as a consecutive (C), indexed (I), or relative (R) file. The default value is I.

If you want your file to be alternate indexed, specify an indexed file here. You will be allowed to specify alternate keys from the screen that follows this one. If you elect to do so, the file organization is alternate indexed rather than indexed.

There are limitations on DATENTRY's ability to modify *consecutive* data files. Table 2-2 summarizes the available DATENTRY operations for all file organizations and types.

Table 2-2. DATENTRY Operations by File Organization and Type

File Organization	File Type	Add	Modify	Delete
Consecutive	Variable or compressed	Yes	No	No
Consecutive	Fixed	Yes	Yes	No
Indexed or relative	All	Yes	Yes	Yes

KEYFIELD (Primary Key Field) -- Enter the name of the field in the data record to be used as the primary key to an indexed file. Since duplicate values are not permitted in the Primary Key field, it should contain data whose value is unique for each record.

For example, in a field containing employee payroll data, the Last Name field would be a poor choice for primary key because of the likelihood that it would contain duplicate values -- i.e., identical surnames. However, a field certain to contain unique values (for instance, social security number or a company-assigned employee number) would serve well as a primary key.

Specify this field only if the value in the FILEORG field is I; i.e., only if the data file is to be indexed or alternate indexed.

The field name must be from one to eight characters long, with no embedded spaces. Valid characters are A - Z, 1 - 9, @, #, and $. The first character must be a letter.

USEREXIT (User Exit Subroutine) -- This field is for the name of a subroutine you may optionally invoke when using the DATENTRY utility. Such subroutines are typically used to validate or manipulate specified data as it is entered. For information about generating and using user exit subroutines, see Section 2.5. By supplying the name here, you insure that DATENTRY will invoke the subroutine each time a record is added to the file.

Subroutine names are limited to the values USER1 through USER10. Only one subroutine may be specified in any control file.

UPDATE (File Update Code) -- Indicates whether the DATENTRY utility can open the data file to add, delete, or change records in the file. Specify 0 to allow DATENTRY access to the file or 1 to prohibit access. (If the Update code is 1, DATENTRY cannot work with the data file in any way.)

DELETE (File Delete Code) -- The value in this field indicates whether the DATENTRY utility can be used to delete records from the data file. *(Note that records cannot under any circumstances be deleted from consecutive files.)* Specify 0 to allow record deletion or 1 to prohibit deletion. The Update code must also be 0 if deletion is allowed. (If UPDATE is 0 and DELETE is 1, DATENTRY can add or change records but cannot delete them.)

REPORT (File Report Code) -- The value in this field indicates whether the REPORT and INQUIRY utilities can access the data file. Specify 0 to allow access or 1 to prohibit access. If the Report code is 1, neither REPORT nor INQUIRY can be used to retrieve information from the data file.

OPENSHAR (Shared Mode Code) -- The value in this field indicates whether the DATENTRY utility is permitted to open the data file in shared mode. Specify Y for Yes or N for No.

TIMEOUT (Shared Mode Timeout) -- If the shared mode code is Y, the value in this field determines how long the DATENTRY utility should wait for another user to release a record. When the data file is open in shared mode, several users can access records in the same file, but only one user at a time has access to a given record. If DATENTRY finds that the record it is trying to access is held by another user, it can either wait indefinitely until the record is released, or wait for a specified time before it stops trying to access the record and displays a message reporting the ID of the user who is holding it.

Specify 0 (the default value) in the Shared Mode Timeout field to have DATENTRY wait for as long as necessary until a record is released. (No other DATENTRY operations are possible during this time.) Specify a value from 1 to 255 (seconds) to have DATENTRY time out after the specified number of seconds and report the ID of the user who is holding the record. If the shared mode code is N, the value in this field is ignored.

DEFAULT DATA FILE (Default Data File) -- Enter the name of the data file with which this control file is to be associated. This is a default only. You can override it by specifying a different data file when you run a utility such as DATENTRY or REPORT.

COMMENT1, COMMENT2, COMMENT3 (Comments) -- Comments are used for documenting a control file internally, so that you can keep track of its contents. The comments appear only when you display the file header by pressing PF6 (List header and fields) from the CONTROL central menu. You can enter three comments; each can be up to 60 characters long.

Defining Alternate Keys

If you specified an indexed file by accepting the default I in the File Organization field of the File Header Definition screen, the CONTROL Alternate Key Path Definition screen appears as soon as you press ENTER to store the file header information. (See Figure 2-4 for an example of this screen.) If the file organization is consecutive or relative, this screen does not appear; instead, the process of specifying fields (Section 2.3.2) begins.

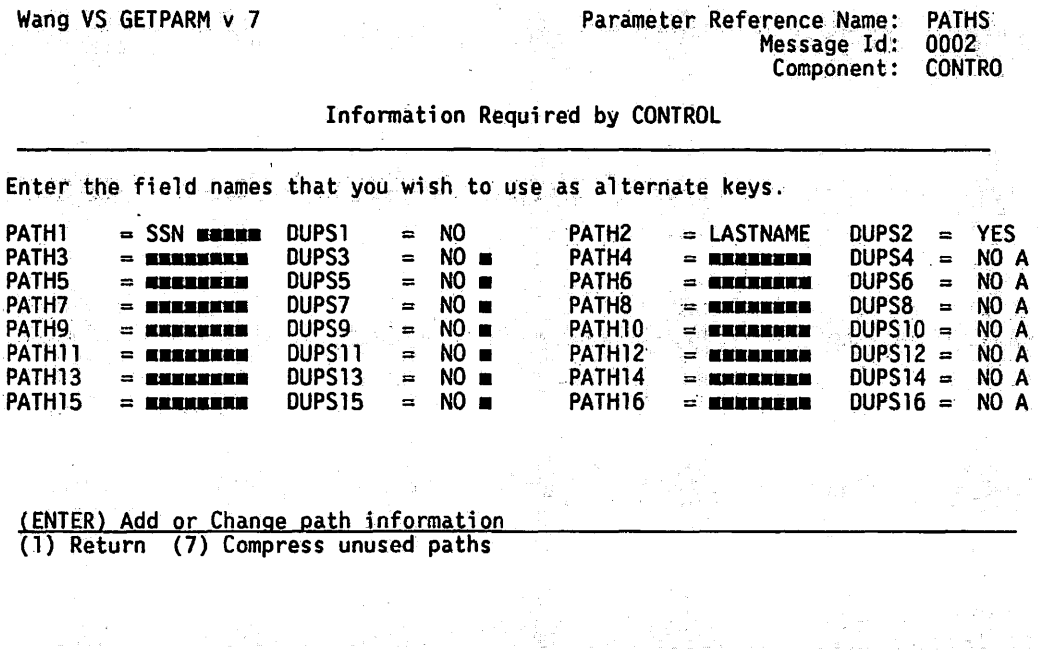


Figure 2-4. A Sample CONTROL Alternate Key Path Definition Screen

If you want a primary indexed rather than an alternate indexed file, simply press ENTER to skip this screen and begin the field specification process, described in Section 2.3.2. However, if you want an alternate indexed file, enter the names of all the alternate key fields in the Path fields provided on this screen. (For information about the relative advantages and disadvantages of indexed and alternate indexed file organizations, see the *VS DMS Reference*.)

Alternate keys, unlike the primary key, can have duplicate values. A field that would be unsuitable as the primary key because it might have the same value in numerous records (for example, the Last Name field in a file containing employee payroll records) would be fully serviceable as an alternate key. For each alternate key field that may contain duplicate values, change the default NO to YES in the DUPS field that corresponds to the Path field for that key. (It has the same number and appears immediately to the right of the Path field.)

When you finish specifying alternate keys, press ENTER. You also have the following options from the Alternate Key Path Definition screen:

PF Key	Option and Description
7	**Compress unused paths** -- This option provides a way to condense the information on the screen when you are editing it. When deletion or random entry leaves gaps between specified fields, you can press PF7 to move all the specifications into consecutive fields at the top of the screen.
1	**Return** -- Return to the File Header Definition screen without specifying alternate keys.

2.3.2 Specifying the Fields in the Data Record

When you press ENTER from the File Header Definition screen for a consecutive or relative file, or from the Alternate Key Path Definition screen for an indexed file, the first CONTROL Field Specification screen (Figure 2-5) appears.

```
                        FIELD SPECIFICATION SCREEN

                      Fill in the appropriate information

Field name  : ■■■■■■■■
Start loc   :  0001       (Starting location within the data record)
Int. format :  C          (P-packed, B-binary, C-character, Z-zoned, U-Unsigned)
Int. length : ■■■         (Maximum values: P-8, B-4, C-132, Z-15, U-15)
Ext. length : ■■■         (only if not calculated from internal length)
Decimal pos :  0          (0-9, specify for numeric fields only)
Occurrences :  01         (number of times this field repeated)
Report code :  0          (0-report on this field, 1-no report allowed)
Update code :  0          (0-modifiable, 1-nonmodifiable, no DATENTRY processing
Display code:  0          (0-blank field, 1-display last value, 2-display only)
0-suppress  :  0          (0-no zero suppress, 1-zero suppress, 2-* protect)
Sign control:  0          (0-no sign, 1-trailing minus, 2-CR on -, 3- DB on -)
Dollar/comma:  0          (0-none  1-comma, 2-dollar, 3-comma and dollar)
Binary edit :  0          (0-hex, 1-decimal, Binary format only)
Stamp field :  0          (0-no, 1-Date(MMDDYY), 2-Date(YYMMDD), 3-Day, 4-Time
                           5-Date & Time created, 6-Date & Time modified)
Cum. field  : ■■■■■■■■    (cumulative field name)
Field alias : ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ (alternate name used by INQUIRY)

(ENTER) Continue field addition OR
                                                  (16) Return to Central Menu
```

Figure 2-5. CONTROL Field Specification Screen

The specifications you enter from this screen define a field in the data record this control file describes. This definition is stored in a field descriptor record in the control file, which must contain one such record for each field the data record contains. As you finish entering the specifications for each field and press ENTER, a new Field Specification screen appears, from which you can define the next field. This process ordinarily continues until all the fields are defined, although you can suspend it at any point and continue the next time you run CONTROL. To terminate or interrupt the field specification process, press PF16.

Figure 2-5 shows the default values as they appear on the initial Field Specification screen. As successive screens appear, the value in the Start Location field changes to represent the first byte in the record that does not belong to a defined field. The default value in the Display Code field is the value specified from the previous screen; if you change 0 to 1 in this field, 1 will remain the default value until you change it to 0 again. All other fields that contain defaults revert to the values shown.

When you finish entering specifications, press ENTER to display the next Field Specification screen. When all the fields are specified, press PF16 to return to the CONTROL central menu.

Note: *There is a possibility of having up to 300 fields, 99 of which can be updated. All of the fields can have associated tables. For more information on using repeated fields to make tables, refer to Section 2.3.4.*

Enter your specifications in the appropriate fields as described in the following list. *Even if you accept all the defaults, you must enter values for at least the Field Name and Internal Length fields.* These two fields have no default values, but must be specified as part of any valid field definition.

Field name (Field Name) -- Enter the name of the field you are defining. The same field name is recognized and used by all the File Management utilities.

A valid field name must be from one to eight characters long, with no embedded spaces. Valid characters are A - Z, 1 - 9, @, #, and $. *The first character must be a letter.*

Note: *If you intend to choose the option to generate COBOL or RPG II code from this control file, or if you intend to use either COBGEN or EZFORMAT to generate COBOL or RPG II code on the basis of this file, you must observe the naming conventions and field size conventions appropriate to the language in which the code is to be generated. COBOL naming conventions are the same as CONTROL's, with the additional stipulation that COBOL "reserved words" cannot be used as field names. RPG II naming conventions are also the same as CONTROL's, except that the maximum length of the name is six characters.*

Start loc (Starting Location) -- Specify the field's starting location (in bytes) within the record. The displayed default value is the first byte in the record that does not belong to a defined field.

Valid starting locations range from the first byte in the record (location 1) to the last, whose location is the same as the record length specified in the file header. The starting location must be an integer and cannot exceed the record length.

When you create the first field descriptor record, you must either accept the default starting location, 0001, or specify a different starting location for the first field. Thereafter, the internal length of one field automatically determines the starting location of the next. You have the option of overriding the default starting location so that you can specify and respecify fields in any order. Cumulative fields, however, must be specified before their source fields. (For more information, see the description of the Cumulative field below.)

When more than one field descriptor defines the same area within a record, the fields are said to be overlapping. Modifiable fields cannot overlap with other modifiable fields. Nonmodifiable fields can overlap with modifiable and nonmodifiable fields.

Int. format (Internal Format) -- The value in this field determines the way the data in the field is physically stored on disk and the character values the field can accept. The following internal formats are available:

B Binary
C Character
P Packed decimal
U Unsigned
Z Zoned decimal

Character fields must have the internal format C. They are stored in ASCII code (one character per byte), and may be up to 67 bytes long if modifiable, or 132 bytes if nonmodifiable.

Numeric fields must have one of the internal formats B, Z, U, or P. Packed fields cannot exceed 8 bytes in internal length; zoned and unsigned fields cannot exceed 15 bytes.

Binary fields can be represented externally as either decimal or hexadecimal, according to the specification in the Binary Edit field (described below). Internally, binary fields cannot exceed four bytes in length.

For information about the zoned, unsigned, and packed decimal formats, see Appendix D in this manual, and the *VS Principles of Operation*.

Table 2-3 summarizes internal format characteristics.

Table 2-3. CONTROL Internal Formats for Data Fields

Internal Format	Maximum Internal Length (bytes)	Valid Internal Length Calculation	Entry Characters
C (character)	67 or 132	1 char/byte characters	All displayable
B (binary-decimal)	4	See Table 2-5	0 to 9, -
B (binary-hex)	4	2 char/byte A to F	0 to 9,
Z (zoned decimal)	15	1 byte/digit	0 to 9, -, .
U (unsigned)	15	1 byte/digit	0 to 9, .
P (packed decimal)	8	1 byte for first digit and sign; 2 digits/byte thereafter	0 to 9, -, .

Ext. length (External Length) -- This field specifies the number of characters or digit positions the DATENTRY utility should use to represent the field externally. If you leave it blank, the external length is calculated according to the formulas in Tables 2-4 and 2-5. You can override this default by specifying a positive integer in this field. (For more information on data formats, see Appendix D.)

Table 2-4. CONTROL Default External Length Values

Internal Format	External Length Calculation (in bytes)
C (character	Internal length
B (binary-decimal)	See Table 2-5
B (binary-hex)	2 x internal length
Z (decimal positions = 0)	Internal length + 1
Z (decimal positions > 0)	Internal length + 2
U (decimal positions = 0)	Internal length
U (decimal positions > 0)	Internal length + 2
P (decimal positions = 0)	2 x internal length
P (decimal positions > 0)	(2 x internal length) + 1

Table 2-5. Default External Length Values for a Binary-Decimal Format

Internal Length	External Length
1	4
2	6
3	8
4	11

Decimal pos (Decimal Position) -- For zoned (Z), packed decimal (P), or unsigned (U) fields only, specify the number of decimal positions, if any, to the right of the decimal point. This field must contain a positive integer from 0 (the default) to 9. If DATENTRY is used to enter data in this field, it insures that the field contains the specified number of decimal positions.

Occurrences (Occurrences) -- If the field you are defining is to be repeated within the data record, specify the number of times in this field. The Occurrences field must contain a positive integer from 1 to 99. If the number of occurrences is greater than 1 (the default), the field name is subscripted whenever it is represented externally by the DATENTRY, REPORT, INQUIRY, or CONDENSE utilities, and you must include a subscript when you specify the field to any of the same utilities.

If the value in the Update Code field is 0 to specify that this field is modifiable, CONTROL checks to make sure that its successive iterations (which must occur consecutively within the data record) do not overlap other modifiable fields.

Report code (Report Code) -- If you want to allow the REPORT and INQUIRY utilities to retrieve information from the field you are defining, specify a value of 0 (the default) in this field. A value of 1 prohibits retrieval. (If the Report field of the CONTROL File Header Definition screen [Figure 2-3] is set to forbid reporting on the file as a whole, a 0 in this field cannot override that prohibition.)

Update code (Update Code) -- The value in this field determines whether DATENTRY can be used to modify the contents of the data field you are defining. Specify 0 (the default) to allow the field to be modified or 1 to make it nonmodifiable. Nonmodifiable fields do not appear on DATENTRY screens.

Display code (Display Code) -- Specify this field only if the update code is 0; its value is otherwise ignored. The display code determines the way DATENTRY displays a modifiable field:

Code	Result
0	The field is cleared as each new record is displayed and contains only blanks when it appears on the next data entry screen. (Default.)
1	The value entered in the field for the last record remains as a default value for the new record.
2	The field is blank the first time the record is displayed, but once a value has been entered in this field it cannot be modified. When the record is recalled, the field is displayed with the value it contains, but new data cannot be entered in it.

0-suppress (Zero-Suppress Code) -- For a numeric data field (internal suppress format B, P, U, or Z), this code determines whether and how zeros will be suppressed when the REPORT utility represents data from the field you are defining. If reporting is not allowed for the data file or the field, do not specify a value for the Zero-Suppress Code field, since its value is ignored.

In reports generated by the REPORT utility, the value in the Zero-Suppress Code field affects the external format of the field as follows:

Code	Result
0	No zero suppression; all zeros are printed. (Default.) Example: $000612.50.
1	Leading zeros are suppressed; the leftmost nonzero digit is the first to be printed. Example: $ 612.50.
2	Leading zeros are replaced with asterisks (*) up to the leftmost nonzero digit. Example: $***612.50.

Sign control (Sign Control Code) -- For numeric data fields (internal format B [binary-decimal], P, U, or Z) only, this code determines the format in which the REPORT utility represents negative values for the field. If reporting is not allowed for the data file or the field you are defining, do not specify a value for the Sign Control Code field, since its value is ignored. The code affects the external format for negative field values as follows:

Code	Result
0	No sign; negative values are not distinguished from positive values. (Default.)
1	Trailing minus sign. Example: $005,935.00-.
2	Trailing CR (credit). Example: $005,935.00CR.
3	Trailing DB (debit). Example: $005,935.00DB.

Dollar/comma (Dollar Sign/Comma Code) -- For numeric data fields (internal format B [binary-decimal$), P, U, or Z), this code determines the placement of dollar signs and commas within the field value as it is represented by the REPORT utility. REPORT reserves extra space for fields lengthened by the addition of dollar signs or commas. If reporting is not allowed for the data file or the field you are defining, do not specify a value for the Dollar Sign/Comma Code field, since its value is ignored.

The code affects the external format as follows:

Code	Result
0	No dollar sign or commas. (Default.) Example: 1575347.52.
1	Commas inserted at 3-digit intervals to the left of the decimal point. Example: 1,575,347.52.
2	Dollar sign printed to the left of the field. Example: $1575347.52.
3	Commas inserted at 3-digit intervals to the left of the decimal point and dollar sign printed to the left of the field. Example: $1,575,347.52.

Instead of specifying this option for the control file, you can specify it through the REPORT utility at report definition time.

Binary edit (Binary Edit Code) -- For binary fields (internal format B) only, this code determines the way DATENTRY controls data entry into the field and the way values in the field are represented externally. A value of 0 (the default) specifies hexadecimal representation, which allows no signs or decimal points. A value of 1 specifies decimal representation. In either case, DATENTRY insures that decimal or hexadecimal entries are valid.

Stamp (Date, Day, or Time Stamp Field) -- A value other than 0 in this field marks the field you are defining as a date, day, or time stamp. When the record is created, any defined date, day, or time stamps are initialized to the system date and/or time in the format specified by the code. If a date and time stamp code of 6 has been defined, this stamp will contain the date and time the record was last updated through DATENTRY. If a Stamp field is defined in CONTROL after the record(s) have been created, these fields will be initialized to the system date and/or time when the records are modified in DATENTRY.

The code selects the date or time format as follows:

Code	Result
0	This is not a stamp field. (Default.)
1	System date in Gregorian MMDDYY format (e.g., March 7, 1988 is 030788).
2	System date in Gregorian YYMMDD format (e.g., December 30, 1991 is 911230).
3	System date in Julian format (YYDDD where DDD is the number of days after January 1: e.g., July 6, 1988 is 88188).

Code	Result
4	System time in HHMMSSHH format (H ours, M inutes, S econds, H undredths). For example, 4:34:51.96 P.M. is 16345196). The last two digits represent hundredths of a second.
5 & 6	System date in Gregorian YYMMDD format and system time in HHMMSSHH format. For example, July 6, 1988 at 4:34:41:96 is 88070604344196.

DATENTRY does appropriate validation checking for date and time stamp fields, insuring that months are in the range 1 to 12, days in the range 1 to 31 for codes 1 and 2, 1 to 365 (366 in leap years) for code 3, hours in the range 1 to 24, and minutes and seconds in the range 1 to 60.

The internal format for all stamp fields may be character (C) or unsigned (U). Packed format (P) is also permitted for all stamp fields with the exception of date and time stamps (codes 5 and 6).

For the Stamp field to function properly with the DATENTRY and REPORT utilities, you must explicitly set the external as well as the internal length. Table 2-6 shows the values appropriate for each code and internal format.

Table 2-6. Internal and External Lengths for CONTROL Stamp Fields

Code	Internal Format	Internal Length	External Length
1	C, U	6	6
	P	4	7
2	C, U	6	6
	P	4	7
3	C, U	5	5
	P	3	6
4	C, U	8	8
	P	5	9
5 & 6	C, U	14	14

Cum. (Cumulative Field) -- A Cumulative field is a numeric field used to sum or accumulate the quantities entered into certain other numeric fields in the same record through the DATENTRY utility. The Cumulative field must be defined before any of its source fields, and the internal format must be zoned (Z) or packed (P). The internal size of the Cumulative field must be at least as large as the largest source field.

Entering the name of a qualified numeric field in this field simultaneously identifies the field whose name is entered as a Cumulative field and the field being defined as one of its source fields. A source field may be of any numeric type.

Data entered in the source fields is accumulated in the Cumulative field. DATENTRY can display the latter, but cannot accept data entry directly into it. The accumulation reflects current source field values only; the Cumulative field is initialized and recalculated whenever any of its source fields is updated.

As an example of the relationship between Cumulative and source fields, you might want an individual payroll record to contain fields for regular and overtime hours worked, and a third field for total hours worked. If you define the latter as a Cumulative field and the first two as its source fields (by specifying, e.g., TOTHRS as the Cumulative field for both REGHRS and OTHRS), DATENTRY will sum the values entered in REGHRS and OTHRS and display them in TOTHRS without the user's having to calculate or enter the total. You cannot specify TOTHRS as the Cumulative field for REGHRS and OTHRS, however, until it has been defined. All updatable fields with the exception of simple table fields (see Section 2.3.4) may have a Cumulative field. If the source field is part of a complex table, all iterations of the source field will be totalled into the Cumulative field.

Field alias (Field Alias) -- You can supply any field with an alternate name or alias to be used with the INQUIRY utility. A field alias may be up to 31 characters long and may consist of several words separated by single spaces. The use of an alias instead of the field name makes it easier to compose English-like queries when you are running INQUIRY, since you are not bound by the constraints of field-naming conventions. You can, for example, use REGULAR HOURS instead of REGHRS or EMPLOYEE NUMBER instead of EMPNUM, if you assign these aliases to the fields in question.

Press ENTER when you finish making specifications from the Field Specification screen. If the field you are defining is modifiable (i.e., the value in the Update Code field is 0) and its internal length is 16 or fewer bytes, the CONTROL Validation Specifications screen appears as soon as you complete your field specifications and press ENTER. (See Figure 2-6 for an example of this screen.) Otherwise, a new Field Specification screen appears.

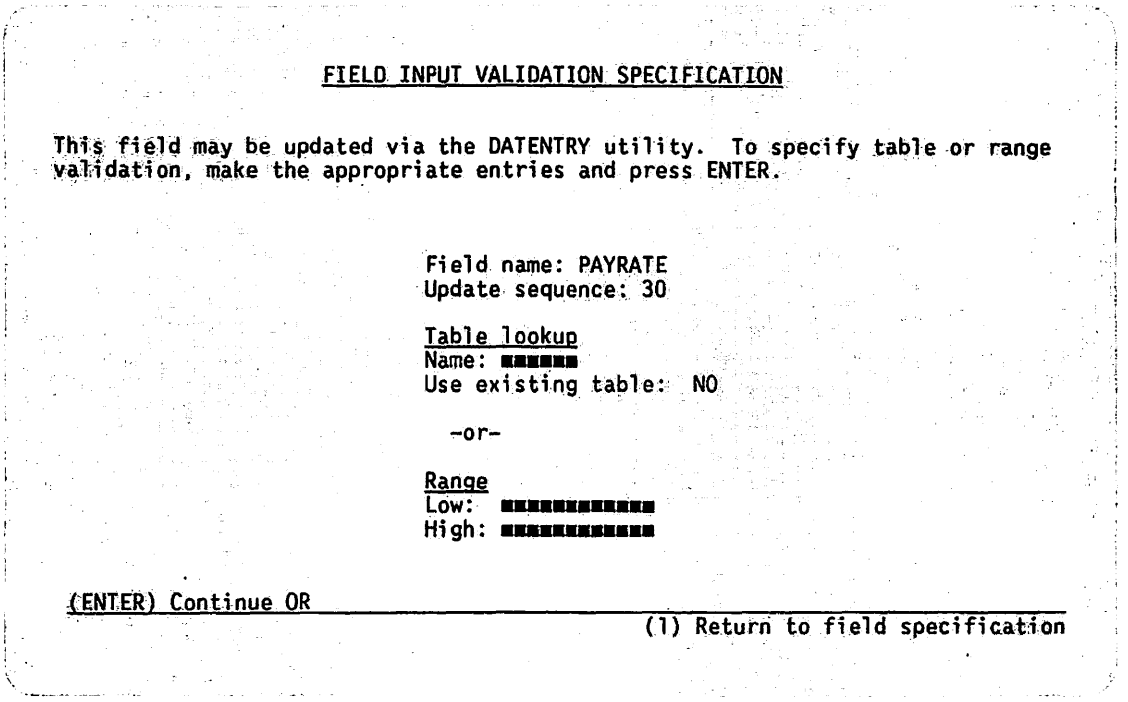


Figure 2-6. A Sample CONTROL Validation Specification Screen

2.3.3 Providing Validation for Data Entered in the Fields

The CONTROL and DATENTRY utilities jointly offer several ways of checking the validity of data as it is being entered. One type of validity checking is done automatically for date and time stamp fields; it is described in the comments on the Stamp field in the previous section.

For other modifiable fields, the following kinds of validation are available:

Table Validation -- DATENTRY compares the entered data with a table of specific acceptable values and rejects the data unless it matches an entry in the table.

Range Checking -- DATENTRY rejects data if it falls outside a specified range.

User Exit Subroutine -- A subroutine can be written to subject data to various tests as it is entered. This subroutine must be identified in the CONTROL field specifications and linked with the DATENTRY utility before data is entered in the field. User exit subroutines can be used to validate fields whose size makes them ineligible for table validation or range checking. Subroutines also permit more complex tests and responses. For information about using them, see Section 2.5.

Validation is relevant only to *modifiable* fields (i.e., fields whose Update code, as specified from the Field Specification screen, is 0).

Table validation and range checking are specified from the CONTROL Validation Specification screen, which appears immediately after a field is defined from the Field Specification screen *if the field is eligible for these types of validation*. (Figure 2-6 shows an example of this screen.) To be eligible, it must be a modifiable field and its internal length must be 16 bytes or less. You can specify either table validation or range checking for an eligible field, but not both.

The CONTROL Validation Specification screen shows the field name and its position in the update sequence (the order DATENTRY follows when it displays the fields for data entry -- see Section 2.4.7 for details).

Use this screen to specify either table or range validation (you cannot specify both). Two fields are provided for each type of validation. Specify only the fields that are relevant to the type you want; do not enter values in the other fields.

If you do not want either type of validation for this field, simply press ENTER to move on to a new Field Specification screen. Alternatively, if you want to change the specifications for the current field, you can return to the screen that displays those specifications by pressing PF1.

Specifying Table Validation

Table validation, in which DATENTRY compares the value entered to a table listing all acceptable values, is available for any modifiable field of 16 or fewer bytes in length, whether it contains character or numeric data. Enter values in the following two fields of the Validation Specification screen (Figure 2-6):

Name -- Enter a name for the validation table in this field. If you are creating a new table, CONTROL uses this name for the file in which it puts the table values. The table file, an indexed file containing fixed-length records, is housed in your control file library. A valid table file name must consist of alphanumeric characters (A - Z and 0 - 9), and have no embedded spaces. The name must begin with a letter, although digits are permitted in other positions.

It is not necessary to create a new validation table; you can use one that already exists. Multiple fields can use the same table. (If you are entering the name of an existing table file, be sure to change the value in the Use Existing Table field from NO to YES.)

Use Existing Table -- If you are creating a new validation table, leave the default NO in this field. If the table you have specified in the Name field already exists, change the value to YES. CONTROL checks your control file library for the table file, and displays an error message if it is not found.

When both fields contain the appropriate values, press ENTER. If you specified a new validation table, the CONTROL Table Specification screen appears. (See Figure 2-7 for an example of this screen.) If you specified an existing table (and if CONTROL located the table and found that its internal format corresponds to that of the field being defined), the definition of the current field is complete, and a new Field Specification screen appears.

To create a new validation table, you enter the acceptable values for the data field, one at a time, in the Table Value field of the Table Specification screen. Each time you press ENTER, the value in the field is added to the table, and the field is cleared to accept another value. Continue until you have entered all the acceptable values for the field, then press PF3 to complete the process. A new Field Specification screen appears, and you can begin defining the next data field.

The Table Specification screen displays the table name. If you want to change this, or if for any other reason you want to return to the Validation Specification screen, press PF1.

The Table Value field is automatically set equal to the internal length of the table. DATENTRY is capable of checking a field against a table whose entries are longer than the field -- for instance, it can check a five-digit Zip Code field against a table containing nine-digit zip code values -- but if such a table does not already exist, you must use the PF7 option from the central menu to create it. See Section 2.4.6 for information on this option.

The same option, which enables you to update as well as create validation tables, makes it possible to specify a validation table but postpone the process of entering values until a later time.

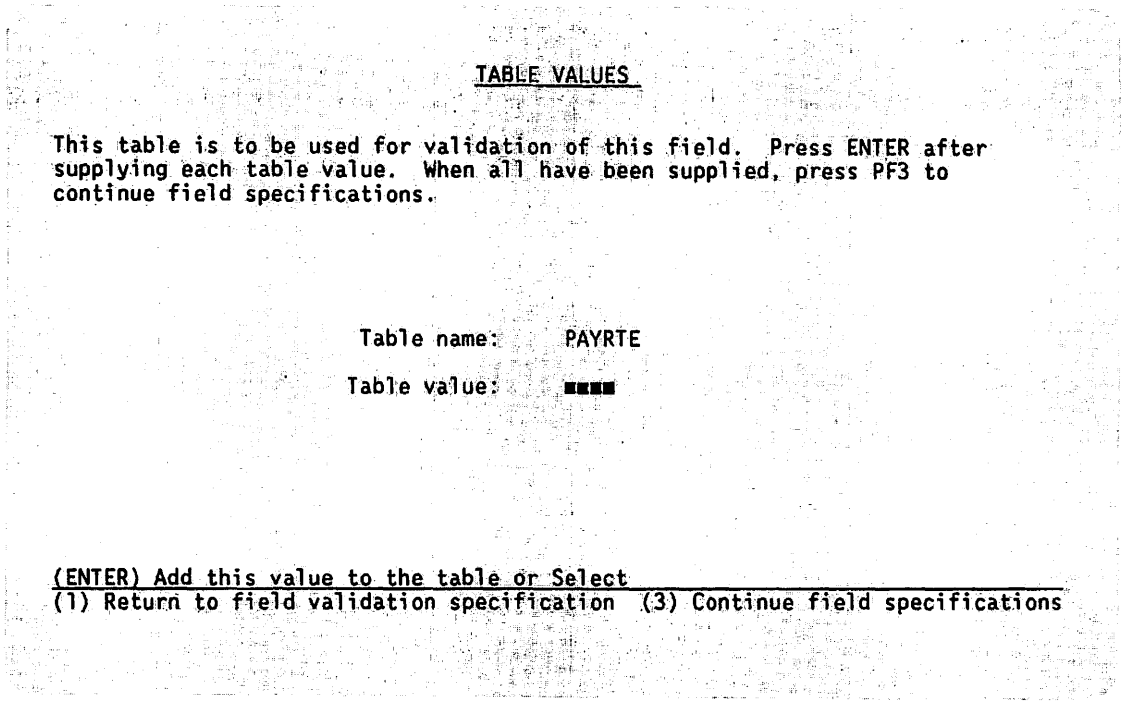


Figure 2-7. A Sample CONTROL Table Specification Screen

Specifying Range Validation

In range validation, DATENTRY compares the value entered in a field with specified high and low values and rejects it if it is greater than the high value or less than the low. Any value within the contiguous range between (and including) these limits is accepted. Range validation is available for any modifiable field up to 16 bytes in length. It is most commonly used for numeric fields, but can be used for character fields as well. (For example, you can insure that a six-byte character field accepts only values beginning with M, N, O, P, or Q by setting the low value to LZZZZZ and the high value to RAAAAA for that field.)

To specify range validation, enter the low and high values in the appropriately labeled fields of the CONTROL Validation Specification screen (Figure 2-6), and press ENTER. (Be sure there is no value in the Name field under "Table lookup," or an error message will result.)

When the range specifications are entered, the definition of the current data field is complete, and the next Field Specification screen appears.

Note: *When the EZFORMAT utility is used to generate COBOL or RPG II code for an EZFORMAT screen image, it uses CONTROL's information about table validation or range validation in creating a file descriptor record (FDR) in the source file.*

2.3.4 Using Repeated Fields To Make Tables

It is often useful to arrange data within a record in tabular form. Information in a table can either be simple or complex. For instance, a company might want to keep track of the hours worked by each of its employees for each month during a fiscal year. The record for each employee might include information that appears only once (such as employee number and name) and the number of hours worked for each month in the year. This monthly information is most conveniently organized as a table which might represent a payroll ledger. It would be possible to build such a table by simply specifying that the modifiable field REGHRS occurs 12 times. In DATENTRY the record would be represented on the screen as

```
NAME XXXXXXXXXXXXXXX   NUMBER XXXXXX  REGHRS (01) XX  REGHRS (02) XX
..... REGHRS (12) XX
```

This representation is adequate for its original purpose. Now, for instance, the company wants to add overtime hours worked by each of its employees for each month as well as keep track of each individual's pay rate. They could define OTHRS as a modifiable field occurring 12 times. In DATENTRY the record would be represented on the screen as

```
NAME XXXXXXXXXXXXXXX   NUMBER XXXXXX   REGHRS (01) XX  REGHRS (02) XX
..... REGHRS (12) XX  OTHRS (01) XX ....OTHRS (12) XX
```

Although the information would be available, it would no longer be representative of the payroll ledger, which has each employee's hours and overtime hours side by side. Although the design of the data record has been simple, the ease of data input and review is greatly hampered by the placement of the information on the data entry screen.

A more complex data design can be implemented which places the information in a more appropriate format. The employee number and name information is defined as before. The table definition, however, is quite different. To begin: Total the length of the two pieces of information (REGHRS-2 long and OTHRS-2 long). Using this information we will define a nonmodifiable field called MONTH (4 long) which occurs 12 times. Let's say MONTH started in column 41 of the data record. Next define REGHRS as starting in column 41, 2 long, modifiable and OTHRS as starting in column 43, 2 long, modifiable. You have now described a table which would appear in DATENTRY as

```
NAME XXXXXXXXXXXXXXX   NUMBER XXXXXX   REGHRS (01) XX  OTHRS (01) XX
..... REGHRS (12) XX  OTHRS (12)
```

The following is a listing of the tables described above.

Simple Tables -- These tables consist of a single updatable field which is to be repeated. These table are represented as FIELD (01) FIELD (02) etc.

Complex Tables -- These tables consist of related pieces of information which are to repeat multiple times. To define a complex table the following must be performed: (1) Define a nonmodifiable field (the total length of a single set of all related pieces of information) which occurs a multiple of times. (2) Define the individual pieces of information (component fields) as modifiable fields occurring a single time. (This field must fall within a single occurrence of the previously defined nonmodifiable field to be part of the table.)

When you build tables by this method, keep the following points in mind:

- Define the individual fields in exactly the order in which you want them to appear for updating. The original update sequence of the modifiable fields within a table field is determined by the order in which they are created. Modification of the update sequence for these fields is more restrictive than with normal single occurrence fields. See "Repeated Fields, Table Fields, and Update Sequence" in Section 2.4.7.

- Do not specify that any component field should occur more than once. (Two-dimensional tables are not supported in CONTROL.)

- A table field may contain nonmodifiable as well as modifiable fields. These do not appear on the DATENTRY screen or affect the update sequence (see Section 2.4.7), but may be useful in reporting.

- When you define a complex table, make its starting location the same as that of the first component field, make its internal length equal to the sum of the lengths of the component fields, and set the update code to 1 (nonmodifiable).

- Set the occurrence count for the table field to the number of times you want it repeated within the record -- i.e., to the number of rows in your table.

Figure 2-8 lists field information for the payroll record used as an example above. (The screen shown is a Field List screen, which is described in Section 2.4.5.)

List Control File Fields

	Field Name	Start Posn	Int Fmt	Int Len	Ext Len	Dec Pos	Occur Count	Report/ Update	Upd Seq	Stamp	Cum Field	Table/ Range
*	EMPNUM	1	U	5	5	0	1	R/U	1			Range
*	SSN	6	P	5	9	0	1	R/U	2			
*	LASTNAME	11	C	15	15	0	1	R/U	3			
*	FIRSTNAME	26	C	15	15	0	1	R/U	4			
*	MONTH	41	U	4	4	0	12	R	5-28			
*	REGHRS	41	U	2	2	0	1	R/U	(5)		YEARLY	
*	OTHRS	43	U	2	2	0	1	R/U	(6)		YEARLY	
*	YEARLY	89	Z	5	5	0	1	R/U	29			
*	PAYRATE	94	P	2	5	2	1	R/U	30			PAYRTE
*	VACATION	96	Z	3	5	1	12	R/U	31-42			
	-Not used-	132		269								

(ENTER) Continue to modification after positioning the cursor
(1) Redisplay header (2) First (3) Last (4) Previous (5) Next (16) Return

Figure 2-8. Field Information for a Sample Record (CONTROL Field List Screen)

Note that the table field MONTH begins at the same location (byte 41) as its first component field REGHRS, and contains 4 bytes, the sum of the internal lengths of REGHRS and OTHRS. The occurrence count is 12 for MONTH, 1 for each of the component fields. MONTH is assigned positions 5 through 28 in the update sequence; that is, it counts as 24 modifiable fields (12 iterations each of REGHRS and OTHRS). REGHRS and OTHRS are assigned positions 5 and 6, but the numbers are placed in parentheses to indicate that they are parts of a table field and will thus occupy other positions also within the range 5 to 28.

Note: *For the way DATENTRY displays the modifiable fields for this record, see Figure 3-5.*

2.4 MAINTAINING A CONTROL FILE

As soon as you specify an existing control file, either by entering its name from the CONTROL File Specification screen or by selecting it from a CONTROL File List screen, or as soon as you halt the definition of a new control file by pressing PF16 from a CONTROL Field Specification screen, the CONTROL central menu (Figure 2-9) appears.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  OPTIONS
                                                      Message Id:  002
                                                       Component:  CONTRO

                        Information Required by CONTROL

Press the appropriate PFkey for the control file option

      CONTROL FILE FUNCTIONS                    UTILITY FUNCTIONS

      (3)   Add new fields                      (9)  Run DATENTRY
      (4)   Modify existing fields              (10) Run REPORT
      (14)   Modify field header                (110) Run INQUIRY
      (5)   Delete fields                       (11) Run EZFORMAT
      (6)   List header and fields
      (7)   Create or maintain table files
      (8)   Create source from the control file
      (12)  Modify the field sequence for DATENTRY


      (16)  Exit to list or respecify the Control File
     (↑16)  Exit CONTROL
```

Figure 2-9. CONTROL Central Menu

To choose an option from the menu, press the indicated PF key. When a function is complete and the menu reappears, the line that identifies the completed function is highlighted. PF16 returns you to the CONTROL File Specification screen, from which you can specify another control file and continue processing, or exit from CONTROL.

Some options available from this menu enable you to examine and modify control files. Others create source code or link to other utilities. The rest of this section describes the options used in maintaining a control file. Other sections describe the other options. The following two lists provide a general description of all of the options. The lists correspond to the two major divisions on the screen, control file functions and utility functions.

The CONTROL central menu lists the following control file functions:

PF Key	Option and Description
3	**Add new fields** -- Enables you to add fields to the record described in an existing control file. Section 2.4.1 describes this process.
4	**Modify existing fields** -- Enables you to change the characteristics of individual fields in the record. Section 2.4.2 describes this process.
14 (20)	**Modify field header** -- Enables you change the general data file characteristics specified in the file header. Section 2.4.3 describes this process.
5	**Delete fields** -- Enables you to delete fields from an existing control file. Section 2.4.4 describes this process.
6	**List header and fields** -- Enables you to display or print the contents of the control file. Section 2.4.5 describes this process.
7	**Creating and modifying validation tables** -- Enables you to create a validation table or list, add, delete, or change the contents of validation tables created by the CONTROL utility. Section 2.4.6 describes this process.
8	**Create source from the control file** -- Enables you to create a source file containing code that defines the data file described by the control file. Three source languages are available: COBOL, PL/I, and RPG II. Section 2.6 describes this process.
12	**Modify the field sequence for DATENTRY** -- Enables you to rearrange the order in which modifiable data fields appear on the DATENTRY screen. Section 2.4.7 describes this process.

The CONTROL central menu lists the following utility functions:

PF Key	Option and Description
9	**Run DATENTRY** -- Runs the DATENTRY utility, which enables you to add, modify, or delete information in a data file described by a control file. Chapter 3 describes this utility.
10	**Run REPORT** -- Runs the REPORT utility, which enables you to create a report from one or two data files described by control files. Chapter 6 describes this utility.
10 (26)	**Run INQUIRY** -- Runs the INQUIRY utility, which enables you to query a data file described by a control file. Chapter 5 describes this utility.
11	**Run EZFORMAT** -- Runs the EZFORMAT utility, which enables you to define a screen image for data entry. Chapter 4 describes this utility.

You also have these options from the central menu:

PF Key	Option and Description
16	**Exit to list or respecify the Control File** -- Returns you to the CONTROL File Specification screen (Figure 2-1), if you specified the current control file from that screen. If, however, you selected it from a CONTROL File List screen (Figure 2-2), PF16 returns you to the first CONTROL File List screen for your control file library.
↑16 (32)	**Exit CONTROL** -- Terminates CONTROL processing and returns you to the VS Command Processor or to the program or procedure from which CONTROL was called.

In addition to the options available from this menu, you can impose further restrictions on a field by means of a user exit subroutine. Section 2.5 describes this process.

For descriptions of the remaining options, see the sections indicated in the "Control File Functions" and "Utility Functions" lists above.

2.4.1 Adding Fields to a Control File (PF3)

Note: *You can add fields to an existing control file at any time. If, however, the control file describes an existing data file whose records conform to its original specifications, you will have to recreate the data file in conformity with the new specifications. The CREATE utility (Chapter 10) can be used for this purpose.*

To add fields to an existing control file, press PF3 from the CONTROL central menu (Figure 2-9). A new Field Specification screen (Figure 2-4) appears. The default value in the Starting Location field is the first available byte in the data record, i.e., the first byte that does not belong to a previously defined updatable field.

You define the field in exactly the same way as when you are creating a new control file; Section 2.3.2 describes the process in detail.

If the data record is already filled with modifiable fields and contains no available bytes, only nonmodifiable fields can be added. CONTROL displays a message to this effect on the Field Specification screen, and places a value of 1, which you cannot change, in the Update Code field. You can make room for modifiable fields only by deleting existing fields (see Section 2.4.3) or by modifying the file header to increase the length of the record (see Section 2.4.2).

When the DATENTRY utility displays fields for data entry, it will put the newly added fields after all the others, unless you modify the field update sequence in the control file. Section 2.4.7 describes modifying the field update sequence for DATENTRY (PF12).

2.4.2 Modifying Fields (PF4)

Note: *You can modify any field in the control file at any time, although you must take care that your changes do not interfere with other fields. You cannot, for example, increase the internal length of a modifiable field so that it overlaps another, or change the internal format of a stamp or cumulative field in a way that makes its previously defined field type invalid. If a control file corresponding to an existing data file has its field specifications modified in such a way that it no longer describes the data file with complete accuracy, the data file must be modified or recreated.*

To select a field for modification, press PF4 from the CONTROL central menu (Figure 2-9). A Field Selection screen appears. (Figure 2-10 shows a sample CONTROL field selection screen.)

```
     Select a field from Control File PAYROLL  For Modification

Supply the name or position the cursor to specify a field

Field name : ■■■■■■■■

    Field    Start Int Int Ext Dec Occur Report/ Upd  Stamp  Cum     Table/
    Name     Posn  Fmt Len Len Pos Count Update  Seq         Field   Range
■ EMPNUM       1   U    5   5   0    1    R/U    1
■ FRSTNAME    26   C   15  15   0    1    R/U    4
■ LASTNAME    11   C   15  15   0    1    R/U    3
■ MONTH       41   U    4   4   0   12    R      5-28
■ OTHRS       43   U    2   2   0    1    R/U   (6)
■ PAYRATE     94   P    2   5   2    1    R/U   30
■ REGHRS      41   U    2   2   0    1    R/U   (5)
■ SSN          6   P    5   9   0    1    R/U    2
■ VACATION    96   Z    3   5   1   12    R/U   31-42
■ YEARLY      89   Z    5   5   0    1    R/U   29

 File organization is indexed, with key field = EMPNUM   Alternate Paths are:
   SSN      LASTNAME

(ENTER) Display field to be modified or Select
(1) Return  (2) First  (3) Last  (4) Previous  (5) Next  (7) Rename
```

Figure 2-10. A Sample CONTROL Field Selection Screen

The Field Selection screen shows the name of the control file and provides a blank field (Field name) in which you can enter the name of the field you want to modify. The screen also lists the fields in the record, ten at a time, *in alphabetical order*, with a pseudoblank to the left of each field name. The format of the list is the same as that of the Field List screen (Figure 2-12), which is described in detail in Section 2.4.5. A line below the list describes the organization of the file.

You can select a field by either

- Entering its name in the Field Name field and pressing ENTER
- Placing the cursor on the pseudoblank to the left of the field name in the list and pressing ENTER

To find a particular field in the list, you can move from screen to screen using the following PF keys:

PF2 Move back to the first screen
PF3 Move forward to the last screen
PF4 Move back to the previous screen
PF5 Move forward to the next screen

The TAB key moves the cursor from the blank field to the first pseudoblank, and from one pseudoblank to the next.

You also have these options from the Field Selection screen:

PF Key	Option and Description
1	**Return** -- Return to the central menu.
7	**Rename** -- Rename the field indicated by the cursor. When you press PF7, a screen appears that shows the current name and provides a blank field in which to enter the new name. When you press ENTER, the Field Selection screen reappears. (Since this screen lists files alphabetically, you may not see the name of the file you have just renamed; it may now appear on a different screen.) *This function provides the only means of renaming a previously defined field; you cannot do so by entering a new name from the Field Specification screen.*

When you have selected a field to modify and pressed ENTER, the Field Specification screen for the selected record appears, showing the specifications previously entered in its fields. Modify these as appropriate by entering new values. The fields of this screen are described fully in Section 2.3.2.

When your modifications are complete, press ENTER, and the CONTROL Validation Specification screen (Figure 2-6) for the same field appears. You can change the validation specifications by modifying values in the fields on this screen, which are described in Section 2.3.3. (Alternatively, you can return to the Field Specification screen for the same field by pressing PF1.) When your changes are complete, or if you choose not to change the validation specifications, press ENTER. The Field Selection screen reappears, and you can select another field to modify, or, if all field modifications and name changes are complete, press PF1 to return to the CONTROL central menu.

2.4.3 Modifying the File Header (PF14)

To modify the control and data file header, press PF4. The File Header Definition screen (Figure 2-3) appears, and you can change any of the general file characteristics specified in the fields of that screen. The fields are fully described in Section 2.3.1.

As always, you should be careful of making changes (such as shortening the record length) that would require the data file, if any, to be recreated. When you finish making changes in the file specifications, press ENTER. The CONTROL central menu (Figure 2-9) reappears.

2.4.4 Deleting Fields (PF5)

You can delete fields from a control file at any time, but you should be cautious about making modifications that may require you to reconstruct an existing data file. See the note at the beginning of Section 2.4.1.

To select a field for deletion, press PF5. A Field Selection screen appears. Its layout is like that of the sample screen illustrated in Figure 2-10, except for a heading indicating that the field is to be selected for deletion rather than modification, and the absence of an option to change a field name. You can select a field either by entering its name in the Field Name field or by placing the cursor on the pseudoblank to the left of the name where it appears on the screen. All fields on the screen *are listed alphabetically*. (See Section 2.4.2 for a detailed description of Field Selection screens and the ways you can move around them.)

When you have indicated a field, press ENTER. The Field Deletion screen (not shown) appears. It displays the specifications for the field you have selected, and prompts you to press ENTER to confirm the deletion. When you do this, the Field Selection screen reappears with a message reporting the deletion of the field. If, before pressing ENTER, you decide not to delete the field, you can press PF1 instead of ENTER to return to the Field Selection screen. In this case, a message reports that the field was not deleted.

When you have deleted all the fields you want to delete, press PF1 from the Field Selection screen to return to the CONTROL central menu.

2.4.5 Listing File Header and Field Information (PF6)

In the course of creating or modifying a control file, it is useful to be able to list the characteristics of the file and the fields already defined. By pressing PF6 from the CONTROL central menu, you can either display this information at your workstation or send it to the printer.

When you press PF6, the CONTROL File Header Information screen appears. (Figure 2-11 shows an example of such a screen.)

```
Wang VS GETPARM v 7                          Parameter Reference Name:  CTLSRC
                                                          Message Id:  0001
                                                           Component:  CONTRL

                        Information Required by CONTROL

_____List for Control File PAYROLL  in Library USRCTL   on Volume VOL111 ___

          Records are fixed-length, record size is  400
          File organization is indexed, with key field = EMPNUM
          Alternate paths are:
  SSN     LASTNAME

Operation       Allowed?  Operation       Allowed?  Operation       Allowed?
Report            YES     Record update     YES     Record deletion   YES

Comment 1:    Employee Payroll Record
Comment 2:    Updated Weekly

            LISTING DEVICE   =  SCREEN A (SCREEN, PRINTER)

(ENTER) Continue to display detail                 (16) Return to Central Menu
```

Figure 2-11. A Sample CONTROL File Header Information Screen

The CONTROL File Header Information screen displays the characteristics of the control and data files as they were specified from the CONTROL File Header Definition screen (Figure 2-3). They are listed in Table 2-7.

Table 2-7. File Information Displayed on CONTROL File Header Information Screen

Information	Source Field[a] or Screen
File name, library, and volume	Control File Specification screen
Record type	FILETYPE
Record length	RECLEN
File organization	FILEORG
Key field[b]	KEYFIELD
Alternate key paths[c]	Alternate Key Path Definition screen
Reporting permitted?	REPORT
Record update permitted?	UPDATE
Record deletion permitted?	DELETE
Comments[d]	COMMENT1, etc.

[a] All fields named in the table belong to the File Header Definition screen (Figure 2-3).

[b] Indexed and alternate indexed files only.

[c] Alternate indexed files only.

[d] This is the only place where the comments appear.

If you want a printed copy instead of a screen display of the file and field information, change the value in the Listing Device field (in the lower part of the screen) from the default SCREEN to PRINTER and press ENTER. The information on this and all following screens is immediately sent to the printer. After a "Print in progress" message, the File Header Information screen reappears. All field and file information is printed; you need not display the field information on your workstation screen in order to include it.

Note: *When the File Header Information screen reappears after you have printed the file and field information, the value in the Listing Device field is still PRINTER. Pressing the ENTER key at this point merely prints another hard copy. Instead, press PF16 to return to the CONTROL central menu, or change the value in the Listing Device field to SCREEN and press ENTER to display field information on the workstation screen.*

When you press ENTER from the File Header Information screen, the first CONTROL Field List screen appears. Figure 2-12 shows a sample of this screen.

List Control File Fields

Field Name	Start Posn	Int Fmt	Int Len	Ext Len	Dec Pos	Occur Count	Report/ Update	Upd Seq	Stamp	Cum Field	Table/ Range
■ EMPNUM	1	U	5	5	0	1	R/U	1			Range
■ SSN	6	P	5	9	0	1	R/U	2			
■ LASTNAME	11	C	15	15	0	1	R/U	3			
■ FRSTNAME	26	C	15	15	0	1	R/U	4			
■ MONTH	41	U	4	4	0	12	R	5-28			
■ REGHRS	41	U	2	2	0	1	R/U	(5)		YEARLY	
■ OTHRS	43	U	2	2	0	1	R/U	(6)		YEARLY	
■ YEARLY	89	Z	5	5	0	1	R/U	29			
■ PAYRATE	94	P	2	5	3	1	R/U	30			
■ VACATION	96	Z	3	5	1	12	R/U	31-42			PAYRTE
-Not used-	132		269								

(ENTER) Continue to modification after positioning the cursor
(1) Redisplay header (2) First (3) Last (4) Previous (5) Next (16) Return

Figure 2-12. A Sample CONTROL Field List Screen

The Field List screen displays up to 15 fields *in the order of their starting location within the record.* A pseudoblank at the left of each row is followed by the field name and 11 columns that display the characteristics of the data field as they were defined from the Field Specification Screen (Figure 2-5).

Note: *If you are looking for a particular field on the Control Field List screen (Figure 2-12), remember that all the fields are listed in order of their starting location within the record; they are not listed alphabetically. To look for a particular field alphabetically, refer to the Control Field Selection screen (Section 2.4.2) or the Field Deletion screen (Section 2.4.4). If you still cannot find a particular field, you may be looking in the wrong control file.*

Table 2-8 lists the field characteristics shown on the CONTROL Field List screen.

Table 2-8. Information Shown on Field List Screen

Column Heading	Source Field[a] and Comments
Start Posn	Starting position (first byte of this field within the data record).
Int Fmt	Internal format (Binary, Character, Packed, Unsigned, or Zoned).
Int Len	Internal length.
Ext Len	External length.
Dec Pos	Decimal positions (number of decimal positions specified -- packed, unsigned, or zoned fields only).
Occur Count	Occurrences (number of times this field occurs in the record).
Report/Update	Report and update codes. (This column combines two fields. R indicates that only reporting is allowed, U that only updating is allowed, and R/U that both are allowed. If there is no entry, neither reporting nor updating is allowed.)
Upd Seq	The update sequence -- the order in which DATENTRY presents the fields for modification -- is not specified from the Field Specification screen. Initially it corresponds to the order in which the fields are defined, but you can modify it. (See Section 2.4.7.)

(continued)

Table 2-8. Information Shown on Field List Screen (continued)

Column Heading	Source Field[a] and Comments
Stamp	Date or time stamp. If this is a stamp field, the entry indicates the format specified: Date1, Date2, D&T5, D&T6, Day, or Time. If it is not a stamp field, there is no entry.
Cum Field	Cumulative field. (If a cumulative field has been specified for which this is source field, the name of the cumulative field appears here.)
Table/Range	Table or range validation is specified not from the Field Specification screen but from the Validation Specification screen (Figure 2-6). If either has been specified for this field, this column shows either the entry "Range" or the name of the relevant table file.

[a] All fields belong to the Field Specification screen (Figure 2-5).

Since a single Field List screen can list only 15 fields, it may take several screens to list all the fields defined for a particular file. PF keys 2 through 5 let you move back and forth among screens to see all the fields in the file:

PF2 Move back to the first screen
PF3 Move forward to the last screen
PF4 Move back to the previous screen
PF5 Move forward to the next screen

You can also use PF1 to redisplay the File Header Information screen, or PF16 to return to the CONTROL central menu.

As a message at the bottom of the screen indicates, you can also initiate the field modification process from the Field List screen. Use either the tab or arrow keys to place the cursor on the pseudoblank to the left of the field you want to modify. Then press ENTER, and the appropriate Field Specification screen appears. The process of modifying fields is described in Section 2.4.2. Once you begin it by selecting a field to modify, the field listing function is completed. When you finish modifying the field, a Field Selection screen (Figure 2-10) appears, rather than a Field List screen.

2.4.6 Creating and Modifying Validation Tables (PF7)

You can create a validation table to check input into a data field while you are defining that field (as described in Section 2.3.2). However, CONTROL also provides another way to create and maintain tables through an option available from the CONTROL central menu. This option provides the only way to update an existing table by adding or deleting entries. It also enables you to create a table file whose record size is independent of the field length.

Note: *Although you can create a table through this option, DATENTRY cannot use it to validate data for a particular field until you have specified the table from the Validation Specification screen for that field. See the section "Creating a Table (PF2)," below.*

To create or modify a validation table, press PF7 from the CONTROL central menu. The CONTROL Table menu (Figure 2-13) appears.

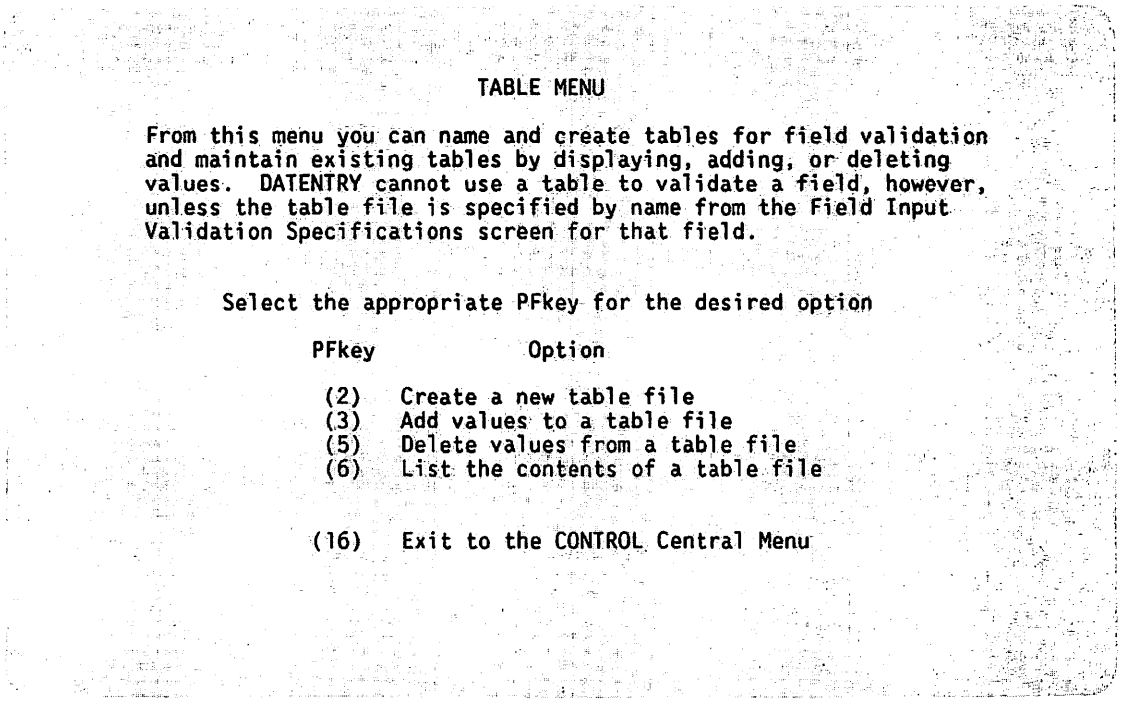


Figure 2-13. CONTROL Table Menu

Creating a Table (PF2)

To create a new table, press PF2. The CONTROL Table Creation screen (Figure 2-14) appears.

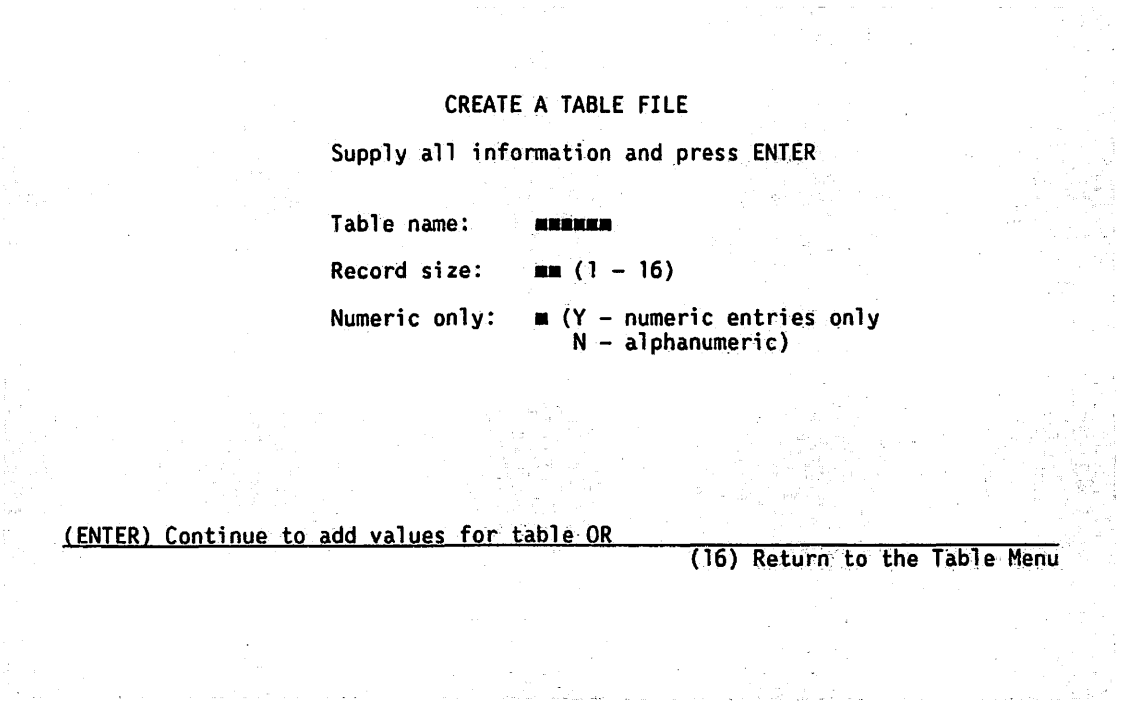


Figure 2-14. CONTROL Table Creation Screen

Enter values in the fields as follows:

Table Name -- Since CONTROL stores the values for each table in a unique file, you must first supply a file name. Enter it in this field. The table file, an indexed file containing fixed-length records, is housed in your control file library. A valid table file name must consist of alphanumeric characters (A - Z and 0 - 9), and have no embedded spaces. The name must begin with a letter, although digits are permitted in other positions.

Record Size -- Each value you supply for the table is one record in the table file. Enter the length of the longest value. If the table will be used for checking only one field, the record size should correspond to the internal length of the field, but see the next section, "Table Record Size."

Numeric Only -- Enter Y if all acceptable values are numeric and N if alphanumeric values are acceptable.

You must enter values in all three fields of the Table Creation screen. When you finish, press ENTER.

The Table Value field then appears at the bottom of the screen. (It is shown in Figure 2-15, which presents a sample CONTROL Add Table Values screen.) Its length corresponds to the record size you specified. If the table is alphanumeric, all letters entered in the Table Value field are converted to uppercase letters, but if you want to permit lowercase letters to be entered, you can do so by first pressing PF2. Enter the acceptable values for the table in this field, pressing ENTER after each one to store the value you have just entered and clear the field for the next. (It is unnecessary to enter all the acceptable values at once; you can use another option from the CONTROL Table menu to add values to a table at any time.) When you are through entering values, press PF16 to return to the CONTROL Table menu.

Before the table can be used to validate a field, you must enter its name from the Validation Specification screen (Figure 2-6) for that field. To reach the Validation Specification screen for an existing field, press PF3 from the CONTROL central menu, select the field for modification (as described in Section 2.4.2), and press ENTER when the Field Specification screen appears.

Table Record Size

When you create a table from the Validation Specifications screen for a specific field, its record size is automatically set to the internal length of that field. However, DATENTRY can check a field against a table whose records are longer than the field. For example, a table file might contain a set of valid 9-digit zip codes. DATENTRY could check a 5-digit zip code field against this table, because it compares only as many characters as the field contains, and would in this case ignore the last four characters of each table entry. The same table could therefore be used to validate 5-digit zip codes in some records and 9-digit zip codes in others. This option enables you to set the record size of a validation table independently of any specific file.

Adding Values to a Table (PF3)

To add values to an existing table file, press PF3 from the CONTROL Table menu (Figure 2-13). The Table File Specification screen (not shown) appears, prompting you for the name of the table to modify. (Alternatively, you can return to the CONTROL Table menu by pressing PF16.) The Table Name field is either blank or contains as a default the name of the table you most recently created or modified. Enter a table name if necessary and press ENTER. The CONTROL Add Table Values screen appears. (See Figure 2-15 for an example of this screen.)

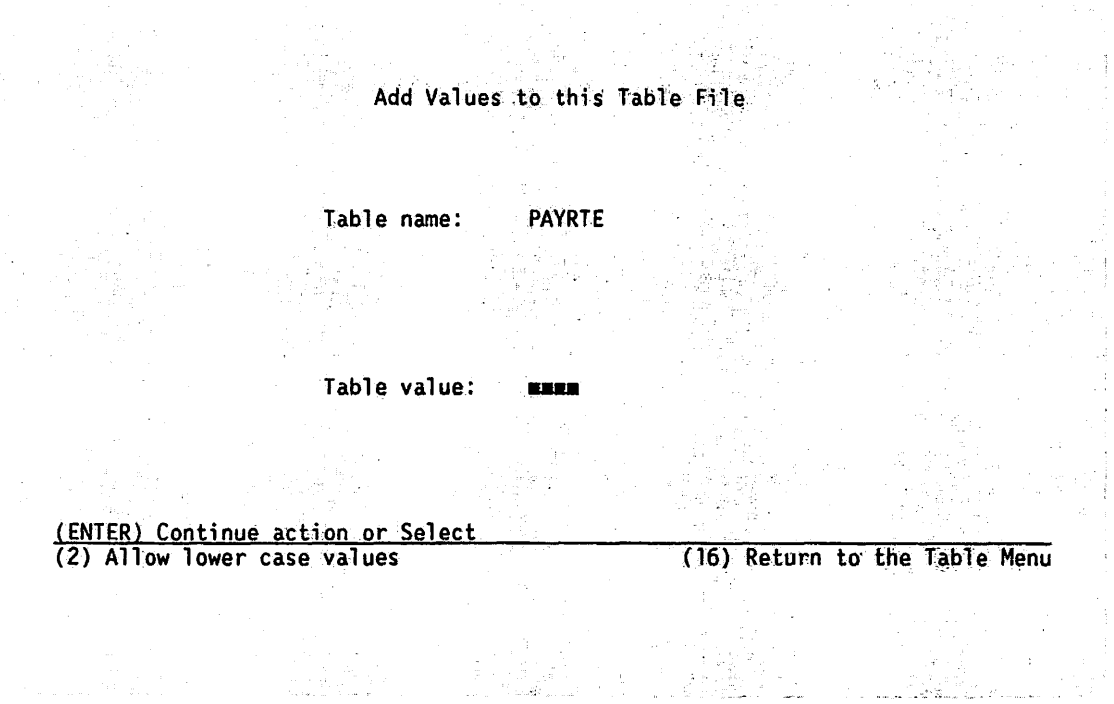


Figure 2-15. A Sample CONTROL Add Table Values Screen

This screen displays the name of the table you specified and has a Table Value field in which to enter the values you wish to add to the table. The length of this field corresponds to the table file's record size.

Enter the values you want to add to the table in the Table Value field, pressing ENTER after each one to store the value you have just entered and clear the field for the next. Letters entered in the Table Value field for an alphanumeric table are ordinarily converted to uppercase, but you can enter a value containing lowercase letters if you first press PF2.

When you finish adding values to the table, press PF16 to return to the CONTROL Table menu.

Deleting Values From a Table (PF5)

To delete specific values from a validation table, press PF5 from the Table menu. The Table File Specification screen (not shown) appears, prompting you for the name of the table to modify. (Alternatively, you can return to the CONTROL Table menu by pressing PF16.) The Table Name field is either blank or contains as a default the name of the table you most recently created or modified. Enter a table name if necessary and press ENTER. The Delete Table Values screen (not shown) appears. Except for the heading, this screen is identical in appearance to the Add Table Values screen (Figure 2-15).

Begin entering the values you wish to delete from the table in the Table Value field in the lower part of the screen. Press ENTER after each one to delete the value you have just entered and clear the field for the next. When you have deleted all the values you want to remove from the table, press PF16 to return to the CONTROL Table menu.

Listing the Values in a Table (PF6)

In the process of adding or deleting values to maintain a table, you will obviously find useful a list of the values that the table currently contains. By pressing PF6 from the CONTROL Table menu, you can obtain both a printout and a screen display showing the table header and all values in the table.

When you press PF6, the Table File Specification screen (not shown) appears, prompting you for the name of the table you want to examine. (Alternatively, you can return to the CONTROL Table menu by pressing PF16.) The Table Name field is either blank or contains as a default the name of the table you most recently created or modified. Enter a table name if necessary and press ENTER. This screen contains a Table Name field which is highlighted and contains either pseudoblanks or, as a default, the name of the table file you have most recently created or modified. Enter a table name in this field (if necessary), and press ENTER. CONTROL links to the DISPLAY utility, and a DISPLAY screen showing the first part of the table file (the header and the first few records) appears. Figure 2-16 illustrates a sample screen. The menu at the top of the screen shows how to move through the display in order to see the entire file. For detailed information about using the DISPLAY utility, see Chapter 11.

At the same time that the table file is displayed, a print file of its entire contents is automatically created. This is either printed immediately or held, according to the value set in the PRNTMODE field of your usage constants.

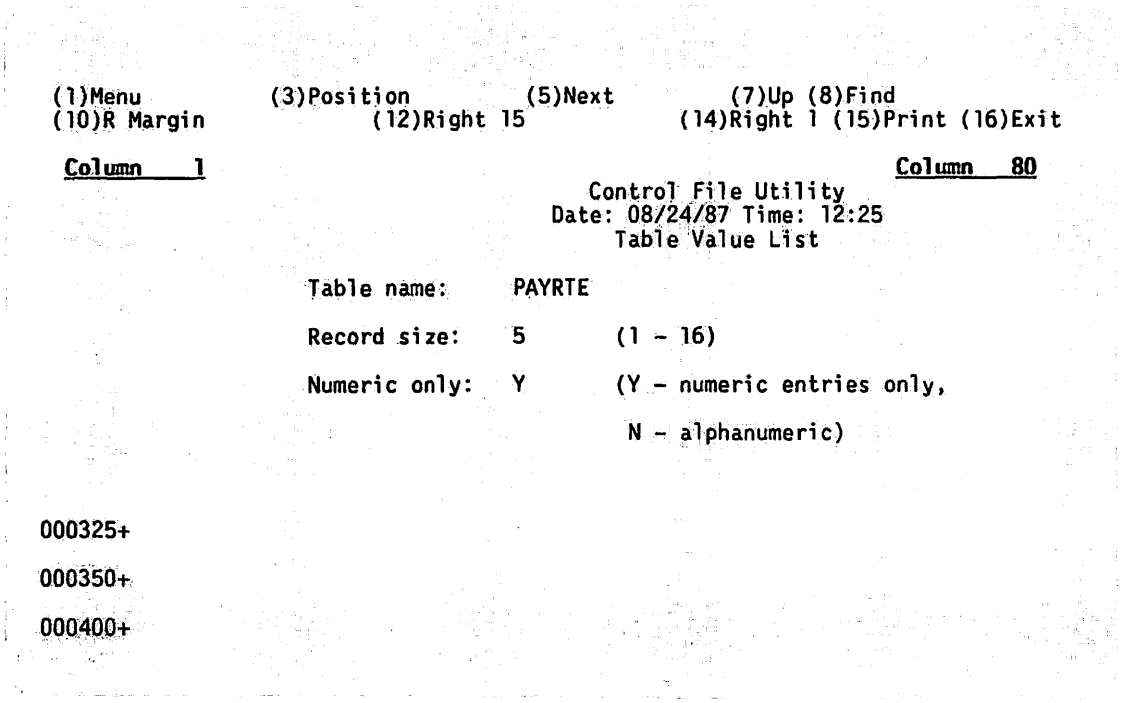


Figure 2-16. A Sample CONTROL Table File DISPLAY Screen

2.4.7 Modifying the Field Update Sequence for DATENTRY (PF12)

When the DATENTRY utility is used to create or update a data file, it displays a screen containing the names of all the modifiable fields in the record, each name followed by the appropriate number of pseudoblanks for the entry of data. You can use an option available from the CONTROL central menu (Figure 2-9) to determine the order in which these fields are displayed. If you do nothing to modify the sequence, they are displayed in the order in which the fields occur in the record (i.e., in ascending order of starting location).

To display the current update sequence and decide if you want to modify it, press PF12 from the CONTROL central menu. The first Update Sequence Modification screen appears. Figure 2-17 shows an example of this screen.

```
                UPDATE SEQUENCE MODIFICATION FOR CONTROL FILE PAYROLL

File organization is indexed.  The primary key is  EMPNUM
Records are fixed-length, record size is   400

The range of sequence numbers for this file is 01 to  42

  Old       New      Field      Start Int Int Ext Dec Occur    Table Info
Sequence Sequence   Name        Posn Fmt Len Len Pos Count   Name  # Elements
    1      ■■  01   EMPNUM        1   U    5   5   0   1
    2      ■■  02   SSN           6   P    5   9   0   1
    3      ■■  03   LASTNAME     11   C   15  15   0   1
    4      ■■  04   FRSTNAME     26   C   15  15   0   1
    5      ■■  05   REGHRS       41   U    2   2   0  12    MONTH    02
    6      ■■  06   OTHRS        43   U    2   2   0  12    MONTH    02
   29      ■■  29   YEARLY       89   Z    5   5   0   1
   30      ■■  30   PAYRATE      94   P    2   5   2   1
   31      ■■  31   VACATION     96   Z    3   5   1  12

(ENTER) Update new sequence on the screen or Select
(1) Return     (2) First     (3) Last     (4) Previous     (5) Next
(7) List by Old Seq  (8) List by New Seq  (14) Exit without Modifying
```

Figure 2-17. A Sample CONTROL Update Sequence Modification Screen

The screen lists the fields in order of the current update sequence (which is expressed numerically in the third column, immediately after the blank highlighted field, under the heading "New Sequence"). The first column (headed "Old Sequence") shows the position of each field in the update sequence before the last change entered. When the screen appears, these columns of figures are identical.

The screen shows 10 fields at a time. A given control file may have more modifiable fields than 10, and CONTROL provides as many Update Sequence Modification screens as necessary to list them. You can use PF keys 2 through 5 to move among screens as follows:

PF2 Move back to the first screen
PF3 Move forward to the last screen
PF4 Move back to the previous screen
PF5 Move forward to the next screen

To change the positions of the fields in the sequence, enter the number of the new position in the highlighted blank field opposite the name of each field you want to move. Then press either ENTER or PF8.

- When you press ENTER, the numbers in the New Sequence column change to indicate the new position of each field. The field listings remain in their original positions.

- When you press PF8, each field listing for which you specified a new position moves to that position, and the rest of the fields on the screen change position as necessary to adjust to the move. Numbers change in the Old Sequence column to show the position that each field visible on the screen occupied before the move. The New Sequence column shows the numbers in ascending order, since the fields are now listed in this sequence.

Although CONTROL can accept and incorporate as many simultaneous changes as there are fields displayed on the screen, it is usually easier to make changes one or two at a time until you have the sequence arranged to your satisfaction.

You can redisplay the original sequence (i.e., the sequence displayed when the Update Sequence Modification screen first appeared) at any time, merely by pressing PF7. If you decide that you prefer this original sequence, even though you have modified it in the meantime, you can press PF14 to return to the CONTROL central menu without saving the changes you have made.

If you decide to make your changes permanent, press PF1 to return to the CONTROL central menu and save the new sequence -- i.e., the sequence indicated by the numbers in the New Sequence column. This sequence will now determine the order in which DATENTRY displays the fields for modification, and, if you press PF12 again, the Update Sequence Modification screen will display it as the "Old Sequence."

Besides the field name, each listing also shows the following information, from left to right:

- Starting position
- Internal format
- Internal length
- External length
- Decimal positions
- Occurrence count
- Table field information (see the next section)

Repeated Fields, Table Fields, and Update Sequence

Repeated Fields (Simple Tables) -- When a field, such as VACATION, is defined as a simple table (to occur more than once in the data record), the sequence is calculated to account for all of its occurrences. In our example, the field VACATION occurs 12 times and therefore has an update sequence range of 31 through 42. These sequence numbers are reserved for the field VACATION. If, for instance, PAYRATE was given a new sequence of 31 or greater, the new sequences for the field VACATION would be 30 through 41 and 42 for PAYRATE. Whenever a new update sequence that falls within a table update sequence range is given, the field will be placed after the update sequence range. (When the same field appears on the Field List screen (Figure 2-12), the Update Sequence column contains the entry 31-42, to indicate that this field has an update sequence range including positions 31 through 42.)

Table Fields (Complex Tables) -- A complex table is a nonmodifiable field that contains one or more modifiable fields and occurs more than once in the record. (For more information about table fields, see Section 2.3.4.) A table field affects the update sequence as follows:

The modifiable fields are updated in a sequence which is repeated once for each occurrence of the table field. For example, if MONTH contains modifiable fields REGHRS AND OTHRS, and occurs 12 times, DATENTRY presents the fields for modification in the order REGHRS (01) OTHRS (01).....REGHRS (12) OTHRS (12).

CONTROL displays table field information in the three columns at the right of the screen because MONTH is a nonmodifiable field and therefore does not appear on the Update Sequence Modification screen. The first column shows the occurrence count, adjusted to account for all repetitions of the table field. The last two (jointly headed "Table Info") show the name of the table field and the number of elements it contains (counting only modifiable fields). This is how the Modification screen shows only modifiable fields; the name of the table field would not otherwise appear.

Update Sequence -- To determine the update sequence numbers for each component field, CONTROL takes the last update sequence + # of table elements * the table occur count (shown on the update sequence modification as the field occur count). As in this example, modifiable fields REGHRS and OTHRS are numbers 5 and 6 in the sequence. They are components of the MONTHLY table that has two elements and occurs 12 times. Therefore, REGHRS has the update sequence 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27; OTHRS has the update sequence 6, 8, 10, 12, 14, 16, 18, 20, 22, 26, 28; and the table field MONTH has the update sequence range 5 through 28.

Individual components of a complex table can be moved within the starting update sequence group. For example, OTHRS can be modified to update sequence 5, and REGHRS can be update sequence 6. Components of a table field cannot be split up. They can be swapped within the starting update sequence range (in this case 5 and 6) or moved as an entity. If the update sequence of the entire table field as a whole is to be modified, modify the first updatable component to the desired new update sequence. For example, if the month table is to be placed first on the data entry screen, put a 1 in the new sequence for the REGHRS field (the first update field within the table field MONTH). Both component fields will be moved accordingly. Figure 2-18 displays the Update Sequence Modification screen after the update has been effected and PF8 (list by new seq) selected. As can be seen, both REGHRS and OTHRS have been appropriately moved and resequenced.

2.5 USER EXIT SUBROUTINES

Among the specifications that define the control and data file header (Section 2.3.1), one enables you to introduce a subroutine of your own devising -- called a user exit subroutine -- into the data entry process. The user exit subroutine operates on the data after it is entered but before it is stored in the file. It can therefore be used for such purposes as

- Modifying the entered data (for instance by combining what is entered with some constant value).

- Writing a duplicate of the record, or a part of it, to another file.

- Subjecting the data to validation tests more complex than the internal table and range checking functions described in Section 2.3.3. Data validation of this kind is the usual function of user exit subroutines.

Besides writing the subroutine and specifying it in the control file, you must use the VS Linker to link the subroutine with the DATENTRY utility. The result is a linked data entry program that functions in the following way:

1. A data record, on entry, is first checked against all the validation tables and ranges you specified for its fields in creating the CONTROL file.

2. If it is accepted as valid, the record is then passed to the specified user exit subroutine. The subroutine alters the data in the record, tests it further, or does both, according to its design.

3. If the subroutine tests the record, as soon as it finds the data in one of the fields invalid, it sends DATENTRY a return code of 1. It also returns the name of the offending field and a specific error message, which DATENTRY displays. If all the data tested is valid, the code returned is 0 and the record is placed in the file.

4. The subroutine can test any number of fields in the record. When the user reads the error message and corrects the indicated field, the process begins again. If, on this pass, the subroutine finds a second field invalid, it again returns a code of 1, together with the name of this field and a different error message. The process is repeated until all the data is found valid and the return code is 0. The subroutine's operation is transparent to the user except for any error messages it returns.

The linked data entry program should be saved, since it, not DATENTRY, must henceforth be used to update this data file.

The user exit subroutine must be specified by name in the control file header (see Section 2.3.1). The following sections describe the process of writing the subroutine, linking it with the DATENTRY utility, and running the resultant program.

2.5.1 Writing a User Exit Subroutine

You can write the user exit subroutine in any of the programming languages supported by the VS. Besides conforming to the specific language's requirements for an external subroutine, it must also conform to two requirements imposed by its relationship with DATENTRY:

1. The subroutine code must identify it by a name in the range USER1 through USER10, which is identical with the user exit subroutine name specified in the header of the control file. See Table 2-9 for examples of code statements in which subroutine names are specified.

2. DATENTRY expects the subroutine to return the following four arguments, in the order shown:

   Return code (1 Byte) -- The value of the return code must be 0 (data validity confirmed) or 1 (invalid data encountered). If the value is 0, the subsequent arguments are ignored. If the subroutine does not test the data for validity, set this argument to 0.

   Note: *To avoid errors, be sure that the subroutine code explicitly sets the value of this argument to 0 when all validity tests are successful.*

Field name (8 Bytes) -- This argument gives the name of the field found to contain invalid data. It must be identical with the name used for the same field in the control file. If the return code is 1, DATENTRY blinks the name of the field on its data entry screen and places the cursor on it.

User message (64 Bytes) -- This argument passes an error message specific to the error and to the field in which it was found. (The subroutine can be written to test a field for several kinds of errors, returning an error message appropriate to each.)

Record area[1] -- The entire record (including alterations made by the subroutine, if any) is passed back to DATENTRY.

Table 2-9. Specifying External Subroutine Names in VS Programming Languages

Language	Subroutine Specification Statement
Assembly	USER5 CODE
BASIC	SUB USER5
C	extern cle void user5(arg1, arg2, arg3, arg4)
COBOL	PROGRAM ID. USER5
FORTRAN	SUBROUTINE USER5
PL/I	USER5: PROCEDURE (arg1, arg2, arg3, arg4);
RPG II	Not applicable

[1] The length of the record area argument is equal to the record length as specified in the control file header.

The following example of a user exit subroutine is written in COBOL. It tests the first character in the EMPNUM field, and, if this is a 0, sets the return code to 1 and passes the field name and user message to DATENTRY. DATENTRY then displays the message "FIRST CHARACTER MUST NOT BE ZERO" and blinks the EMPNUM field. It does not place the record in the data file. If, however, the first character in the field is not a 0, the subroutine sets the return code value to 0, and DATENTRY places the record in the file.

```
IDENTIFICATION DIVISION
PROGRAM-ID.  USER1.
ENVIRONMENT DIVISION.
DATA DIVISION.
LINKAGE SECTION.
     01  RET-URN-CODE              PIC 9.
     01  FIELD-NAME                PIC X(8).
     01  USER-MESSAGE              PIC X(64).
     01  RECORD-AREA.
          03  FILLER                     PIC X(4).
          03  EMPNUM.
               05  FIRSTC                PIC X.
               05  FILLER                PIC X(4).
          03  FILLER                     PIC X(71).
PROCEDURE DIVISION USING RET-URN-CODE
                         FIELD-NAME
                         USER-MESSAGE
                         RECORD-AREA.

BEGIN-IT SECTION.
VALIDATE-IT.
     IF FIRSTC IS EQUAL TO "0"
          MOVE 1 TO RET-URN-CODE
          MOVE "EMPNUM" TO FIELD-NAME
          MOVE "FIRST CHARACTER MUST NOT BE ZERO" TO USER-MESSAGE
     ELSE MOVE 0 TO RET-URN-CODE.
     EXIT PROGRAM
```

2.5.2 Linking DATENTRY to the User Exit Subroutine

Before you can use a newly written user exit subroutine, you must create a new and specific data entry program by using the VS Linker to combine the subroutine with the DATENTRY utility. This procedure creates a new object program file, which must be given a new name. Using this name, you can run the data entry program from the Command Processor menu or from a procedure.

For detailed information about running the Linker, see the *VS Linker Reference*. The following is a sketch of the procedure to follow:

1. Run the Linker by pressing PF1 from the Command Processor menu.

2. Since DATENTRY contains no subroutines, you should leave the first Linker option screen unchanged (unless the user exit subroutine itself contains additional subroutines).

3. Specify the two object files to be linked in the order of their execution: first DATENTRY, then the user exit subroutine.

4. Specify as an output file the name and location you wish to assign to the new data entry program. You can use any valid VS file name for the program.

5. The link is successful if a system return code of 4 or less is returned.

2.5.3 Running the Linked Data Entry Program

To run the data entry program you have created by linking DATENTRY with your user exit subroutine, press PF1 (Run Program or Procedure) from the Command Processor menu and specify the new program by the name and location you supplied for the output file when you ran the Linker.

This program processes information in the same manner as DATENTRY, with the addition of the subroutine functions. The way it handles data entry is described in detail at the beginning of Section 2.5. It offers all DATENTRY's other options (described in Chapter 3) as well.

Note: *A data entry program created by linking the DATENTRY utility with a particular user exit subroutine must be used to process any data file whose control file specifies that subroutine. The simple DATENTRY utility (i.e., not linked with a subroutine) cannot process a data file whose control file specifies a user exit subroutine. You can, however, create another control file that also describes the data file but does not specify any user exit subroutine. DATENTRY can then process the data file as long as this new control file is specified.*

2.6 Creating Source Code From a Control File

The CONTROL utility, by linking to a language compiler, can generate source code that describes a control file and its fields in the COBOL, RPG II, and PL/I programming languages. Although these fragments of code are not programs in themselves, you can insert them in programs, either by using the XCOPY function of the VS Editor (see the *VS Editor Reference*) or by placing a COPY statement in the program code.

To generate source code from a control file, press PF8 from the CONTROL central menu (Figure 2-9). The CONTROL Source Language menu (Figure 2-18) appears.

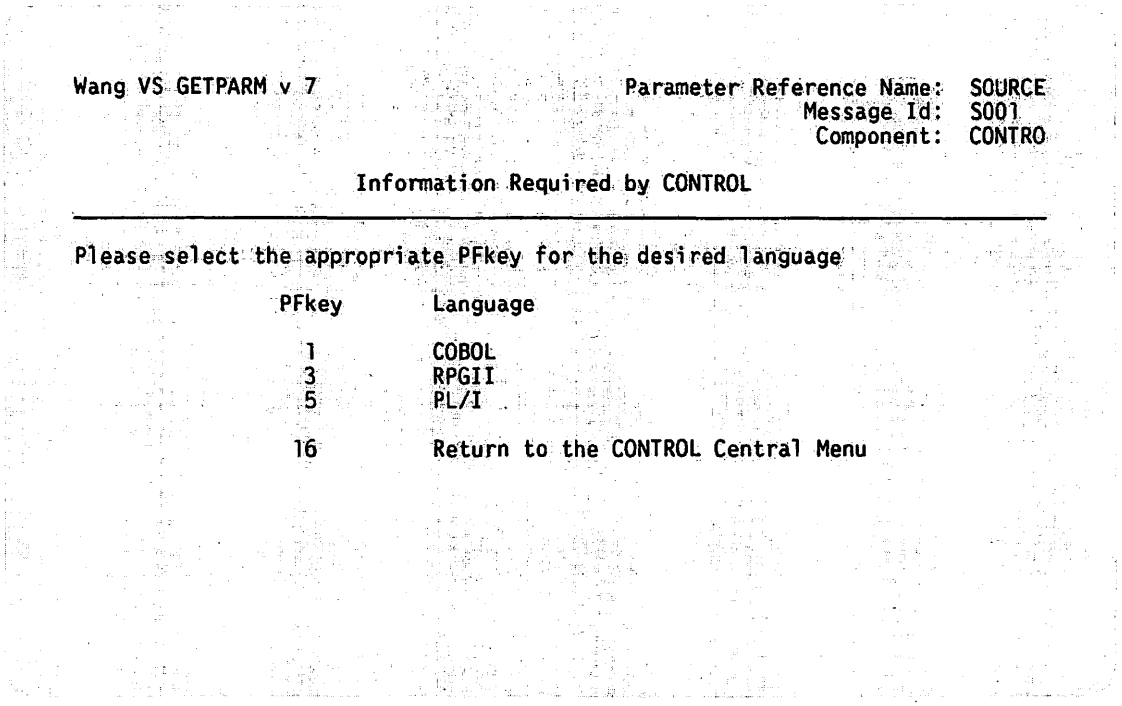


Figure 2-18. CONTROL Source Language Menu

Select COBOL as the source language by pressing PF1, RPG II by pressing PF3, or PL/I by pressing PF5. PF16 provides a way to abort the process and return to the central menu.

When you select a language, the Source File Definition screen (not shown) appears. This screen contains File, Library, and Volume fields in which you specify a name and location for the file that will contain the source code. As soon as you enter the file specifications and press ENTER, the file is created and the CONTROL central menu reappears.

Note: *For code generation to take place, the compiler for the specified source language must be present in the system library.*

2.7 Running Other Utilities From Within CONTROL

The other data file management utilities -- DATENTRY, REPORT, INQUIRY, and EZFORMAT -- are directly accessible from the CONTROL central menu (Figure 2-9). The following list shows the PF Key associated with each utility and the chapter in this reference that describes its operation.

PF Key	Option	Chapter
9	Run DATENTRY	3
10	Run REPORT	5
↑10 (26)	Run INQUIRY	6
11	Run EZFORMAT	4

When you press one of these PF keys, the initial screen of the indicated utility appears. When you finish running the utility and press PF16 to exit, you are returned to the CONTROL central menu instead of the Command Processor menu.

When you run another utility from within CONTROL in this manner, the current control file -- i.e., the one you specified from the Control File Specification screen (Figure 2-1) -- is automatically specified for processing by that utility. (You must still specify one or more data files, however.) You can change the control file specification for the second program, but this change has no effect on CONTROL. When you exit from that program and return to the CONTROL central menu, the control file that was specified at the time you began running the second program is still current.

CHAPTER 3
DATENTRY

3.1 INTRODUCTION

The DATENTRY utility enables you to create a data file in the format defined by its related control file. (For more information about control files, see the description of the CONTROL utility in Chapter 2.) Using the information in the control file, DATENTRY displays the modifiable fields for record addition, modification, or deletion. The order in which the fields are displayed is determined by their update sequence as it is defined in the control file. The manner in which they are displayed (underlined, bright, blinking, single, or double spaced, etc.) can be modified in DATENTRY. If a more customized display is desired, you can generate a screen display for use by EZFORMAT (see Chapter 4). The EZFORMAT utility allows the user to customize the data entry screen and produce a data entry program uniquely designed for a particular data file.

3.1.1 Software Requirements

For DATENTRY to run correctly, the INQUIRY and DISPLAY utilities must reside in the system library or in the same library as DATENTRY.

3.1.2 Running DATENTRY

DATENTRY can be run from the CONTROL Utility main menu or the Command Processor. From the Command Processor menu, press PF1 (RUN Program or Procedure) and specify DATENTRY in the Program field. From the CONTROL Utility main menu, press PF9.

The following sections describe DATENTRY processing:

Section	Process
3.2	Specifying Control and Data Files
3.3	Creating a New Data File
3.4	Building or Maintaining a Data File
3.4.1	Adding Records
3.4.2	Modifying Records
3.4.3	Deleting Records
3.4.4	Displaying a Data File
3.4.5	Modifying the Data Entry Screen Display
3.4.6	Saving the Screen Contents for EZFORMAT
3.4.7	Running the INQUIRY Utility

3.2 SPECIFYING CONTROL AND DATA FILES

When DATENTRY processing begins, The Data and Control File Specification screen appears, prompting you for the names and locations of the related data and control files. (Figure 3-1 shows a sample of such a screen.) If DATENTRY has been run from CONTROL, the control file information will already be filled in; this will include the related data file if it was specified. If the names and locations of related data and control files are not present, they can be provided at this time. If modifications are needed, they can also be made at this time.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  INPUT
                                                  Message Id:  0001
                                                   Component:  DATENT

                    Information Required by DATENTRY

Specify the data and control files and press ENTER to continue to the menu.

To obtain a list of existing control files, supply the library and volume
names only and press ENTER.

Data file:
FILE     = PAYROLL■ LIBRARY  = USRDATA■ VOLUME   = VOL111

Control file:
CTLFILE  = PAYROLL■ CTLLIB   = USRCTL■■ CTLVOL   = VOL111

(ENTER) Continue to Menu/View list of files or Select
(2) Create a new Data file
                                                  (16) Exit DATENTRY
```

Figure 3-1. A Sample DATENTRY Data and Control File Specification Screen

This screen contains two sets of file specification fields, one for the data file and one for the control file that describes it. If the information was not previously supplied or is to be modified, enter the file specification in the fields as indicated in the following paragraphs.

3.2.1 Data File Fields

FILE -- To specify a particular data file (either an existing file to be modified or examined or a new file to be created), enter the name in this field. A valid file name must be from 1 to 8 characters in length and may not contain embedded blanks. Valid characters are A through Z, 0 through 9, @, #, and $.

LIBRARY -- Enter the name of the library where the data file is currently residing or is going to reside. A valid library name follows the same conventions as a valid file name. If you have specified the INLIB field of your usage constants (either through the Set Usage Constants function [PF2] of the Command Processor or through a procedure), the value from that field appears in this field as a default.

VOLUME -- Specify the volume where the data file is currently residing or is going to reside. If the INVOL field of your usage constants is set, the value from that field appears in this field as a default.

3.2.2 Control File Fields

CTLFILE -- Enter the name of an existing control file that describes the data file you want to maintain or create. DATENTRY requires a control file, which must be created by the CONTROL utility (described in Chapter 2 of this manual).

If you want to choose from a list of the files in your control file library, you can leave this field blank and press ENTER. DATENTRY then links to the CONTROL utility, which displays the first Control File List screen. Section 2.2.1 describes these screens and tells how to locate and select a particular control file from them.

If you invoked DATENTRY by pressing PF9 from the CONTROL File Header Definition screen (Figure 2-3), the name of the control file you last specified within the CONTROL utility appears as a default value in this field.

CTLLIB -- If a value was not supplied from the CONTROL utility, the program supplies a default control file library name in this field by concatenating your user ID with the string *CTL*, e.g., USRCTL.

CTLVOL -- Specify the volume on which the control file resides. If you have specified the INVOL field of your usage constants (either through the Set Usage Constants function [PF2] of the Command Processor or through a procedure), the value from that field appears here as a default.

You can override the default in any field by entering a new value.

Note: *If the file specifications are supplied from the CONTROL utility, you can change them for DATENTRY processing and not affect CONTROL. For example, the control file PAYROLL is changed to ACCNT on the DATENTRY File Specification screen. When you return to CONTROL, the control file is still PAYROLL. If you again run DATENTRY from CONTROL, the control file PAYROLL will again appear as the default. Any modifications to the file specification information are lost when the user selects PF16 to exit DATENTRY.*

When you have supplied all the necessary information in the fields of the Data and Control File Specification screen, you have the following options:

- To create a new data file, press PF2. (Section 3.3 describes the creation process.)
- To build, examine, or maintain an existing data file, press ENTER. The DATENTRY central menu offers several options, described in Section 3.4.
- To exit from DATENTRY without further processing, press PF16.

3.3 CREATING A NEW DATA FILE

When you press PF2 from the Data and Control File Specification screen (Figure 3-1) to indicate that you want to create a new data file, DATENTRY checks that the file does not already exist.

- If you specified a file that already exists, DATENTRY determines if you have access rights to scratch the existing file. If you do not have access, you will be informed and asked to respecify your data file specifications. If you do have access to scratch the existing file, you will be requested to scratch it by pressing PF3 or respecify the data file specifications. The scratch does not take place until PF3 has been selected. If you decide not to choose either option, you can select PF16 to exit DATENTRY.

- When creating the new data file, DATENTRY displays the Data File Creation screen. (Figure 3-2 shows a sample of this screen.) A line indicates the specified control file and data file information. This information cannot be modified on this screen. The defaults which are modifiable are in bright mode (number of records, number of days to retain the data file, release unused space, and file class). If the file specification information needs to be modified, or you wish to exit the creation mode completely, press PF1 to return to the Data and Control File Specifications screen. Press ENTER to create the data file after reviewing and/or modifying the parameters.

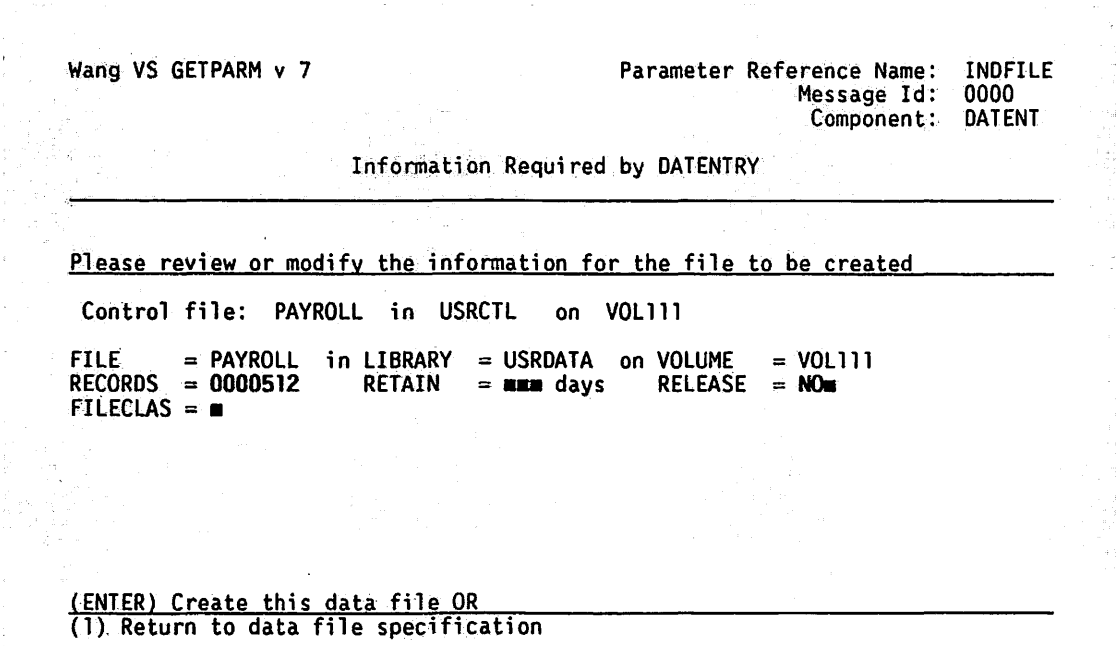


Figure 3-2. A Sample DATENTRY Data File Creation Screen

The following list describes the modifiable fields found on the DATENTRY Data File Creation screen.

RECORDS -- The value in this field should equal the approximate number of records the data file is to contain. The default value, as shown, is 512.

RETAIN -- At your option, you can specify the number of days until the file's expiration date. The file cannot be deleted, either from the command processor or by a procedure, until this date is reached. The maximum is 999 days. The default value is 0 -- the field is blank -- meaning that the file can be deleted at any time.

RELEASE -- Specify whether you want unused extents (i.e., extents of disk space allocated to the file, but in which no data is written) to be released for use by other files when this file is closed. The default is NO.

FILECLAS -- This field is for the default file protection class. (See the *VS System User's Introduction* for information about file protection classes.) If the FILECLAS field of your usage constants is set, that value appears here as a default. Otherwise, both the field and the default file protection class are blank.

Enter or change information in these four fields as necessary, and press ENTER. The DATENTRY central menu (Figure 3-3) appears.

The data file now exists, but as yet it contains no records. You can begin storing records in the file by choosing the "Add Records" option from the menu. Since this process is the same for new and existing data files, it is described in the next section, "Building or Maintaining a Data File."

3.4 BUILDING OR MAINTAINING A DATA FILE

When you finish specifying data and control files and press ENTER (from either the Control and Data File Specification screen or the Data File Creation screen), the DATENTRY central menu (Figure 3-3) appears.

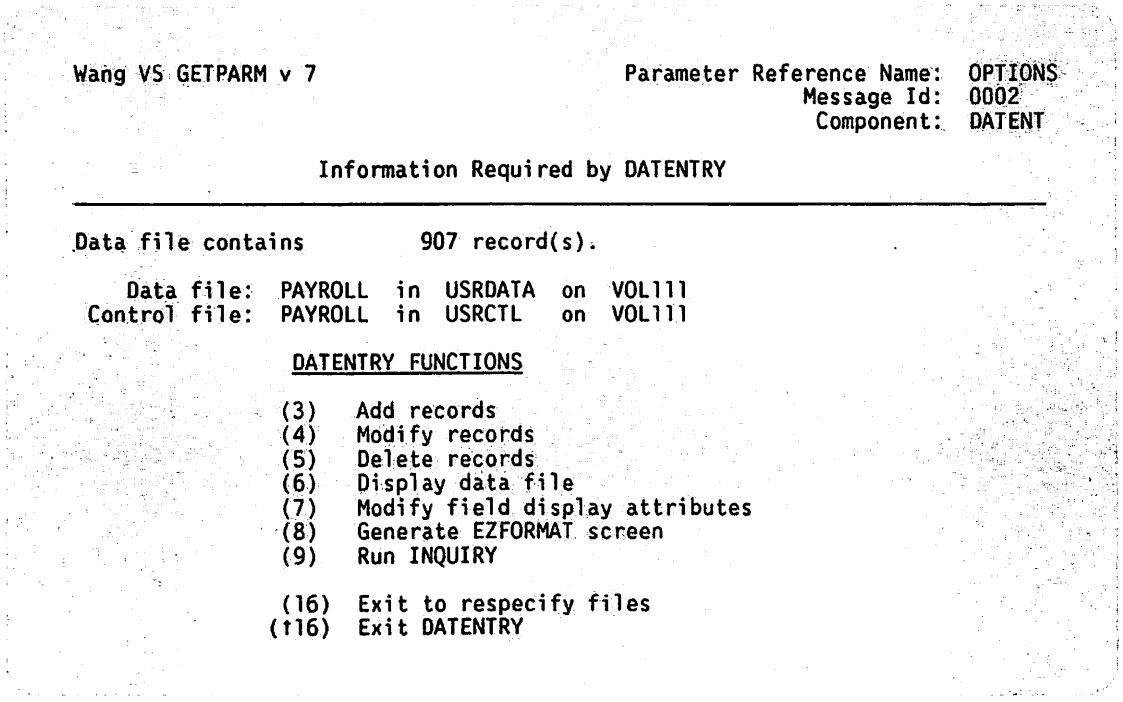


Figure 3-3. DATENTRY Central Menu

Note: *The full list of menu options does not appear for every file. When a data file has just been created and contains no records, the only options available, and the only ones to appear on the list, are Add records, Modify field display attributes, Generate EZFORMAT screen, and Exit Datentry or Exit to respecify files. The Delete records option is never available for consecutive files since their records cannot be deleted.*

To choose an option from this menu, press the appropriate PF key. When a function is completed and the menu reappears, the line that identifies the completed function is highlighted.

PF16 returns you to the Data and Control File Specification screen (Figure 3-1), from which you can specify new data and control files and run DATENTRY again. PF32 (↑16) terminates DATENTRY processing.

The following list describes the DATENTRY central menu options individually.

PF Key	Option and Description
3	**Add records** -- Enables you to enter new records in an existing data file. Section 3.4.1 describes this process.
4	**Modify records** -- Retrieves individual records from the file for modification. Section 3.4.2 describes this process.
5	**Delete records** -- Enables you to delete individual records from the file. Section 3.4.3 describes this process.
6	**Display data file** -- Links to the DISPLAY Utility so that you can examine the contents of the data file on your workstation screen. Section 3.4.4 describes this process.
7	**Modify field display attributes** -- Lets you make certain changes in the way DATENTRY displays fields on the data entry screen. Section 3.4.5 describes this process.
8	**Generate EZFORMAT screen** -- Saves the contents of the data entry screen in a form accessible to the EZFORMAT utility, which can be used to edit and rearrange them. Section 3.4.6 describes this process.
9	**Run INQUIRY** -- Links to the INQUIRY utility so that you can query the data file using an English-like syntax, and either print or display the result. Section 3.4.7 describes this process.

3.4.1 Adding Records to a File (PF3)

Whether the data file already contains records or has just been created by the procedure described in Section 3.3, you begin the process of entering new records by pressing PF3 (Add Records) from the DATENTRY central menu. A Data Entry screen appears (see Figure 3-4 for an example of this screen).

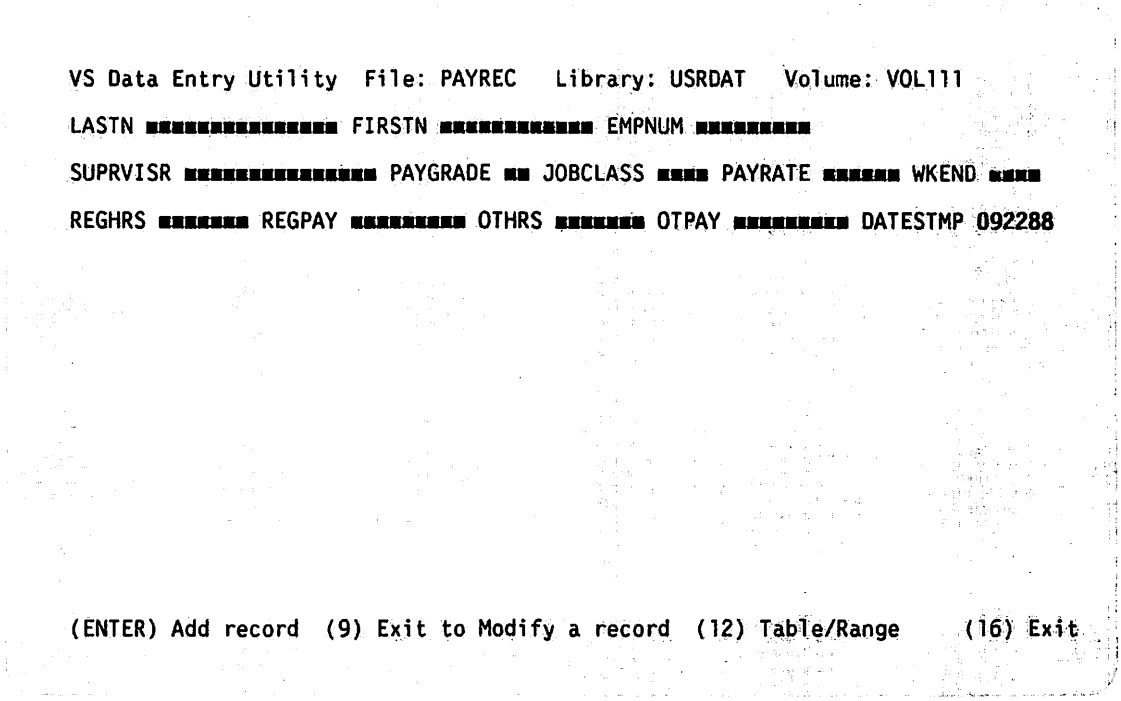


Figure 3-4. A Sample DATENTRY Data Entry Screen

This screen displays all the modifiable fields in the record, in order of the update sequence specified in the control file. The field name appears before a field of highlighted pseudoblanks (equal in number to the external length of the field as specified or calculated in the control file). You can enter data in these blank fields, using the TAB key to move from one field to the next. When you have specified all the data, press ENTER to store the record in the file.

If you specified data validation in the control file (as described in Sections 2.3.3 and 2.5), DATENTRY does not store the record until the validation procedure is successfully completed. If the data is not accepted as valid when you press ENTER, the fields on your screen, instead of being cleared, continue to show the values you entered. The cursor is placed on the first field in the sequence that contains invalid data; this field is made to blink, and an error message appears at the top of the screen. Correct the value in the field and press ENTER again. When all the data is valid, DATENTRY stores the record and initializes the screen for the next addition. You can add as many records as you wish (until you reach the maximum number specified at the creation of the data file).

The following options are available from the Data Entry screen.

PF Key	Option and Description
ENTER	**Add Record to EOF** -- This option adds the record to the current end of the data file for consecutive and relative files. It adds the record by primary and (if specified) alternate key sequence for indexed or alternate indexed files. The screen is reinitialized and readied for the next addition.
3	**Add Relative Record Number (RRN)** -- *This option applies only to relative files, and appears on the screen only when a relative file is specified.* At the bottom of the screen is a blank field in which you can enter a specific relative record number. When you press PF3, the record is stored in the appropriate record slot, and the screen is initialized for the next record addition.
9	**Exit to modify a record** -- This option, which allows you to locate an existing record in order to modify it, is the equivalent of pressing PF4 from the DATENTRY central menu. This function is described in Section 3.4.2.
12	**Table/Range** -- If you press PF12, DATENTRY displays a screen listing all the fields in the record for which either table or range validation has been specified. Upper and lower range values are also shown. To see the values allowed for any table, place the cursor on the pseudoblank before its name and press ENTER.
16	**Exit** -- Stop adding records and return to the central menu.

The Format of the Data Entry Screen

DATENTRY displays fields in the specified update sequence, breaking lines wherever necessary, *but only between fields*. Neither fields nor field names are aligned vertically; instead, names and blank fields within a line are always separated by single spaces.

If there is enough room, DATENTRY double-spaces the lines on the screen, leaving a blank line after each line of fields, but if more than 11 lines are needed to show all the fields, the screen is single-spaced. DATENTRY can put a maximum of 21 lines of fields on the screen. The number of fields that will fit in this space depends primarily on how long both the fields and their names are. To a lesser extent it also depends on the order in which they appear, since DATENTRY can break a line only between fields.

Note: *DATENTRY displays only one data entry screen for each record. It therefore cannot work with a control file that specifies fields too numerous or too long to fit on a single screen. As soon as such a control file is specified, an error message reports a screen overflow condition. That control file cannot be used with DATENTRY unless it is changed so that DATENTRY can display the full set of modifiable fields it specifies on one screen.*

Through an option available from the DATENTRY central menu (PF7, Modify field display attributes, described in Section 3.4.6), you can change the spacing from double to single or vice-versa. (Changing from single-spacing to double is possible only if there are 11 or fewer lines of fields.) The same option enables you to make certain changes in the way fields are displayed. However, you cannot use DATENTRY to change field length or position, or to make any other changes in the screen.

You can change the order in which the fields appear by running the CONTROL utility to respecify the update sequence. CONTROL can also change the external length and format of specific fields. Other changes in screen arrangement are possible only if you run the EZFORMAT utility to create a data entry program custom-tailored for the data file. You can use EZFORMAT to arrange the position of fields on the screen, add text captions, and so on. Chapter 4 describes EZFORMAT.

How DATENTRY Displays Information

Figure 3-4 shows how DATENTRY displays various types of information such as single fields, repeated fields, table field components, a cumulative field, and a field which will not be cleared after every addition. Single fields such as EMPNUM and SSN appear only once. Repeated fields and table field components appear as many times as they have been specified to occur within CONTROL. They are immediately apparent, because they are followed by a set of parentheses which contain the proper sequence number for that element within its grouping (i.e., VACATION (01)). Repeated fields such as VACATION appear as many times as the field is to occur. Table field components appear together one after the other in their proper grouping as many times as their table field is to occur.

Our example shows that the table field MONTH occurs 12 times. Fields REGHRS and OTHRS are component fields and appear one after the other for 12 times. The field YEARLY was defined in CONTROL as being a cumulative field for the fields REGHRS and OTHRS. If displayed, cumulative fields will always appear as nonmodifiable to preserve their integrity. On addition screens cumulative fields are always blank, because the cumulative process has not yet been accomplished. PAYRATE is another field with special display attributes. When defined in CONTROL the display code was assigned a "1" for display last value. This means that during record addition, the PAYRATE field will always display the value entered on the previous screen. It may be modified at any time. This is used in our example primarily to help insure accuracy and speed during data entry. Once an accurate pay rate has been entered, it need not be entered again until a modification is required.

3.4.2 Modifying Records in a File (PF4)

To recall a record from your data file for modification, you can press PF4 from the DATENTRY central menu or, if you are in the process of adding records, PF9 from the data entry screen. The structure of the selection screen which follows is determined by the file type it represents. The next three sections describe the process of selecting records from consecutive, relative, indexed, and alternate indexed files.

Consecutive Files

When you press a PF key to indicate that you want to modify records in a consecutive file, DATENTRY displays the Consecutive File Record Specification screen. (Figure 3-5 shows a sample of this screen.)

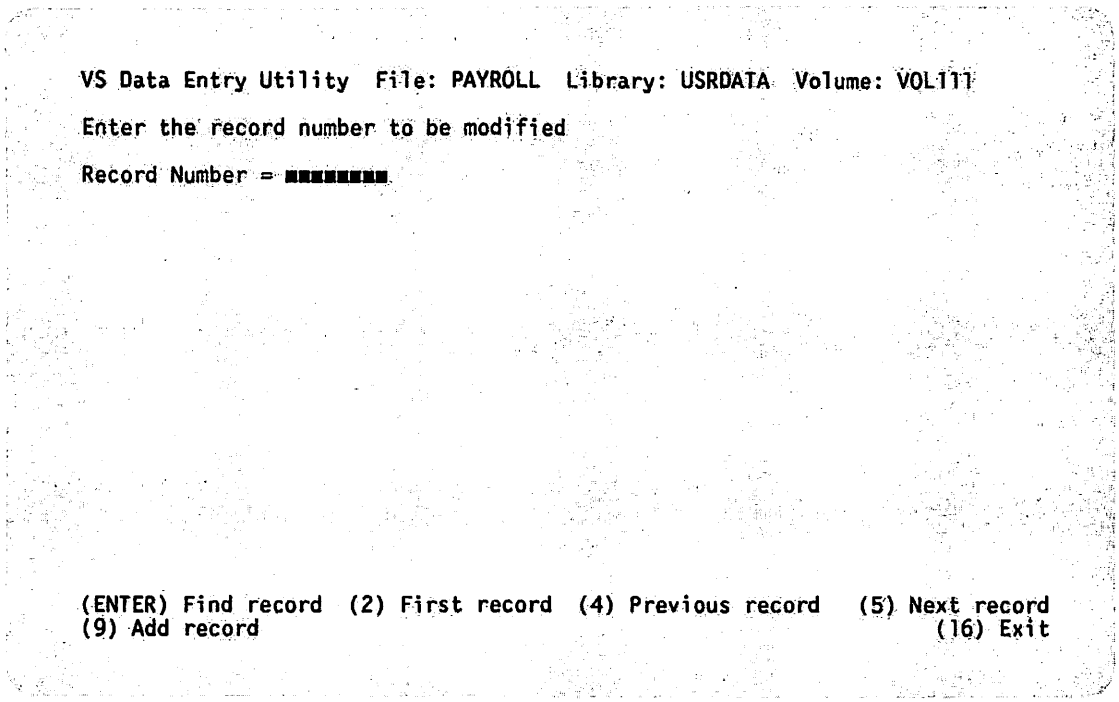


Figure 3-5. A Sample DATENTRY Consecutive File Record Specification Screen

Records in consecutive files are located by number (i.e., by the record's position in sequence from the beginning of the file).

The following options are available from the DATENTRY Consecutive File Record Specification screen:

PF Key	Option and Description
ENTER	**Find record** -- Select the specified record.
2	**First record** -- Select the first record in the file.
4	**Previous record** -- Move back to the last record entered or modified.
5	**Next record** -- Move forward one record.
9	**Add record** -- Switch from modifying to adding records, as described in Section 3.4.1.
16	**Exit** -- Return to DATENTRY central menu.

Relative Files

When you press PF4 from the DATENTRY central menu or PF9 from a data entry screen to indicate that you want to modify records in a relative file, DATENTRY displays the Relative File Record Specification screen (not shown). With the exception of an additional option provided in the abbreviated menu at the bottom of the Relative File Record Specification screen, the relative and consecutive file screens shown when PF4 and PF9 are pressed are identical.

In consecutive files, records are added one after the other and cannot be deleted. Therefore, the record number represents a position in a consecutive sequence of records. In relative files, records are also associated with a record number. Record numbers in relative files correspond with a physical location or slot within the data file itself. Relative file records can be added one right after the other (to the end of the file), or randomly by relative record number. When specifying a relative record number, you are requesting information from that physical location or slot in the data file. If information has not been written for that record number, you will be informed that the record number cannot be found although the slot does in fact exist.

PF6 (Find record by Inexact RRN) has been added to the abbreviated menu because the exact relative record might not be known for a particular record. The selection of this PF key will allow the user to search the relative file and display for modification the relative record that has the relative record number equal to or greater than the number specified. If the number requested is past the last relative record number used in the file, the user will be warned and asked to respecify with another number. All other PF keys on the abbreviated menu are identical to those described above for the Consecutive File Record Selection screen.

Indexed Files

When you press PF4 from the DATENTRY central menu or PF9 from a data entry screen to indicate that you want to modify records in an indexed file, DATENTRY displays the Indexed File Record Specification screen. (An example of this screen is shown in Figure 3-6.)

```
VS Data Entry Utility  File: PAYROLL  Library: USRDATA  Volume: VOL111

Enter the Primary key of the record to be modified

 Record key =
 EMPNUM ■■■■■■

(ENTER) Find Record     (2) First record  (4) Previous record  (5) Next record
(6) Find by Inexact key                   (9) Add record           (16) Exit
```

Figure 3-6. A Sample DATENTRY Indexed File Record Specification Screen

Records in indexed files are specified by the value of the primary key. Key values may be made up of more than one field name for ease of input. Up to 10 field names can be displayed on the Indexed File Record Specification screen. If more than 10 field names were defined in CONTROL to describe the primary key, the user is given a warning message that the entire key is not shown. In this case it is recommended that PF6 (described below) be selected.

The following options are available from the Datentry Indexed File Record Specification screen:

PF Key	Option and Description
ENTER	**Find record** -- Select the specified record.
2	**First record** -- Select the first record in the file.
4	**Previous record** -- Move back to the last record entered or modified.
5	**Next record** -- Move forward one record.
6	**Find by Inexact key** -- Find a record whose primary key value has a matching character string of equal or greater value. For example, John Smith's paperwork has been damaged due to a water spill, and his employee number is not quite legible. You can see that is starts with a 56. By requesting employee number 56000 you will be positioned in the file at the first employee record that starts with this number. In this example John Smith's record is not the first one found. The record found belongs to Charlie Doe, whose employee number is 56009. The file was positioned to the number nearest to 56000 in ASCII collating sequence. If attempting to find a record by this method does not provide the expected results, check to insure that the record being sought has a primary key which is equal to or greater than the value specified in the ASCII collating sequence. Now to find John Smith's record, you would select PF5 to proceed record by record until John Smith's record is found.
9	**Add record** -- Switch from modifying to adding records, as described in Section 3.4.1.

Alternate Indexed Files

When you press PF4 from the DATENTRY central menu or PF9 from a data entry screen to indicate that you want to modify records in an alternate indexed file, DATENTRY displays the Key Path Selection screen. (Figure 3-7 shows a sample of this screen.)

```
Wang VS GETPARM v 7                          Parameter Reference Name:  PATH
                                                          Message Id:  0003
                                                           Component:  DATENT

                          Information Required by DATENTRY

Define the key path for the data file.

Enter the key path for reading the data file and press ENTER, or
press the PFkey relating to the desired path number
Use (↑1) to define the Primary key as the path.

The primary key is EMPNUM

PATH      = EMPNUM■■

      Path     Dups
(1) SSN        NO
(2) LASTNAME   YES


Press (↑16) Exit
```

Figure 3-7. A Sample DATENTRY Key Path Selection Screen

Alternate indexed files can be located by the primary or any one of the alternate keys defined through CONTROL. The key selected to access the data file is referred to as the key path. This also determines the sorted order in which the records will be displayed. Alternate indexed files can be accessed by either a primary or an alternate key and are used when it is desirable to access data sequentially in a different order. As in the situation discussed under "Find by Inexact key" (in the section on Indexed Files), it is often convenient to have other methods by which to locate and access information. If our file PAYROLL has been defined as an alternate indexed file (Figure 3-8), we can access information by social security number or last name in addition to the employee number. In our example, had we known John Smith's social security number, we could have chosen that path and entered his number for immediate access. Or we could have chosen last name as the path, entered his name, and started with the first Smith in the file.

Because each social security number is unique to an individual and has duplicates set to "NO," we are insured that if the one we specified is found, it is for John Smith. However, many people can have the last name and therefore duplicates are allowed. As in our previous example where we used John Smith's employee number and in our current example using the last name, the record displayed may not be the one we are looking for, and we may have to page (using PF5) into the file until it is located. As Figure 3-7 shows, upon initial entry into the Key Path screen, the path will always be the primary key name. Upon returning to this screen this field will contain the field previously selected as the key path. Below the path name the alternate keys are listed, along with corresponding PF key numbers. To select a path, type in the key field name and press ENTER or select its associated PF key number. The primary key is always PF↑1 (PF17) while the alternate PF keys correspond with the order in which the alternate keys are listed in CONTROL.

As soon as you select a path by pressing ENTER or you select one of the PF keys, DATENTRY displays a Record Specification screen. It is identical to the Indexed File Record Specification screen (Figure 3-6), except that the abbreviated menu at the bottom contains one more option: PF7 ("Change path"). By pressing this key you return to the Key Path Selection screen.

Except for the additional step of selecting a key path, the process of modifying records is exactly the same for indexed and alternate indexed files (see the preceding section).

Record Modification Screen

As soon as you complete the record specification and select the appropriate PF key, a record modification screen appears. It is similar to a data entry addition screen, except that the fields contain the data from the record you specified rather than being blank. As on all screens, the bottom of the screen is reserved for record information and the abbreviated menu options. The file organization determines which options and information are displayed.

The following are standard PF key options for all file organizations.

PF Key	Option and Description
ENTER	**MODIFY** -- Modify the current record being displayed and proceed to the next one.
1	**Find (Fnd)** -- Return to the Record Specification screen to enter a new record number/key path value. (For consecutive/relative files, the number of the record just selected appears as a default.)

PF Key	Option and Description
2	**First record** -- Select the first record in the file.
4	**Previous record** -- Move back to the last record entered or modified.
5	**Next record** -- Move forward one record.
9	**Add record** -- Switch from modifying to adding records, as described in Section 3.4.1.
12	**Table/Range (Tbl/Rng)** -- List the fields for which table or range validation has been specified and show table values or ranges (this option is described in more detail in Section 3.4.1).
16	**Exit** -- Return to the DATENTRY central menu.

Consecutive and Relative Files

The record number of the record appears as the first entry on the abbreviated options line, followed by the PF key options listed above.

Indexed and Alternate Indexed Files

Standard indexed files have only the standard PF key options listed above. Alternate indexed files also have the additional option PF7 ("Change path"). Pressing this key returns you to the Key Path Selection screen.

3.4.3 Deleting Records From a File (PF5)

You can delete records from relative, indexed, and alternate indexed files by pressing PF5 from the DATENTRY central menu (Figure 3-3). Since records cannot be deleted from consecutive files, this option is absent from the menu when you specify a consecutive file.

Except for the end result, deleting records is so similar to modifying them that the process is merely summarized below. Records are specified for deletion in ways that differ according to file organization. For details, see Section 3.4.2.

- For a relative file, you specify the record by its relative record number.
- For an indexed file, you specify the record by its primary key value.

- Deleting a record from an alternate indexed file is much the same as for an indexed file, except that you must first select the key path along which to search for the record. When you have chosen the key, you specify the value.

Once the record is specified, a record deletion screen (similar to the data entry and data modification screens) appears. Verify that the record displayed on this screen is the one you want to delete, and press ENTER. DATENTRY deletes the record and displays the next one in the relative file, or the next one on the same key path in an indexed or alternate indexed file.

Except that you enter no data from the deletion screen, the entire process of deleting records is nearly identical to the process of modifying them. The only other difference is that none of the screens that appear during the deletion process offers the alternative of switching directly to the "Add records" option. To select the latter you must exit to the DATENTRY central menu.

3.4.4 Displaying a Data File (PF6)

When you press PF6 from the DATENTRY central menu (Figure 3-3), DATENTRY links to the DISPLAY utility and specifies the current data file to be displayed in record access mode. A DISPLAY options screen, appropriate to this access mode and to the file structure, appears on your workstation screen. Press PF1 to display the first screen in the file. (For information about other DISPLAY options, see Chapter 11.)

When you terminate DISPLAY processing, you are returned to the DATENTRY central menu.

3.4.5 Modifying the Data Entry Screen Display (PF7)

The organization and appearance of DATENTRY's data entry screens are determined in most respects by the control file. (See "The Format of the Data Entry Screen" in Section 3.4.1 for details.) To make radical changes, you must run the EZFORMAT utility and create a new data entry program. However, DATENTRY does permit the following minor changes to be made by means of an option available from the DATENTRY central menu (Figure 3-3):

- You can change the line spacing from double to single and vice versa. (For double-spacing to be possible, there must be fewer than 11 lines to display.)

- You can change the display attributes of any field in the following ways:

  - The name and field can be underlined.

- You can cause DATENTRY to display the field (characters, pseudoblanks, or both) in highlighted intensity or standard intensity, to make it blink, or to blank it out so that neither pseudoblanks nor any characters entered are visible on the screen.

- You can have a character field accept both uppercase and lowercase letters, instead of having all letters automatically converted to uppercase.

To make any of these changes, press PF7 from the DATENTRY central menu. The Line Spacing screen (not shown) appears. This screen has one single-character field in which you enter a 1 to select single-spacing or a 2 for double-spacing. If there are 11 lines or fewer on the data entry screen for the specified file, the default value is 2, but if the number of lines on the screen is between 12 and 21, the default value is 1 and cannot be changed. Double-spacing is an option only if there are fewer than 12 lines of fields to display. (For more information about this limitation, see "The Format of the Data Entry Screen" in Section 3.4.1.)

To confirm your specification in the Spacing field or to accept the default, press ENTER. DATENTRY then displays the Field Attribute screen (Figure 3-8) for the first field in the update sequence. (Alternatively, if you wish to return to the central menu without changing the screen spacing or field attributes, press PF16.)

```
Wang VS GETPARM v 7                          Parameter Reference Name:  FAC
                                                        Message Id:  0001
                                                         Component:  DEUFAC

                        Information Required by DATENTRY

        Please specify field display attributes for field xxxxxxxx

        Field Underlining                    ULINE    = NO

        Field Display Intensity              DISPLAY  = BRIGHT
        BRIGHT, DIM, BLINK, or BLANK

        Both upper and lower case allowed    UPLOW    = NO


(ENTER) Set the display attributes and display the next field or Select
(1) Return to spacing   (2) First field    (4) Previous field   (5) Next field
(8) Find  FIELD    =                       (16) Exit to DATENTRY Central Menu
```

Figure 3-8. DATENTRY Field Attribute Screen

The name of the field in the data record appears near the top of the screen. (Although the name is highlighted, this is not a modifiable field.) At your option, enter information in the fields below it as follows:

ULINE -- Enter YES if you want the field and its name underlined on the screen. The default value is NO. This option is compatible with BRIGHT, DIM, or BLINK in the DISPLAY field, but not with BLANK. When DISPLAY is set to BLANK, DATENTRY accepts YES in this field, but does not display name, field, or underlining on the screen.

DISPLAY -- Attributes specified in this field affect the field (characters, pseudoblanks, or both), but not the name that precedes it on the screen. The four acceptable values are BRIGHT, DIM, BLINK, and BLANK. The default value, BRIGHT, represents the usual highlighted field. DIM represents normal video intensity without highlighting. BLINK causes the field (which is also highlighted) to blink on and off. (If only one field blinks, the cursor is placed on it when the screen appears. If more than one field is blinking, the cursor appears in the usual position on the first field.) When the value is BLANK, neither the empty field nor the characters entered in it appear on the screen, except as a space between the name of this field and the name of the next. This mode is commonly used for the entry of passwords.

UPLOW -- Enter YES if you want the field to accept both uppercase and lowercase letters. The default value is NO, which causes lowercase letters to be converted to uppercase.

Enter new values for any attributes you wish to change and press ENTER. DATENTRY accepts the changes (or the unchanged default values) and displays a Field Attribute screen for the next field in the update sequence. You also have these options for navigating among fields:

PF Key	Option and Description
ENTER	**Modify attributes** -- Modify the field attributes on a screen and display those for the next field.
1	**Return to spacing** -- Redisplay the Line Spacing screen.
2	**First field** -- Display the Field Attribute screen for the first field in the update sequence.
4	**Previous field** -- Display the Field Attribute screen for the field that precedes this one in the update sequence.
5	**Next field** -- Display the Field Attribute screen for the field that follows this one in the update sequence.
8	**Find FIELD** -- To move directly to a specific field, enter its name in the blank field (FIELD) provided and press PF8. This option saves time when the data entry screen contains a great many fields.

To return to the DATENTRY central menu when you finish changing field attributes or decide not to change any, press PF16.

3.4.6 Saving the Screen Contents for EZFORMAT (PF8)

If you would like to rearrange the data entry screen more thoroughly than is possible through DATENTRY, the EZFORMAT utility (described in Chapter 4 of this manual) provides a way to create a new data entry program for which you can design the screen.

When you run EZFORMAT, you can enter all the screen information manually, but DATENTRY provides a convenient shortcut. By pressing PF8 from the DATENTRY central menu, you can save the contents of the DATENTRY screen in a form that EZFORMAT can use to reconstruct them. Then you can use EZFORMAT's editing functions to rearrange the screen contents as you choose.

To save the DATENTRY screen for the currently selected data file, press PF8. The Screen and Field Name Files screen (Figure 3-9) appears.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  GENSRC
                                                    Message Id:  0001
                                                     Component:  DATENT

                        Information Required by DATENTRY

Supply the EZFORMAT save screen and fields file to be created


Screen file:
SAVEFILE = ▪▪▪▪▪▪▪▪ SAVELIB  = USRSAVE▪ SAVEVOL  = VOL111

Fields file:
FIELDFIL = ▪▪▪▪▪▪▪▪ FIELDLIB = USRFLDS▪ FIELDVOL = VOL111


(ENTER) Create the EZFORMAT files OR
                                        (16) Return to DATENTRY Central Menu
```

Figure 3-9. DATENTRY Screen and Field Name Files Screen

DATENTRY creates two files through this option: a *screen file* that describes the screen, and a *field name file* that lists all the fields in the data file and identifies them with the control file where they are described. (EZFORMAT also creates both kinds of file; for more information, see Chapter 4.)

This screen requests file information for both the screen file and the field name file. Enter specifications in the fields as follows:

SAVEFILE -- Enter a valid file name for the screen file.

SAVELIB -- DATENTRY creates a default value for this field by concatenating the characters of your user ID with the string *SAVE*, e.g., USRSAVE.

SAVEVOL -- Specify the volume where you want the screen file to reside. If you have specified the INVOL field of your usage constants (either through the Set Usage Constants function [PF2] of the Command Processor or through a procedure), the value from that field appears in this field as a default.

FIELDFIL -- Enter a valid file name for the field name file.

FIELDLIB -- DATENTRY creates a default value for this field by concatenating your user ID with the string FLDS, e.g., USRFLDS.

FIELDVOL -- Specify the volume where you want the field name file to reside. If the INVOL field of your usage constants is specified, the value from that field appears in this field as a default.

You can override any default by entering a new value in its place.

When you have specified both the screen file and the field name file, press ENTER. DATENTRY created both files in the specified locations and returns you to the central menu with no further message.

If you decide not to save the DATENTRY screen contents for EZFORMAT, press PF16. DATENTRY returns you to the central menu without creating any files.

3.4.7 Running the INQUIRY Utility (PF9)

The INQUIRY utility enables you to retrieve information or to extract records from your file by using an English-like syntax to formulate queries. INQUIRY is documented in Chapter 5 of this manual. You can move directly from DATENTRY to INQUIRY to interrogate the currently specified data file by pressing PF9 from the DATENTRY central menu.

When you run INQUIRY from within DATENTRY in this manner, the data file specified for DATENTRY is automatically specified for INQUIRY. You can change this specification within INQUIRY, but the change has no effect on DATENTRY. When you complete INQUIRY processing and return to the DATENTRY central menu, the data file that was specified at the time you began to run INQUIRY is still specified.

CHAPTER 4
EZFORMAT

4.1 INTRODUCTION

The EZFORMAT utility makes it easy to develop program screens for use in applications. After you design and create the screen in an interactive process, you can use the new screen in one of three ways:

1. You can generate a program menu with PF keys assigned to specific programs. From the menu, a user can run any program that resides in the same library as the menu object code or in @SYSTEM@ on the system volume. (This function is described in greater detail in Section 4.3.)

2. You can design a screen containing modifiable alphanumeric and numeric fields and have EZFORMAT generate source code in any of four programming languages that describes the screen. You can then import this source code into your program by using the external copy feature of the VS Editor. The screen becomes part of the program without your having to write code to describe it.

   EZFORMAT generates source code in the following languages:

   - BASIC
   - COBOL
   - RPG II
   - VS Assembly Language

3. A data entry screen can become part of a new data entry program uniquely created to interact with a particular data file. You can create this data entry program without having to write a line of source code. (An existing control file that describes the data file is necessary for this function. Chapter 2 describes control files and the use of the CONTROL utility to create them.)

4.1.1 Overview

EZFORMAT processing involves three basic steps:

1. Specify the EZFORMAT function (i.e., generate a menu, generate source code, or generate a data entry program).

2. Create the screen image. (This process is identical for the source code and data entry functions. A similar process is used for the menu function.)

3. Define the output files. These depend partly on the function.

   - For any function, there is a source code file.

   - For the menu and data entry functions there is an executable object file.

   - For any function, you can save any screen you design in a file accessible to EZFORMAT so that it can be recalled for modification or for use as a template.

   - For the source code generation and data entry functions, depending on the programming language you select, there may also be a file that saves information about the modifiable fields you place on the screen.

4.1.2 Running EZFORMAT

You can run the EZFORMAT utility by pressing PF1 (Run Program or Procedure) from the Command Processor menu or by pressing PF11 from the CONTROL Utility menu (see Section 2.7).

The following sections describe EZFORMAT processing:

Section	Process
4.2	Specifying the EZFORMAT Function
4.3	Initiating the Menu Function
4.3.1	Assigning PF Keys to Menu Functions
4.4	Creating the Screen Image
4.4.1	Formatting Options
4.4.2	Screen Manipulation Options
4.4.3	The Image Design Screen
4.5	Specifying EZFORMAT Output

4.2 SPECIFYING THE EZFORMAT FUNCTION

When EZFORMAT processing begins, the Function Specification screen (Figure 4-1) appears.

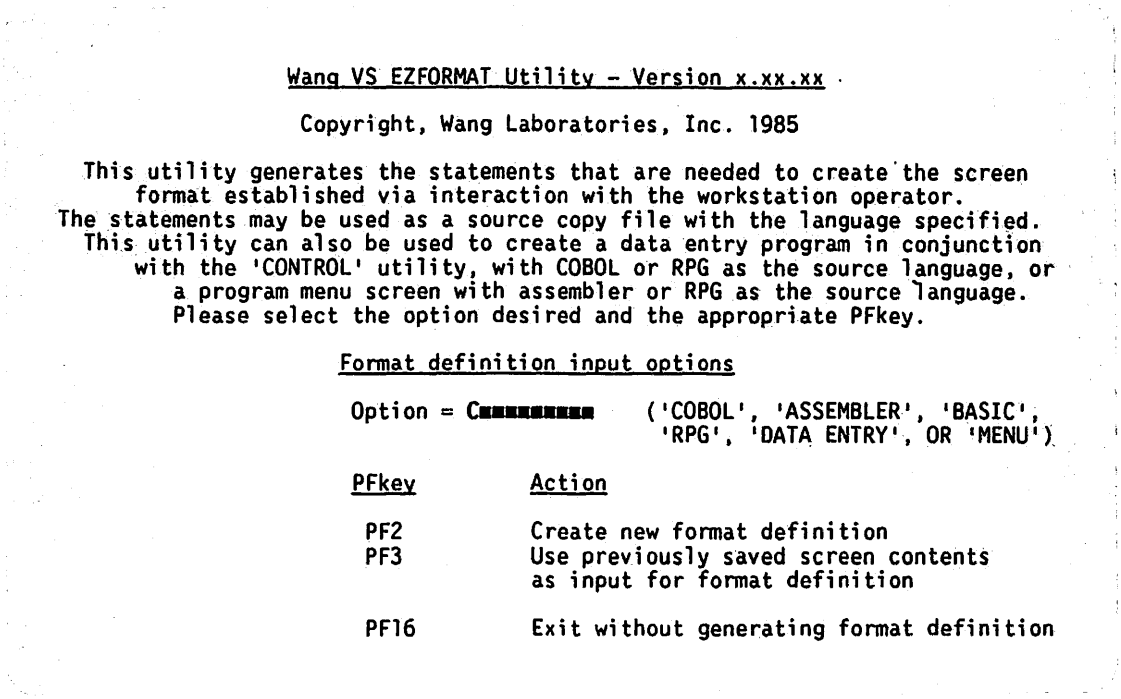


Figure 4-1. EZFORMAT Function Specification Screen

Specify the EZFORMAT function you want in the Option field as follows:

- To generate an executable *program menu*, specify MENU.
- To generate a *source code description of a screen image*, specify the source language by entering COBOL (the default), ASSEMBLER, BASIC, or RPG.
- To generate an executable *data entry program* in conjunction with the CONTROL utility, specify DATA ENTRY.

(The first letter is sufficient to specify any of these values, e.g., *D* for DATA ENTRY, *C* for COBOL.)

The next step, which is nearly the same for all three functions, is to create the screen image. Before you begin, however, you must indicate whether you intend to start from scratch or to edit a previously created screen image.

- To create a completely new screen image, without using any existing screen as input, press *PF2*.

- To recall an existing screen image so that you can use EZFORMAT to modify it, press PF3. The screen image, to be accessible to EZFORMAT, must have been saved in a specially formatted file. Such files can be created by the DATENTRY utility (Section 3.3) or by EZFORMAT itself.

  When you press PF3, EZFORMAT displays a screen from which you are prompted to supply the name, library, and volume of the file that contains the screen image you want to use. The Library field contains a default constructed by concatenating your user ID with the string *SAVE*, e.g., USRSAVE. If the INVOL field among your user constants has been set (by the pressing of PF2 from the Command Processor menu or by a procedure), the value from this field appears in the Volume field as a default.

If you decide to forego all EZFORMAT functions and to exit from the program without further processing, press *PF16*.

4.3 INITIATING THE MENU FUNCTION

Note: *If you are generating either a data entry program or a source code screen image, you begin creating the screen image as soon as you specify your choice from the EZFORMAT Function Specification screen (Figure 4-1). Section 4.4 describes this process in detail. If you are generating a program menu, however, there is a preliminary step that must be completed first (see Section 4.3.1).*

When you specify the menu function from the Function Specification screen, you can use EZFORMAT to create a menu through which you can run any other program that resides in the same library, or in the system library @SYSTEM@. This menu is itself an executable program, created by EZFORMAT to your specifications. You can assign the ENTER key plus any or all of the 32 PF keys on the VS keyboard to specific programs. Alternatively, you can assign one of these keys to a "user program" option which causes the VS Run Program or Procedure screen to appear when the key is pressed. In this way the user can specify and run any program or procedure from the menu. Other functions you can assign to specific keys are terminating the menu program and logging the user off the system. You can also disable the HELP key while the menu program is running.

Such menus can save time, since they permit many programs to be invoked with one keystroke apiece. They are also helpful as a way for inexperienced users to run programs on the VS without having to become familiar with the VS Command Processor. When the menu program is invoked automatically through a logon procedure, such a user need never confront the Command Processor directly.

It is up to you to design the menu screen, using the EZFORMAT process described in Section 4.4. Before you do so, however, you must inform EZFORMAT of the programs and other functions you intend to make available from the menu and the specific keys these programs and functions are assigned to.

4.3.1 Assigning PF Keys to Menu Functions

If you specify the menu option from the Function Specification screen, the Menu Generator screen appears. Figure 4-2 shows a sample of this screen.

```
                              Menu generator screen

Please associate the names of the programs which will be executed from the
     menu to be defined, with the PFkey list displayed on this screen.
Special functions:  Enter if desired beside PFkey 'EXIT' to exit from menu,
  'LOGOFF' TO logoff, or 'USERPROG' to run program not listed on the menu.
 On the blank format definition screen, define the menu exactly as you wish
     it to be displayed (do not delimit the text with double quotes).

PFKEY   Program name        PFkey   Program name        PFkey   Program name
ENTER     EXIT■■■■          PF1       DISPLAY■          PF2       SORT■■■■
PF3       COPY■■■■          PF4       LISTVTOC          PF5       ■■■■■■■■
PF6       ■■■■■■■■          PF7       ■■■■■■■■          PF8       ■■■■■■■■
PF9       ■■■■■■■■          PF10      ■■■■■■■■          PF11      ■■■■■■■■
PF12      ■■■■■■■■          PF13      ■■■■■■■■          PF14      ■■■■■■■■
PF15      USERPROG          PF16      LOGOFF■■          PF17      ■■■■■■■■
PF18      ■■■■■■■■          PF19      ■■■■■■■■          PF20      ■■■■■■■■
PF21      ■■■■■■■■          PF22      ■■■■■■■■          PF23      ■■■■■■■■
PF24      ■■■■■■■■          PF25      ■■■■■■■■          PF26      ■■■■■■■■
PF27      ■■■■■■■■          PF28      ■■■■■■■■          PF29      ■■■■■■■■
PF30      ■■■■■■■■          PF31      ■■■■■■■■          PF32      ■■■■■■■■
          Disable help key during menu execution  NO   (YES or NO)
                      ** Press ENTER to continue **
```

Figure 4-2. A Sample EZFORMAT Menu Generator Screen

This screen displays a list of the 32 PF keys, plus ENTER, with a blank field after the name of each key. To assign a program or function, enter its name in the field after the name of the key to which it is assigned. In Figure 4-2, four utilities are assigned to PF keys 1 through 4. PF15 summons the Run screen, and PF16 logs the user off the system. The ENTER key terminates the menu program.

You can assign any program in either the system library (@SYSTEM@) or the library where the menu program resides to any key listed on the screen. To assign the other available functions, enter the following values in the appropriate fields:

USERPROG -- Causes the VS Run Program or Procedure screen to appear. The user can run any program accessible on the system and will be returned to the menu when processing is terminated.

EXIT -- Terminates the menu program and returns the user to the Command Processor, or to the program that linked to the menu.

LOGOFF -- Terminates all processing and logs the user off the system.

If you want the HELP key disabled while the menu program is running, enter YES in the field at the bottom of the screen. The default is NO.

Note: *If you assign neither the EXIT nor the LOGOFF function, do not disable the HELP key, as this will leave the user no way to exit from the menu. See Section 4.6.2.*

When all the menu programs and functions are assigned, press ENTER. The menu screen that communicates these choices to the user must still be created. This process begins with the appearance of the Screen Manipulation Options menu (Figure 4-4), described in Section 4.4.2.

4.4 CREATING THE SCREEN IMAGE

EZFORMAT provides three screens to assist you in the process of creating the screen image:

1. The Screen Format Options screen (Section 4.4.1)

2. The Screen Manipulation Options menu (Section 4.4.2)

3. The Image Design screen (Section 4.4.3)

The first two screens provide information. You do the actual work of designing and editing the screen image on the Image Design screen. The information screens can be recalled readily during this process.

Since the information provided by the Screen Format Options screen is unnecessary for the creation of a menu screen, this screen does not appear when the menu function has been specified. Instead, the Screen Manipulation Options menu appears first.

4.4.1 Formatting Options

Note: *This section does not apply if you have specified the EZFORMAT menu function and are about to create a menu screen image. Move on to Section 4.4.2 if you are following that process.*

When you press ENTER from the Function Specification screen after specifying either the data entry or the code generation function, the EZFORMAT Screen Format Options screen (Figure 4-3) appears.

```
                        Screen Format Options

These formats are used to arrange and define the fields on the screen.

  Field Type                 Valid Formatting Options
  Text                       Must have double quotes ["].
  Numeric Modifiable         Must be [0 - 9], [+], [-], [.].
  Alphanumeric Modifiable    Must begin with an alphabetic character.

The following characters are included in the length of a modifiable field.
   *   This is a required alphanumeric field (cannot be blank).
   +   This is a zoned numeric field; result may be [+] or [-].
   -   This is a zoned numeric field; result may be space or [-].
   .   This is a decimal point (position) in a numeric field.

Source and object data-names are generated for all modifiable fields.
Source and object data-names can be defined after screen format definition.
Value clauses are generated for modifiable fields where the first
 character is not one of the following: [X]  [*]  [9]  [+]  [-]  [.]

A printed copy of these instructions plus the menu of options can be
 obtained by pressing PF13 from the next menu.

                    **  Press ENTER to continue  **
```

Figure 4-3. EZFORMAT Screen Format Options Screen

The Screen Format Options screen describes the options available for formatting the various fields in the screen image you are about to create. There are three kinds:

1. Text fields (nonmodifiable)

2. Numeric modifiable fields

3. Alphanumeric modifiable fields

Text Fields

Everything visible on the screen that is not a modifiable field (i.e., a field in which the user can enter information) is considered a text field. Text fields are by definition nonmodifiable. On a data entry screen, for example, the following information is contained in text fields: the labels that identify the modifiable fields, all other text that provides information or describes options to the user, and characters used to form graphic delimiters, such as dotted lines and rows of asterisks.

For the data entry and code generation functions, EZFORMAT recognizes as a text field any sequence of characters enclosed between double quotation marks on the Image Design screen. The quotation marks are only a signal to EZFORMAT, and do not appear on the generated screen. All text characters will occupy exactly the same positions on the generated screen that they occupy on the Image Design screen, regardless of the position of the quotation marks that surround them (on the Image Design screen only). For example, if the Image Design screen contains the text field "LABEL:", placed so that the first *L* is in character position 12, the first *L* will also occupy character position 12 in the generated screen (where LABEL: will appear without surrounding quotation marks).

Numeric Modifiable Fields

If your screen is to contain modifiable fields in which only numeric data is acceptable as input, they should be formatted as numeric modifiable fields. These fields accept numbers (0 - 9), plus and minus signs (+, -), and decimal points (.) as input; other characters are rejected.

EZFORMAT recognizes as a numeric modifiable field any sequence of characters on the Image Design screen that begins with a number or a plus or minus sign, and is not enclosed in double quotation marks. The end of the field is delimited by a space, the end of a line, or the double quotation mark that indicates the beginning of a text field.

The way to specify a "generic" numeric field -- one that will be filled, in the generated screen, with pseudoblanks instead of a default value -- is to enter a 9 for each character in the field. For example, 999999 on the Image Design screen specifies a numeric field capable of receiving a 6-digit number (or a 5-digit number that includes a decimal point). A field specified in this way, however, can receive only positive values.

You can provide for positive and negative input in a field by specifying either a plus or a minus sign as the first character. Either type of field accepts numeric input preceded by a plus or a minus sign, and stores input preceded by a minus sign as negative values. The only difference is in the display. A field specified with a plus sign (e.g., +9999) displays values with a leading plus or minus sign. A field specified with a minus sign (e.g., -9999) displays negative values with a leading minus sign, but positive values with a leading space.

When a field is specified with a leading plus or minus sign, it is necessary to remember that the sign accounts for one character of the field. That is, a 7-character field of either type (+999999 or -999999) could not accept numbers of more than six digits, although it would show seven pseudoblanks on the generated screen.

You can also specify a numeric modifiable field that is not "generic," but is initialized to a default value that appears in the field on the generated screen. To do this, make any digit other than 9 the first digit when you specify the field on the Image Design screen (e.g., 123456, +123456, -123456). EZFORMAT generates the program statements necessary to initialize the field to the specified value. (The default can be overridden.) The only way to initialize a numeric field to a value beginning with 9 is to precede it with a 0 on the Image Design screen, if the field length permits -- e.g., 099545.

Note: *When you press PF14 to display a screen image in progress as it will appear on the generated screen (see Section 4.4), a leading plus or minus sign is always displayed as a pseudoblank.*

Table 4-1 summarizes the characteristics of numeric modifiable fields in a series of examples.

Table 4-1. Examples of Numeric Modifiable Fields

Field as Specified	Field as It First Appears	Value Entered	Value Stored	Value Displayed
99999999	■■■■■■■■	177.92	+177.92	00177.92
		+177.92	+177.92	00177.92
		-177.92	+177.92	00177.92
+9999999	■■■■■■■■	327867	+327867	+0327867
		+327867	+327867	+0327867
		-327867	-327867	-0327867
-9999999	■■■■■■■■	58.319	+58.319	058.319
		+58.319	+58.319	058.319
		-58.319	-58.319	-058.319
00010000	00010000	3010.93	+3010.93	03010.93
		+3010.93	+3010.93	03010.93
		-3010.93	+3010.93	03010.93
+8724193	08724193	9137184	+9137184	+9137184
		+9137184	+9137184	+9137184
		-9137184	-9137184	-9137184
-0099545	0099545	87313	+87313	0087313
		+87313	+87313	0087313
		-87313	-87313	-0087313

Alphanumeric Modifiable Fields

If your screen is to contain modifiable fields in which alphanumeric data is acceptable as input, the fields should be formatted as alphanumeric modifiable fields. These fields accept all displayable characters as input.

EZFORMAT recognizes as an alphanumeric modifiable field any sequence of characters on the Image Design screen that begins with a character other than a number, a plus sign, or a minus sign, and is not enclosed in double quotation marks. The end of the field is delimited by a space, the end of a line, or the double quotation mark that indicates the beginning of a text field.

The way to specify a "generic" alphanumeric field -- one that will be filled, in the generated screen, with pseudoblanks instead of a default value, is to enter an X for each character in the field. For example, XXXXXX on the Image Design screen specifies an alphanumeric field capable of receiving a 6-character value.

If the screen image you are designing is to be converted into BASIC or COBOL source code, you can mark a required alphanumeric field by beginning the field specification with an asterisk. Such a field must be specified before the screen can be processed; the user is not free to leave it blank. A field specified with an X following an asterisk (e.g., *XXXXXXX) appears as a series of pseudoblanks on the generated screen. (Although this method applies most obviously to generating a source code screen image, you can also apply it when you generate a data entry program. You must specify COBOL as the source language -- see Section 4.12.1.)

To specify an alphanumeric field with a default value, use any character other than X as the initial character (or, for a required field, the character that follows the asterisk) when you specify the field on the Image Design screen (e.g., DISPLAY, *LOWELL). EZFORMAT generates the program statements necessary for initializing the field to the specified value. (This default can be overridden.)

Note: *When you press PF14 to display a screen image in progress as it will appear on the generated screen (see the next section), a leading asterisk is always displayed as a pseudoblank.*

4.4.2 Screen Manipulation Options

When you press ENTER from the Menu Generator screen (Figure 4-2) or the Screen Format Options screen (Figure 4-3), the Screen Manipulation Options menu (Figure 4-4) appears.

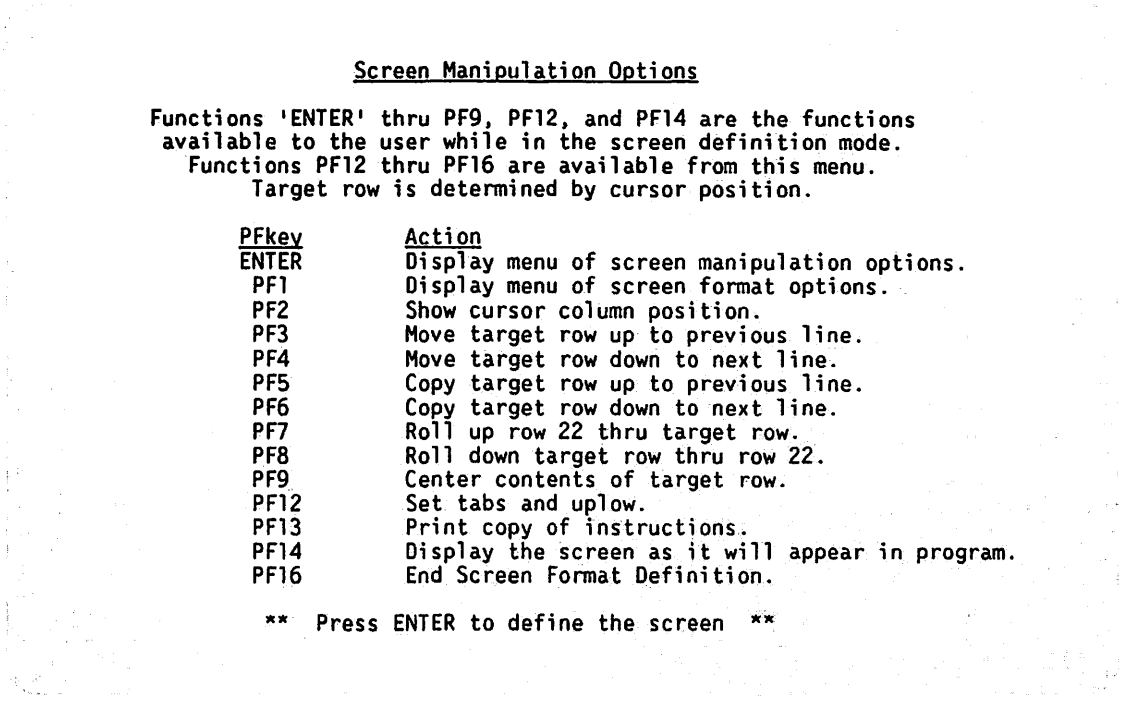


Figure 4-4. EZFORMAT Screen Manipulation Options Menu (Data Entry Function)

This menu is unusual in that most of the options it describes are available not directly from the menu, but from the Image Design screen (Figure 4-5) for which it serves as a guide. These options save you inconvenience in arranging the screen contents as you work out your design. Since your work space on the Image Design screen fills nearly the whole workstation screen, the menu cannot be displayed at the same time, but you can use the ENTER key to switch back and forth at will between the menu and the Image Design screen.

Although most options listed in the menu are the same for all three EZFORMAT functions, there are a few differences. The lowest line you can use for the screen image is Row 22 for the data entry function, Row 23 for the menu function, and Row 24 for the code generation function. For another example, PF14 has one meaning for the data entry and source code generation functions and another meaning for the menu generation function. The Screen Manipulation Options menu has slightly variant versions that reflect these differences.

All differences are discussed in the two following lists, which provide a general description of the screen manipulation options.

Functions Available From the Image Design Screen

The functions available from the Image Design Screen are listed below.

PF Key	Function	Description
ENTER	Display menu of screen manipulation options	Press ENTER to alternate between the Screen Manipulation Options menu and the Image Design screen.
1	Display menu of screen format options	Press PF1 to display the Screen Format Options screen (Figure 4-3). This screen summarizes the information for formatting different fields presented in Section 4.4.1. **Note:** *Since a menu screen contains text fields only, this function is not needed in designing a menu screen image, and does not appear on the Screen Manipulation Options menu when the program menu option has been specified.*
2	Show cursor column position	Press PF2 to display the current column position of the cursor. This information appears at the upper right of the Image Design screen, *temporarily* overwriting any field defined in that position. This display does not destroy field data.
3	Move target row up to previous line	Press PF3 to move the entire contents of the target row (i.e., the row the cursor is on) to the same column positions in the row above. The entire contents of the row above are overwritten, and the target row is filled with pseudoblanks (or simple blanks, for the Menu option). Nothing happens if you try to move the top row. In the following example, Row 7 is the target row: **Row # Before After** 5 AAAAA AAAAA 6 BBBBB CCCCC 7 CCCCC ■■■■■ 8 DDDDD DDDDD

PF Key	Function	Description
4	Move target row down to next line	Press PF4 to move the entire contents of the target row (i.e., the row the cursor is on) to the same column positions in the row below. The entire contents of the row below are overwritten, and the target row is filled with pseudoblanks (or simple blanks, for the Menu option). Nothing happens if you try to move the bottom row.
5	Copy target row up to previous line	Press PF5 to copy the entire contents of the target row (i.e., the row the cursor is on) to the same column positions in the row above. The entire contents of the row above are overwritten, and the target row remains unchanged, so that both rows are identical when the action is completed. Nothing happens if you try to copy the top row upwards. In the following example, Row 7 is the target row:

Row #	Before	After
5	AAAAA	AAAAA
6	BBBBB	CCCCC
7	CCCCC	CCCCC

6	Copy target row down to next line	Press PF6 to copy the entire contents of the target row (i.e., the row the cursor is on) to the same column positions in the row below. The entire contents of the row below are overwritten, and the target row remains unchanged, so that both rows are identical when the action is completed. Nothing happens if you try to copy the bottom row downwards.

PF Key	Function	Description
7	Roll up Row 22 (Row 23, Row 24) thru target row	Press PF7 to move the contents of each row, from the target row to the bottom row, up to the row above. The contents of each row including the target row are overwritten by the contents of the row below it as the rows "roll" upwards. The bottom row (22, 23, or 24 depending on the EZFORMAT function) is filled with pseudoblanks. In the following example, Row 18 is the target row and Row 22 is the bottom row:

Row #	Before	After
16	RRRRR	RRRRR
17	SSSSS	SSSSS
18	TTTTT	UUUUU
19	UUUUU	VVVVV
20	VVVVV	WWWWW
21	WWWWW	XXXXX
22	XXXXX	■■■■■

8	Roll down target row thru Row 22 (Row 23, Row 24)	Press PF7 to move the contents of each row from the target row to the bottom row down to the row below. The contents of each row are overwritten by the contents of the row above it as the rows "roll" downwards. The target row is filled with pseudoblanks, and the contents of the bottom row (22, 23, or 24 depending on the EZFORMAT function) are lost.
9	Center contents of target row	Press PF9 to center the contents of the target row horizontally.

Functions Available From the Screen Manipulation Options Menu

The functions available from the Screen Manipulation Options Menu are listed below.

Note: *If you are working on the Image Design screen, you must press ENTER to return to the Screen Manipulation Options menu before you press a PF key to select one of these functions.*

PF Key	Function	Description
12	Set tabs and uplow	Press PF12 to set tab positions for the Image Design screen and to determine whether modifiable fields on the screen you are designing will accept lowercase input. When you press PF12, EZFORMAT displays the Set Tabs and UPLOW screen (Figure 4-5). The numbers represent column positions on the screen. Eight default tabs are set in the positions shown. To delete any of these tabs, delete the *1* or type a blank over it. To set a new tab, type any nonblank character below the position where you want the tab. You can set a maximum of ten tabs.

PF Key	Function	Description
(12)	Set tabs and uplow (continued)	The second item on the Set Tabs and UPLOW screen is the UPLOW field. The default value is UPLOW. If the field contains this value, modifiable alphanumeric fields on the screen will accept both uppercase and lowercase input. If you want these fields to accept only uppercase input, change the value to UPPER. The UPLOW field affects neither the text fields on the screen nor the default values in the modifiable fields. You can set it to UPPER and still supply lowercase default values for these fields. However, the user can enter only uppercase letters in the fields; lowercase letters are converted to uppercase as they are entered. ***Note:*** *When you are creating fields on the Image Design screen,* all *modifiable fields must be entered in uppercase letters, whether or not they are designed to refuse lowercase input.*
13	Print copy of Instructions	Press PF13 to create a print file that reproduces the information given on the Screen Format Options screen (Figure 4-3) and the Screen Manipulation Options menu (Figure 4-4).
14	Display the screen as it will appear in program	*For the data entry and source code generation functions only,* press PF14 to preview the screen as your program will display it. The double quotation marks around text fields are suppressed, and all pseudoblanks except those that represent modifiable fields disappear. Everything is shown in the position it will occupy when the finished screen is displayed in your program. (If you press ENTER from this display, you are returned to the Screen Manipulation Options screen.)

PF Key	Function	Description
14	Redisplay the list of PF keys selected	*For the menu generation function only,* press PF14 to review the assignments you made from the EZFORMAT Menu Generator screen (Figure 4-2). If you press ENTER from this display, you are returned to the Screen Manipulation Options screen.
16	End Screen Format Definition	Press PF16 to halt the process of creating the screen image.

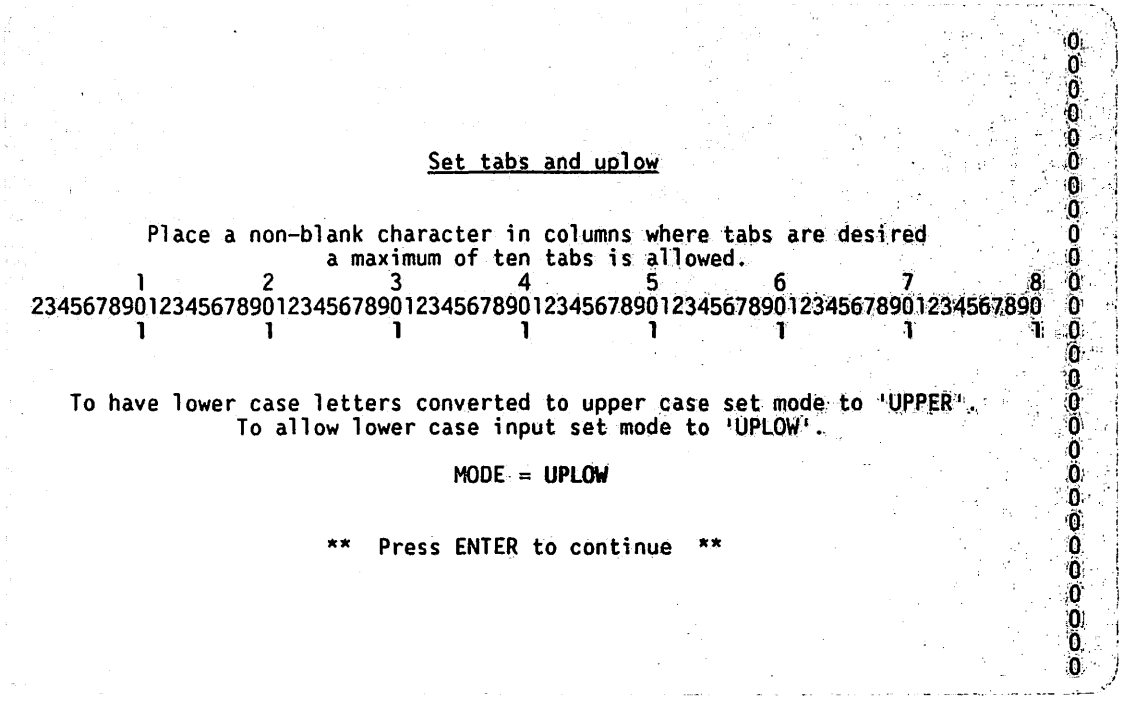


Figure 4-5. EZFORMAT Set Tabs and UPLOW Screen

4.4.3 The Image Design Screen

When you press ENTER from the Screen Manipulation Options screen (Figure 4-4), the Image Design screen (Figure 4-6) appears.

This screen, when it first appears, is filled with pseudoblanks. (Each row contains 80 pseudoblanks. There are 22 rows for the data entry function, 23 rows for the menu function, and 24 rows for the source code generation option.) You design the screen image by entering information in place of the pseudoblanks.

For the menu generation function, replace the pseudoblanks with characters in the positions where you want them to appear on the menu.

For the data entry and source code generation functions, replace the pseudoblanks with characters in accordance with the format conventions described in Section 4.4.1. Pressing PF1 from the Image Design screen recalls the Screen Format Options screen for easy reference.

For all three functions, you can use the screen manipulation options described in Section 4.4.2 to arrange and rearrange the information you have entered on the Image Design screen. Pressing ENTER from this screen recalls the Screen Manipulation Options menu for easy reference.

Figure 4-6 shows a sample Image Design screen containing the information needed to generate a data entry program for payroll records. Figure 4-7 shows the screen as it appears when the generated program is run. (Figure 3-4 shows a DATENTRY data entry screen for the same file.)

```
"ACE CORP. WEEKLY PAYROLL RECORD"
"Employee Information"
"Last Name:"LAST
"First Name:"FIRST
"Employee Number:"EMPNUM
"Supervisor:"SUPRVISR
"Job Classification:"JOBCLASS
"Pay Grade:"PAYGRADE
"Pay Rate:"99999
"Weekly Pay Record"
"Week Ending:"WKEND
"Regular Hours:"99999"Regular Pay:"9999999
"Overtime Hours:"99999"Overtime Pay:"9999999
"Rows 23 and 24 are reserved for data entry messages.                    "
"                                                                        "
```

Figure 4-6. A Sample EZFORMAT Image Design Screen

```
                    "ACE CORP. WEEKLY PAYROLL RECORD"

Employee Information

      Last Name:            ■■■■■■■■■■■■■■■■
      First Name:           ■■■■■■■■■■■■■
      Employee Number:      ■■■■■■■■■

      Supervisor:           ■■■■■■■■■■■■■■■■
      Job Classification:   ■■■■
      Pay Grade:            ■■
      Pay Rate:             ■■■■■

Weekly Pay Record

      Week Ending:          ■■■■
      Regular Hours:        ■■■■■       Regular Pay:     ■■■■■■■
      Overtime Hours:       ■■■■■       Overtime Pay:    ■■■■■■■
```

Figure 4-7. A Sample Data Entry Screen Generated by EZFORMAT

4.5 SPECIFYING EZFORMAT OUTPUT

Once you have chosen an EZFORMAT function and designed a screen image, the final steps in EZFORMAT processing involve specifying the nature and location of the various output files the program creates. Since this process varies according to the EZFORMAT function, it is described separately for each function in the remaining sections of this chapter:

Section 4.6 -- Specifying Output for the Menu Function

Section 4.7 -- Initial Output Specifications for the Source Code Generation Function

Section 4.8 -- Specifying Assembly Language Output Files

Section 4.9 -- Specifying BASIC Output Files

Section 4.10 -- Specifying COBOL Output Files

Section 4.11 -- Specifying RPG II Output Files

Section 4.12 -- Specifying Output for the Data Entry Function

4.6 SPECIFYING OUTPUT FOR THE MENU FUNCTION

The output specification process for the EZFORMAT menu function consists of the following steps:

1. Choosing whether to save the screen image file, the source code file, or both (Section 4.6.1). EZFORMAT can use the screen image file to reproduce your screen design on the Image Design screen. The source code file contains code for a menu program that incorporates the screen image and PF key assignments you have designed.

   Note: *Steps 2 through 5 are all described in Section 4.6.2.*

2. Specifying the screen image file, if you have chosen to save it.

3. Selecting the source language for the menu program. EZFORMAT can create source code for the menu in Assembly Language or RPG II.

4. Specifying the source file for the data entry program.

5. Specifying the object file for the data entry program.

4.6.1 Selecting an Output Option

As soon as you press PF16 (End screen format definition) from the Screen Manipulation Options menu (Figure 4-4), the Output Options menu (Figure 4-8) appears.

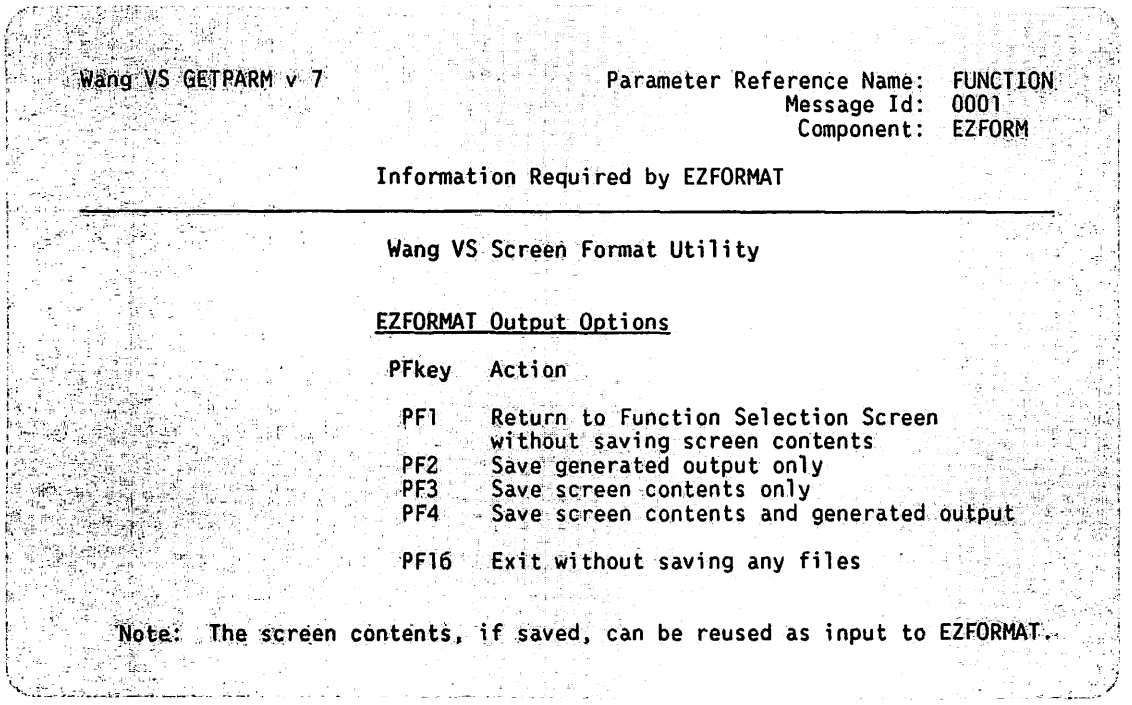


Figure 4-8. EZFORMAT Output Options Menu

Use this screen to tell EZFORMAT whether to save only the screen image file, only the source and object files for the menu program, or both. The available options are as follows:

PF Key **Option and Description**

1 **Return to the Function Selection Screen without saving screen contents** -- This is equivalent to starting EZFORMAT processing again from the beginning. Any work you have done on the Image Design screen is lost.

2 **Save generated output only** -- This option saves only the source and object files for the menu program. The screen image file is not saved, although the screen you have designed will be described in the source code and generated by the menu program. If you later want to use EZFORMAT to modify this screen image for another menu, you will have to enter all the screen information on a blank Image Design screen.

PF Key	Option and Description
3	**Save screen contents only** -- This option saves only the screen image file. No source code or menu program is generated, although you can later run EZFORMAT, reproduce your screen design by calling up the screen image file, and go directly to the output specification process.
4	**Save screen contents and generated output** -- This option saves both the screen image file and the source and object files for the menu program.
16	**Exit without saving any files** -- This option terminates EZFORMAT processing without saving any output. Any work you have done while running the program is lost.

4.6.2 Specifying the Source Language and the Output Files

If you press either PF3 or PF4 from the Output Options menu, the Image File Specification screen (not shown) appears. This screen has three fields in which to enter the specifications for the screen image file. If, at the beginning of EZFORMAT processing, you recalled an existing screen image by pressing PF3 from the Function Specification screen (Figure 4-1), the name, library, and volume you specified at that time for the screen image file appear here as defaults.

FILE -- Supply a valid name for the screen image file, unless the field contains a default you want to accept.

LIBRARY -- If you did not specify an existing screen image file, EZFORMAT provides a default library name by concatenating your user ID with the string SAVE (e.g., USRSAVE).

VOLUME -- The default is the value of the INVOL field of your usage constants, if it has been set. Otherwise, the field is blank and you must supply the name of the volume where you want the screen image file to reside.

When you have provided specifications for the screen image file, press ENTER.

If you are saving only the screen image file (i.e., if you pressed PF3 from the Output Options menu), this action completes EZFORMAT processing. The screen image file is created, and EZFORMAT returns you to the Command Processor or to the program or procedure from which EZFORMAT was called.

If you are also generating a menu program (i.e., if you pressed PF4 from the Output Options menu), the Menu Source Language screen (not shown) appears next. Use this screen to select the programming language in which the source code for the menu program will be generated. EZFORMAT offers two options:

PF Key	Option and Description
2	**Assembler** -- Press PF2 to have the source code generated in VS Assembly Language.
3	**RPG** -- Press PF3 to have the source code generated in RPG II.

Note: *If the source code is created in RPG II, the CEXIT and LINK VSSUBS must be present in the VSSUBS library on the system volume so that EZFORMAT can link them to the menu program.*

As soon as you have made this selection, the Menu Source File Specification screen (not shown) appears. This screen contains three fields in which to enter the specifications for the file that will contain the Assembly Language or RPG II source code for the menu program.

FILE -- If you began by specifying an existing screen image file, its name is used as the default. Otherwise this field is blank.

LIBRARY -- EZFORMAT concatenates your user ID with the string *COPY* (e.g., USRCOPY) and displays this value as the default.

VOLUME -- EZFORMAT uses as a default the value in the OUTVOL field of your usage constants, if this has been set.

Enter file information in these fields as necessary and press ENTER to continue.

The Menu Object File Specification screen (not shown) appears. This screen has three fields in which to enter specifications for the file that will contain the executable object code for the menu program.

FILE -- The default is the name you supplied for the source file.

LIBRARY -- EZFORMAT uses as a default the value in the OUTLIB field of your usage constants, if this has been set.

VOLUME -- EZFORMAT uses as a default the value in the OUTVOL field of your usage constants, if this has been set.

Enter file information in these fields as necessary. When your specifications are complete, press ENTER. EZFORMAT creates the source and object files for the data entry program and terminates processing, returning you to the Command Processor or to the program or procedure from which the utility was called. The print file generated during compilation is automatically assigned the same name as the source file, and is placed in your print library.

4.6.3 Running the Menu Program

If your menu is to run programs that do not reside in the system library @SYSTEM@, these programs must all be in the same library as the menu object file. If necessary, move the object file and the programs into the same library before you run the menu program.

To run a menu program, press PF1 (Run program or procedure) from the Command Processor menu. Specify the name of the menu program and the library and volume where the program resides; then press ENTER.

The screen you created through EZFORMAT appears when the menu program is run. You can execute any function, program, or procedure listed in the menu (provided it resides in the same library as the menu object file or in @SYSTEM@) by pressing the assigned PF key. The menu screen reappears when a program terminates. A message in row 24 indicates whether the program completed processing or was cancelled.

The USERPROG function does not retrieve the user's default RUNLIB and RUNVOL values as the regular Command Processor Run function does. After the USERPROG function has been invoked, the values last entered in the File, Library, and Volume fields appear as defaults.

Menu processing ends when you select and press a PF key associated with either the EXIT or the LOGOFF function. If neither has been assigned, you must cancel processing by pressing the HELP key and then PF16. *Be sure not to disable the HELP key if you do not intend to assign one of these functions.*

4.7 INITIAL OUTPUT SPECIFICATIONS FOR THE SOURCE CODE GENERATION FUNCTION

When you specify the source code generation function from the Function Specification screen (Figure 4-1), your specification determines which of four programming languages the source code will be generated in. The first part of the process is the same for all four languages, and is described in Sections 4.7.1 and 4.7.2.

Since the last part of the output specification procedure varies according to the source language, however, it is described separately for each language in the following sections:

Section 4.8 -- Assembly Language
Section 4.9 -- BASIC
Section 4.10 -- COBOL
Section 4.11 -- RPG II

4.7.1 Selecting an Output Option

As soon as you press PF16 (End Screen Format Definition) from the Screen Manipulation Options menu (Figure 4-4), the Output Options menu (Figure 4-8) appears. Use this screen to tell EZFORMAT whether to save only the screen image file, only the source code file, or both. The available options are as follows:

PF Key	Option and Description
1	**Return to the Function Selection Screen without saving screen contents** -- This is equivalent to starting EZFORMAT processing again from the beginning. Any work you have done on the Image Design screen is lost.
2	**Save generated output only** -- This option saves only the source code that describes your screen (and, for the BASIC and COBOL options, a file that saves additional information about the fields on the screen). The screen image file is not saved, although the screen you have designed is fully described in the source code. If you later want to use EZFORMAT to modify this screen image for another program, you will have to enter all the screen information on a blank Image Design screen.
3	**Save screen contents only** -- This option saves only the screen image file. No source code is generated, although you can later run EZFORMAT, reproduce your screen design by calling up the screen image file, and go directly to the output specification process.

PF Key	Option and Description
4	**Save screen contents and generated output** -- This option saves both the screen image file for EZFORMAT and the source code file that describes the screen.
16	**Exit without saving any files** -- This option terminates EZFORMAT processing without saving any output. Any work you have done while running the program is lost.

4.7.2 Saving the Screen Image File

Note: *If you press PF2 (Save generated output only) from the Output Options menu, this section does not apply. See the list at the end of this section to find out where the rest of the output specification process for the source language you selected is described.*

If you press either PF3 or PF4 from the Output Options menu, the Image File Specification screen (not shown) appears. This screen has three fields in which to enter the specifications for the screen image file. If, at the beginning of EZFORMAT processing, you recalled an existing screen image by pressing PF3 from the Function Specification screen (Figure 4-1), the name, library, and volume you specified at that time for the screen image file appear here as defaults.

FILE -- Supply a valid name for the screen image file, unless the field contains a default you want to accept.

LIBRARY -- If you did not specify an existing screen image file, EZFORMAT provides a default library name by concatenating your user ID with the string *SAVE* (e.g., USRSAVE).

VOLUME -- The default is the value of the INVOL field of your usage constants, if it has been set. Otherwise, the field is blank, and you must supply the name of the volume where you want the screen image file to reside.

When you have provided specifications for the screen image file, press ENTER. EZFORMAT creates the file.

If you are saving only the screen image file (i.e., if you pressed PF3 from the Output Options menu), this action completes EZFORMAT processing. You are returned to the Command Processor or to the program or procedure from which EZFORMAT was called.

If you pressed PF4 to save both image and generated output, the process continues as described in the section that corresponds to the source language you selected:

Section 4.8 -- Assembly Language
Section 4.9 -- BASIC
Section 4.10 -- COBOL
Section 4.11 -- RPG II

4.8 SPECIFYING ASSEMBLY LANGUAGE OUTPUT FILES

If you enter A or ASSEMBLER in the Option field of the Function Specification screen (Figure 4-1), EZFORMAT creates an Assembly Language source file capable of generating the screen you have designed. You can copy this source file into an Assembly Language program using the XCOPY function of the VS Editor (see the *VS Editor Reference*).

Note: *The source file consists entirely of Defined Constant statements; you must code any corresponding WRITE or READ statements in the main Assembly Language program.*

When you finish specifying the screen image file, or when you press PF2 (Source code generation only) from the Output Options screen, the Assembly Language Source File Specification screen (not shown) appears. This screen contains three fields in which to enter the specifications for the file that will contain the Assembly Language source code.

FILE -- If you specified a screen image file, the file name you supplied appears as a default.

LIBRARY -- EZFORMAT concatenates your user ID with the string *COPY* (e.g., USRCOPY) for a default value in this field.

VOLUME -- EZFORMAT uses as a default the value in the OUTVOL field of your usage constants, if this has been set.

Enter file information in these fields as necessary and press ENTER. EZFORMAT creates the source file and ends processing. You are returned to the Command Processor or to the program or procedure from which EZFORMAT was called.

4.9 SPECIFYING BASIC OUTPUT FILES

If you enter *B* or *BASIC* in the Option field of the Function Specification screen (Figure 4-1), EZFORMAT creates a BASIC source file. (It also creates a BASIC field names file, described in the next section.) The code in the source file, consisting of DISPLAY, ACCEPT, and DIM statements, defines and dimensions fields of the appropriate data types corresponding to the fields on the screen. When you incorporate this code in a BASIC program, it can recreate the screen image you designed and read data entered into modifiable fields.

Note: *The generated source code will run when compiled, but the generated screen image is displayed only once, and data entered on it is lost. Before compiling the program, you must supply code that writes data to a record, redisplays the screen, and so on.*

4.9.1 Specifying the BASIC Field Names File

EZFORMAT generates field names for BASIC screen images, but it also provides a way for you to change these to more meaningful values, as well as to set upper and lower bounds for acceptable input. The next section describes this process, but before it begins you must specify a file in which this field information can be saved.

The existence of this file saves you labor if you decide to modify a screen image by adding or deleting modifiable fields. EZFORMAT obtains the original field information from the field names file, and you need to specify only the new fields.

The Field Names File Specification screen (not shown) appears when you finish specifying the screen image file, or when you press PF2 (Save generated output only) from the Output Options menu (Figure 4-8). This screen has the usual three fields in which to enter file information:

FILE -- If you specified a screen image file, its name appears as the default value in this field.

LIBRARY -- A default value is created by concatenating your user ID with the string *FLDS* (e.g., USRFLDS).

VOLUME -- The default volume name is taken from the INVOL field of your usage constants, if this is set.

You can override any of these defaults. Add or change the information as necessary. Then, if the field name file you have specified already exists, press ENTER. If it does not exist, however, and you are creating a new field name file, press PF2.

4.9.2 Defining BASIC Field Names and Ranges

When you finish specifying the field names file, the first Data-Name and Range Definition screen appears. Figure 4-9 shows a sample of this screen.

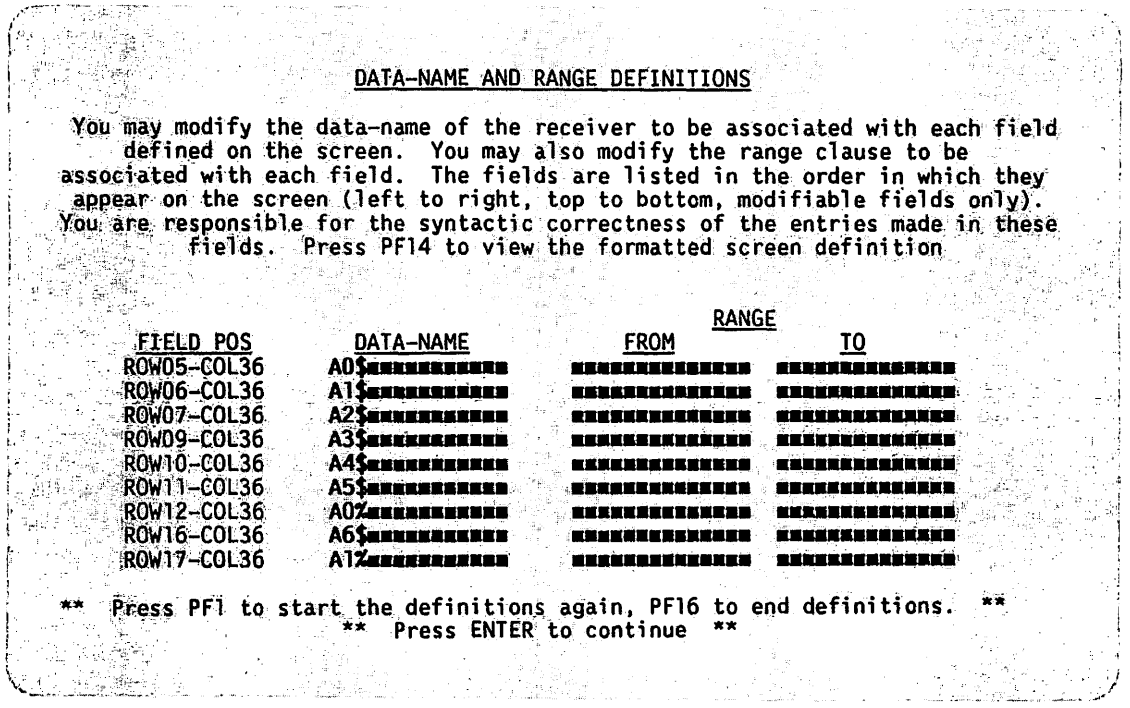


Figure 4-9. A Sample EZFORMAT BASIC Data-Name and Range Definition Screen

Each Data-Name and Range Definition screen lists nine fields. If your screen image has more than nine modifiable fields, EZFORMAT displays as many screens as necessary to list them all. You can move from one screen to the next by pressing ENTER.

For each field, the screen displays the position, a variable name generated by EZFORMAT, and two blank fields to show the range for acceptable input. You cannot modify the field position, but you can change the variable name to something more appropriate to the nature of the field, and you can specify the range. You are also free to decline these options; EZFORMAT can generate source code from the screen as it appears, using the default variable names for the fields (subject to limitations described in the comments on the fields below). Range specifications are optional.

Specify the fields on the Data-Name and Range Definitions screen, if you choose to do so, as follows:

DATA-NAME -- Specify the field name by which the generated source code should reference the modifiable field. Default names for alphanumeric fields are generated on the pattern *Xn$*, where *X* is a letter from A to Z and *n* is a digit from 0 to 9. EZFORMAT thus generates up to 260 unique default names, in a series from A0$, A1$, A3$ to Z7$, Z8$, Z9$. If you have more than 260 alphanumeric fields, you must change the data-names generated for the excess fields lest the first fields be overwritten.

Similarly, integer and fractional fields are given default names from A0% to Z9% up to a maximum of 260. Because the default numeric field name includes a %, which indicates an integer field, you must change the data names for all fractional numeric fields.

Note: *Although VS BASIC supports 64-character variable names, the names you can assign to the fields through EZFORMAT cannot exceed 14 characters.*

RANGE -- At your option, specify the lowest (*From*) and highest (*To*) acceptable input values. Default range values are displayed for any field that was defined in the screen image. You can eliminate these ranges, or make them more restrictive. To apply bounds to fields without default values, supply the appropriate values in the From and To range fields. You can define only one set of range fields for each field listed.

You have these options from any Data-Name and Range Definition screen:

PF Key	Option and Description
ENTER	**Continue** -- If there is more than one Data-Name and Range Definition screen (i.e., if there are more than nine fields in your screen design), the ENTER key moves you to the next screen, and from the last screen back to the first. If there is only one screen, this key has no effect.

PF Key	Option and Description
1	**Start the definitions again** -- This option deletes all your entries in the fields on the currently displayed screen and restores the default values.
14	**View the formatted screen definition** -- Display the screen image as you designed it. Seeing which fields are in which locations helps you identify them correctly on the Data-Name and Range Definition screen.
16	**End definitions** -- When you are satisfied with the field variable names and ranges, and have no more entries to make from any Data-Name and Range Definition screen, press PF16 to complete this part of the process and go on to the next part -- specifying the source file (see the next section).

4.9.3 Specifying the BASIC Source File

When you finish defining the field names and ranges, and press PF16 from a Data-Name and Range Definitions screen (Figure 4-9), the BASIC Source File Specification screen (not shown) appears. This screen contains three fields in which to enter the specifications for the file that will contain the BASIC source code.

FILE -- If you specified a screen image file, the file name you supplied appears as a default value.

LIBRARY -- EZFORMAT concatenates your user ID with the string **COPY** (e.g., USRCOPY) for a default value in this field.

VOLUME -- EZFORMAT uses as a default the value in the OUTVOL field of your usage constants, if this has been set.

Enter file information in these fields as necessary and press ENTER. The source file is created and EZFORMAT processing is terminated. You are returned to the Command Processor or to the program or procedure from which EZFORMAT was called.

4.10 SPECIFYING COBOL OUTPUT FILES

If you enter *C* or *COBOL* in the Option field of the Function Specification screen (Figure 4-1), EZFORMAT creates a COBOL source file. (It also creates a COBOL field names file, described in the next section.) The code in the source file contains the Data Division definition of the screen image record, including data names, data types, and initial values (if any) that correspond to your screen design. When you incorporate this code in a COBOL program, the program recreates the screen image you designed and can read any data that is entered in modifiable fields. The generated source code, however, does not include Procedure Division statements to process the screen image record; you must write these yourself.

4.10.1 Specifying the COBOL Field Names File

EZFORMAT generates field names for COBOL screen images, but it provides a means for you to change these to more meaningful values, as well as to set upper and lower bounds for acceptable input. The next section describes this process, but before it begins, you must specify a file in which the field information can be saved.

The existence of this file saves you labor if you decide to modify a screen image by adding or deleting modifiable fields. EZFORMAT obtains the original field information from the field names file, and you need to specify only the new fields.

The Field Names File Specification screen (not shown) appears when you finish specifying the screen image file, or when you press PF2 (Save generated output only) from the Output Options menu (Figure 4-8). This screen has the usual three fields in which to enter file information:

FILE -- If you specified a screen image file, its name appears as the default in this field.

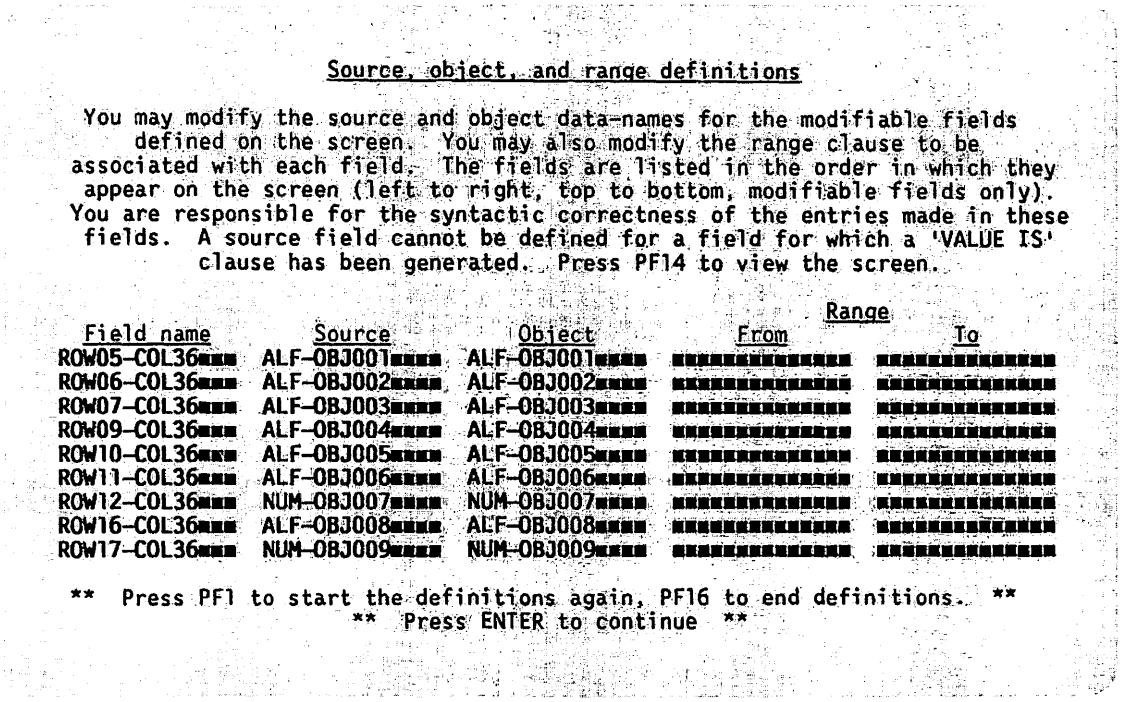
LIBRARY -- A default value is created by concatenating your user ID with the string *FLDS* (e.g., USRFLDS).

VOLUME -- The default volume name is taken from the INVOL field of your usage constants, if this is set.

You can override any of these defaults. Add or change the information as necessary. Then, if the field name file you have specified already exists, press ENTER. If it does not exist, however, and you are creating a new field name file, press PF2.

4.10.2 Defining COBOL Field Names and Ranges

When you finish specifying the field names file, the first Source, Object, and Range Definition screen appears. Figure 4-10 shows an example of this screen.



Source, object, and range definitions

You may modify the source and object data-names for the modifiable fields defined on the screen. You may also modify the range clause to be associated with each field. The fields are listed in the order in which they appear on the screen (left to right, top to bottom, modifiable fields only). You are responsible for the syntactic correctness of the entries made in these fields. A source field cannot be defined for a field for which a 'VALUE IS' clause has been generated. Press PF14 to view the screen.

Field name	Source	Object	Range From	Range To
ROW05-COL36	ALF-OBJ001	ALF-OBJ001		
ROW06-COL36	ALF-OBJ002	ALF-OBJ002		
ROW07-COL36	ALF-OBJ003	ALF-OBJ003		
ROW09-COL36	ALF-OBJ004	ALF-OBJ004		
ROW10-COL36	ALF-OBJ005	ALF-OBJ005		
ROW11-COL36	ALF-OBJ006	ALF-OBJ006		
ROW12-COL36	NUM-OBJ007	NUM-OBJ007		
ROW16-COL36	ALF-OBJ008	ALF-OBJ008		
ROW17-COL36	NUM-OBJ009	NUM-OBJ009		

** Press PF1 to start the definitions again, PF16 to end definitions. **
** Press ENTER to continue **

Figure 4-10. A Sample EZFORMAT COBOL Source, Object, and Range Definition Screen

Each Source, Object, and Range Definition screen has nine fields. If your screen image has more than nine modifiable fields, EZFORMAT displays as many screens as necessary to list them all. You can move from one screen to the next by pressing ENTER.

For each field, the screen displays a name for the field's position, a name for the source and object fields that correpsond to it, and two blank fields to show the range for acceptable input. EZFORMAT generates these variable names, but you can change them to something more appropriate to the nature of the field, and you can also specify the range. You are free to decline these options; EZFORMAT can generate source code from the screen as it appears, using the default variable names for all fields. Range specifications are optional.

The two columns labeled *Source* and *Object* contain fields in which to enter the names by which you want the COBOL program to reference the field whose location is specified at the left. Instead of pseudoblanks, these fields, when the screen first appears, contain dummy names generated by EZFORMAT.

Note: *When you are modifying a screen image that EZFORMAT previously created in COBOL, the utility obtains the field assignments from the field names file (provided this file is still in its original location), and displays them in the correct places on the Source, Range, and Object Definitions screen. Dummy names appear only where you have added new fields to the screen image.*

The *Source* field tells the COBOL program where to obtain input for a field on the screen. When the name of a field in the COBOL program is entered, the value taken from this field will be displayed as a default value in the corresponding field on the screen (i.e., the field identified by row and column coordinates at the left).

The *Object* field tells the COBOL program where to direct output from a modifiable field on the screen. The value entered in this field on the screen (i.e., the field identified by row and column coordinates at the left) is moved to the field in the program whose name you entered in the Object field.

At your option, specify the fields on the Source, Object, and Range Definitions screen as follows:

Field name -- Specify a name that the generated source code will use for a field's screen location. EZFORMAT supplies a default name in the form *ROWxx-COLyy*, where xx and yy are numeric row and column positions.

Note: *The name you specify in a Field Name field must not duplicate a source name, an object name, or a COBOL reserved word.*

Source -- Specify a name for the Source field from which this field will obtain its initial value when the screen is displayed. EZFORMAT supplies a default name that reflects the field's data type and the position in which it occurs on the screen. For example, ALF-OBJ001 indicates an alphanumeric field which is the first input request. NUM-OBJ007 indicates a numeric modifiable field which is the seventh input request.

Note: *If you supplied a default value for the modifiable field when you designed the screen, neither you nor EZFORMAT can assign a Source name. The COBOL program always displays the default value you specified.*

OBJECT -- Optionally, specify a name for the Object field to which the value in this field is moved when the ENTER key (or any defined PF key not in an ON clause of a DISPLAY AND READ statement) is pressed from the displayed screen. EZFORMAT supplies the same default name it supplied in the Source field.

RANGE -- Optionally, specify the lowest (*From*) and highest (*To*) acceptable input values. The COBOL collating sequence determines the interpretation of the upper and lower bounds; see the *VS COBOL 85 Reference* for details.

You have these options from any Source, Object, and Range Definition screen:

PF Key	Option and Description
ENTER	**Continue** -- If there is more than one Source, Object, and Range Definitions screen (i.e., if there are more than nine fields in your screen design), the ENTER key moves you to the next screen, and from the last screen back to the first. If there is only one screen, this key has no effect.
1	**Start the definitions again** -- This option deletes all your entries in the fields on the currently displayed screen and restores the default values.

14 **View the screen** -- Display the screen image as you designed it. Seeing which fields are in which locations helps you identify them correctly on the Source, Object, and Range Definition screen.

16 **End definitions** -- When you are satisfied with the field variable names and ranges, and have no more entries to make from any Source, Object, and Range Definition screen, press PF16 to complete this part of the process and go on to the next part -- specifying the source file (see the next section).

4.10.3 Specifying the COBOL Source File

When you finish defining the field names and ranges and press PF16 from a Source, Range, and Object Definitions screen (Figure 4-10), the COBOL Source File Specification screen (not shown) appears. This screen contains three fields in which to enter the specifications for the file that will contain the COBOL source code.

FILE -- If you specified a screen image file, the file name you supplied appears as a default.

LIBRARY -- EZFORMAT concatenates your user ID with the string *COPY* (e.g., USRCOPY) for a default value in this field.

VOLUME -- EZFORMAT uses as a default the value in the OUTVOL field of your usage constants, if this has been set.

Enter file information in these fields as necessary and press ENTER. The source file is created and EZFORMAT processing is terminated. You are returned to the Command Processor or to the program or procedure from which EZFORMAT was called.

4.11 SPECIFYING RPG II OUTPUT FILES

If you enter *R* or *RPG* in the Option field of the Function Specification screen (Figure 4-1), EZFORMAT creates an RPG II source file. The code in the source file recreates the screen image you designed. In a subroutine, EZFORMAT defines initial values and assigns them to fields corresponding to the modifiable fields described in your screen image. When you incorporate this code in an RPG II program, the program recreates the screen image you designed and accepts any data that is entered in modifiable fields. The RPG II source code also produces calculations to activate the ENTER and PF16 keys when the screen is displayed and to terminate the program when PF16 is pressed.

4.11.1 Defining RPG II Field Names

When you finish specifying the screen image file, or when you press PF2 (Source code generation only) from the Output Options screen (Figure 4-8), the Source and Object Definitions screen appears. Figure 4-11 shows an example of this screen.

```
                        Source and object definitions

  You may modify the source and object data-names for the modifiable fields
    defined on the screen.  The fields are listed in the order in which they
  appear on the screen (left to right, top to bottom, modifiable fields only).
 You are responsible for the syntactic correctness of the entries made in these
    fields.  A source field cannot be defined for a field for which a value has
                 been generated.  Press PF14 to view the screen.


          Row  Column     Source              Object

           05     36      ALF001              ALF001
           06     36      ALF002              ALF002
           07     36      ALF003              ALF003
           09     36      ALF004              ALF004
           10     36      ALF005              ALF005
           11     36      ALF006              ALF006
           12     36      NUM007              NUM007
           16     36      ALF008              ALF008
More fields
**  PF1 - Restart definitions; PF16 - Generate code; PF32 - Terminate  **
                          **  Press ENTER to continue  **
```

Figure 4-11. A Sample EZFORMAT RPG II Source and Object Definition Screen

Each Source and Object Definition screen lists eight fields. If your screen image has more than eight modifiable fields, EZFORMAT displays as many screens as necessary to list them all. You can move from one screen to the next by pressing ENTER.

For each field, the screen lists the field's position and displays a dummy name for the source and object fields that correspond to it. The dummy names remind you whether a given field is alphanumeric or numeric, e.g., ALF001, NUM007. The Source and Object Definitions screen offers an opportunity to change these names to something more appropriate to the nature of the field. If you decline this option, EZFORMAT can generate source code from the screen as it appears, using the default variable names for the fields.

The *Source* field tells the RPG II program where to obtain input for a field on the screen. When you enter the name of a field in the program, the value from this field is displayed as a default value in the corresponding field on the screen (i.e., the field identified by row and column coordinates at the left).

The *Object* field tells the RPG II program where to direct output from a modifiable field on the screen. The value entered in this field on the screen (i.e., the field identified by row and column coordinates at the left) is moved to the field in the program whose name you entered in the Object field.

Specify the fields on the Source and Object Definitions screen, if you choose to do so, as follows:

Source -- Specify the Source field from which this field's initial value is obtained when the screen is displayed. EZFORMAT supplies a default name that indicates the data type and the position in which the field occurs on the screen. For example, ALF001 indicates an alphanumeric field which is the first input request on the screen. NUM007 indicates a numeric field which is the seventh input request on the screen.

Note: *If you supplied a default value for the modifiable field when you designed the screen, neither you nor EZFORMAT can assign a Source name. The RPG II program always displays the default value you specified.*

OBJECT -- Specify the field that receives the value entered in this field on the screen. EZFORMAT supplies the same default that appears in the Source field.

Note: *Although you can assign field names up to 14 characters in length through EZFORMAT, VS RPG II only supports field names whose length is 6 characters or less.*

You have these options from any Source and Object Definition screen:

PF Key	Option and Description
ENTER	**Continue** -- If there is more than one Source and Object Definitions screen (i.e., if there are more than eight fields in your screen design), the ENTER key moves you to the next screen, and from the last screen back to the first. If there is only one screen, this key has no effect.
1	**Restart definitions** -- This option deletes all your entries in the fields on the currently displayed screen and restores the default values.
14	**View the screen** -- Display the screen image as you designed it. Seeing which fields are in which locations helps you identify them correctly on the Source and Object Definition screen.
16	**Generate code** -- When you are satisfied with the field variable names and ranges, and have no more entries to make from any Source and Object Definition screen, press PF16 to complete this part of the process and go on to the next part -- specifying the source file (see the next section).
32	**Terminate** -- Press PF32 (↑16) to exit from EZFORMAT without generating a source code file. You are returned to the Command Processor or to the program or procedure from which EZFORMAT was called.

4.11.2 Specifying the RPG II Source File

When you finish defining the field names and press PF16 from a Source and Object Definition screen (Figure 4-11), the RPG II Source File Specification screen (not shown) appears. This screen contains three fields in which to enter specifications for the file that will contain the RPG II source code.

FILE -- If you specified a screen image file, the file name you supplied appears as a default.

LIBRARY -- EZFORMAT concatenates your user ID with the string *COPY* (e.g., USRCOPY) for a default value in this field.

VOLUME -- EZFORMAT uses as a default the value in the OUTVOL field of your usage constants, if this has been set.

Enter file information in these fields as necessary and press ENTER. The source file is created and EZFORMAT processing is terminated. You are returned to the Command Processor or to the program or procedure from which EZFORMAT was called.

4.12 SPECIFYING OUTPUT FOR THE DATA ENTRY FUNCTION

The output specification process for the EZFORMAT data entry function consists of the following steps:

1. Selecting the source language in which any necessary code is to be generated (Section 4.12.1). The choices are COBOL and RPG II.

2. Choosing whether to save the screen image file, the source code file, or both (Section 4.12.2). EZFORMAT can use the screen image file to reproduce your screen design on the Image Design screen. The source code file contains code for a data entry program that incorporates the screen image you have designed.

3. Specifying the screen image file, if you have chosen to save it (Section 4.12.3).

   Note: *The remaining steps are necessary only if you have chosen to save the source code file.*

4. Specifying the control file that describes the data file this program will be used to update (Section 4.12.4).

5. If you have selected COBOL as the source language in step 1, specifying a field name file, in which the names and definitions of all the modifiable fields on the screen are saved (Section 4.12.5).

6. Associating each field described in your screen image with one of the fields defined in the control file (Section 4.12.6).

7. Specifying the source file for the data entry program (Section 4.12.7).

8. Specifying the object file for the data entry program (Section 4.12.8).

4.12.1 Specifying the Source Language

Since you did not specify a source programming language when you selected the data entry function, EZFORMAT requires you to make this specification before it can save any output. As soon as you press PF16 (End Screen Format Definition) from the Screen Manipulation Options menu (Figure 4-4), the Data Entry Source Language screen (Figure 4-12) appears.

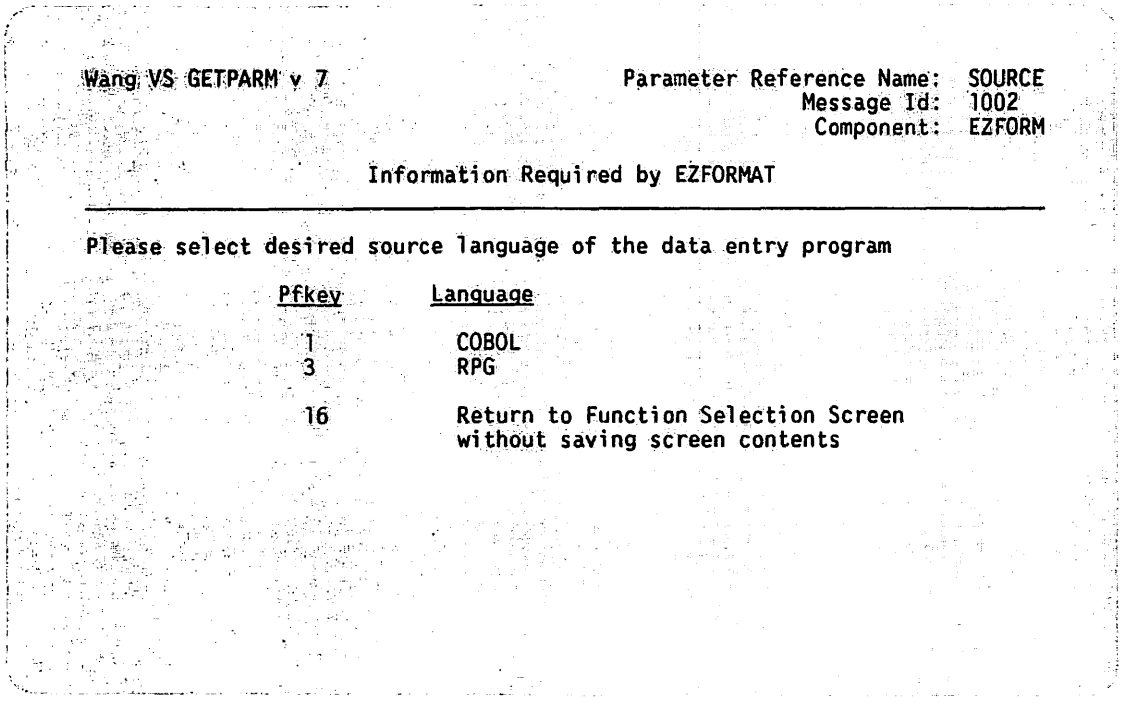


Figure 4-12. EZFORMAT Data Entry Source Language Screen

EZFORMAT uses the programming language you specify to construct both the file that preserves the screen image for EZFORMAT and the source file for the data entry program you are creating. Press PF1 to select COBOL as the source language, or PF3 to select RPG II. Pressing PF16 returns you to the Function Specification screen (Figure 4-1) without saving any work you may have done on the Image Design screen; it is equivalent to starting EZFORMAT processing again from the beginning.

4.12.2 Specifying Output Options

When you press PF1 or PF3 from the Source Language screen, the Output Options menu (Figure 4-8) appears.

Use this screen to tell EZFORMAT whether to save only the screen image file, only the data entry program, or both. The available options are as follows:

PF Key	Option and Description
1	**Return to the Function Specification screen without saving screen contents** -- This is equivalent to starting EZFORMAT processing again from the beginning. Any work you have done on the Image Design screen is lost.
2	**Save generated output only** -- This option saves only the COBOL or RPG II source code for the data entry program. The screen image file is not saved, although the screen you have designed will be generated in this program. If you later want to use EZFORMAT to modify this screen image for another program, you will have to enter all the screen information on a blank Image Design screen.
3	**Save screen contents only** -- This option saves only the screen image file. No source code or data entry program is generated, although you can later run EZFORMAT, reproduce your screen design by calling up the screen image file, and continue the process by generating source code at that time.
4	**Save screen contents and generated output** -- This option saves both the screen image file and the data entry program source file.
16	**Exit without saving any files** -- This option terminates EZFORMAT processing without saving any output. Any work you have done while running the program is lost.

4.12.3 Specifying the Screen Image File

Note: If you are not saving the screen contents (i.e., if you pushed PF2 from the Output Options menu), this section does not apply. Go on to Section 4.12.4.

If you press either PF3 or PF4 from the Output Options menu (Figure 4-8), the Image File Specification screen (not shown) appears. This screen has three fields in which to enter the specifications for the screen image file. If, at the beginning of EZFORMAT processing, you recalled an existing screen image by pressing PF3 from the Function Specification screen (Figure 4-1), the name, library, and volume you specified at that time for the screen image file appear here as defaults.

FILE -- Supply a valid name for the screen image file, unless the field contains a default you want to accept.

LIBRARY -- If you did not specify an existing screen image file, EZFORMAT provides a default library name by concatenating your user ID with the string SAVE (e.g., USRSAVE).

VOLUME -- If you did not specify an existing screen image file, the default is the value of the INVOL field of your usage constants, if it has been set. Otherwise, the field is blank and you must supply the name of the volume where you want the screen image file to reside.

When you have provided specifications for the screen image file, press ENTER. If you are saving only the screen image file (i.e., if you pressed PF3 from the Output Options menu), this action completes EZFORMAT processing. The screen image file is created, and EZFORMAT returns you to the Command Processor or to the program from which EZFORMAT was called. If you are also saving the source file, the Control File Specification screen (Figure 4-13) appears, and the output specification process continues as described in the next section.

4.12.4 Specifying the Control File

Before EZFORMAT can generate a data entry program, you must supply the name and location of a control file that describes the data file you intend this data entry program to update. When you press PF2 from the Output Options menu, or press PF4 and then specify the screen image file, the Control File Specification screen (not shown) appears. It contains three fields in which to enter specifications for the control file:

FILE -- Supply the name of the control file that describes the data file for which this data entry program is being generated. The control file must contain descriptions of all the modifiable fields that appear on the screen.

LIBRARY -- EZFORMAT provides a default library name by concatenating your user ID with the string CTL (e.g., USRCTL). If your control files reside in a different library, enter its name.

VOLUME -- If the INVOL field of your usage constants is set, the value in that field appears here as a default. Enter or correct the volume name as necessary.

When you finish entering the control file specifications, press ENTER.

If the source language is RPG II, you have no choice from this screen other than to enter the name of an existing control file that describes the modifiable fields on the data entry screen. If no such control file exists, you must press HELP and PF16 to cancel out of EZFORMAT.

If the source language is COBOL, you have these options besides specifying the control file:

PF Key	Option and Description
1	**Switch from the data entry option to the COBOL option** -- If the requisite control file does not exist, you can change from one EZFORMAT function to another. The data entry function requires a control file, but the source code generation function does not. Pressing PF1 switches to the latter function. EZFORMAT does not generate a data entry program, but it does generate COBOL source code capable of generating the screen image you have designed. When you press PF1, the process continues with the specification of the field name file as described in the next section.

PF Key	Option and Description
2	**Invoke the CONTROL utility** -- You can use this option to switch from EZFORMAT to the CONTROL utility in order to construct the control file you need. (See Chapter 2 for a description of the CONTROL utility.)
16	**Exit** -- Press PF16 to terminate EZFORMAT processing without generating either the source code or the data entry program.

If you do specify an existing control file, EZFORMAT displays the Control File Field IDs screen. It lists all the modifiable fields in the control file. Figure 4-13 shows an example of an EZFORMAT Control File IDs screen.

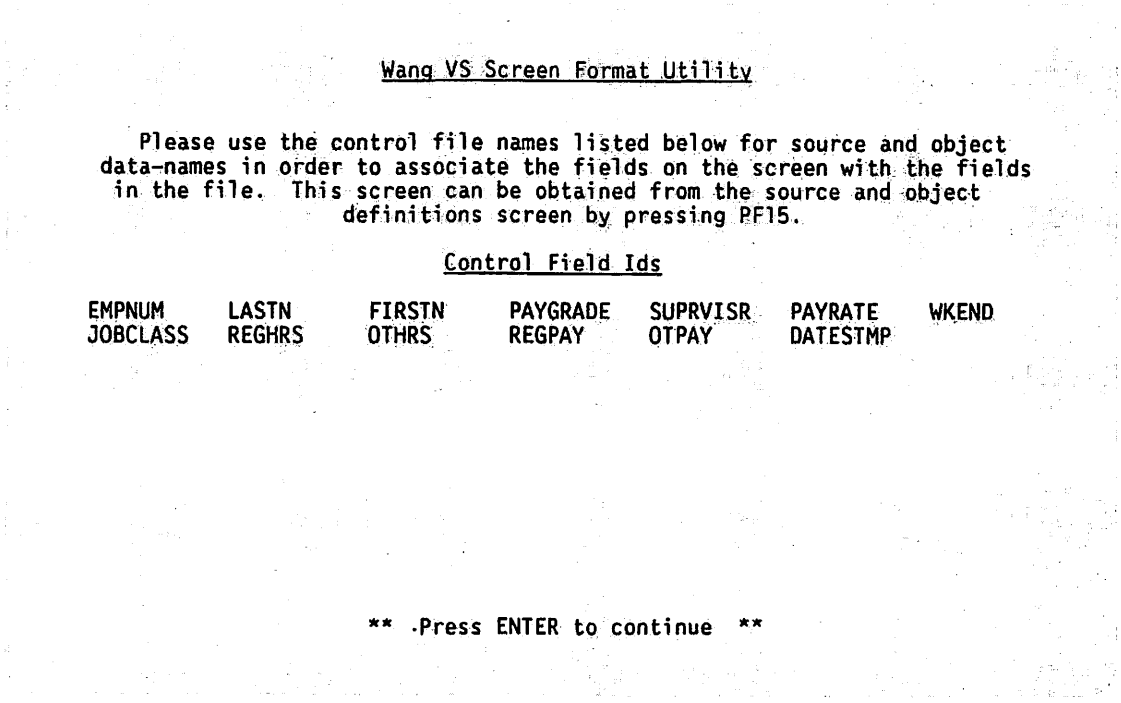


Figure 4-13. A Sample EZFORMAT Control Field IDs Screen

In a later step, you will associate each field you have described in your screen image with one of the fields in the control file. To aid your memory during this process, which is described in Section 4.12.6, you can recall this screen as often as you want. Press ENTER when you are ready to continue.

4.12.5 Specifying the Field Name File

Note: *If you have selected RPG II as the source language, this step is not required. Move on to Section 4.12.6.*

If you have selected COBOL as the source language, the Field Name File Specification screen (not shown) appears when you press ENTER from the Control Field IDs screen. The Field Name File Specification screen also appears when you press PF1 from the Control File Specification screen to switch from the data entry to the source code generation function, as described in the previous section.

Use the Field Name File Specification screen to supply specifications for a file in which the field names and their definitions are saved. When you assign the control fields to the fields described in your screen image (the next step in the process, described in Section 4.12.6), these assignments are also preserved in the field name file. The existence of this file saves you labor if you decide to modify a screen image by adding or deleting modifiable fields. EZFORMAT obtains the original field assignments from the field names file, and you need to specify only the new ones.

The Field Name File Specification screen has the usual three fields in which to enter file information:

FILE -- The default is the name of the control file you specified from the Control File Specification screen (Figure 2-1).

LIBRARY -- A default value is created by concatenating your user ID with the string *FLDS* (e.g., USRFLDS).

VOLUME -- The default volume name is taken from the INVOL field of your usage constants, if this is set.

You can override any of these defaults. Add or change the information as necessary. Then, if the field name file you have specified already exists, press ENTER. If it does not exist, however, and you want to create a new field name file, press PF2.

4.12.6 Assigning Control File Fields to the Screen Image

In order to incorporate the screen image you designed into a program that can enter data into an existing data file, EZFORMAT must identify each modifiable field you described on the screen with a field defined in the specified control file. This control file, which you named from the Control File Specification screen (Figure 4-12), must describe every field the new data entry program will access. In order to be assigned to a field described in your screen image, the control file field must match it in type and length.

If your source language is RPG II, the first Source and Object Definitions screen (Figure 4-14) appears after you specify the control file (as described in Section 4.12.4). If your source language is COBOL, this screen appears after you specify the field name file (as described in Section 4.12.5).

```
                        Source and object definitions

  You may modify the source and object data-names for the modifiable fields
    defined on the screen. The fields are listed in the order in which they
  appear on the screen (left to right, top to bottom, modifiable fields only).
 You are responsible for the syntactic correctness of the entries made in these
  fields. A source field cannot be defined for a field for which a value has
            been generated. Press PF14 to view the screen.
            Press PF15 to view the Control file field names.
  Screen and control file field definitions must be the same size and type.

          Row  Column     Source              Object

           05     36      ALF001              ALF001
           06     36      ALF002              ALF002
           07     36      ALF003              ALF003
           09     36      ALF004              ALF004
           10     36      ALF005              ALF005
           11     36      ALF006              ALF006
           12     36      NUM007              NUM007
           16     36      ALF008              ALF008
  More fields
  **  PF1 - Restart definitions; PF16 - Generate code; PF32 - Terminate  **
                        **  Press ENTER to continue  **
```

Figure 4-14. EZFORMAT Source and Object Definitions Screen (Source Language RPG II)

The COBOL and RPG II versions of the EZFORMAT Source and Object Definitions screen differ slightly. Figure 4-14 illustrates the RPG II version and Figure 4-12 the COBOL version. The lengths of the field names differ (six characters for RPG II, eight for COBOL), and the two screen versions offer different PF key options. Both sets of options are listed further on in this section.

Each Source and Object Definitions screen lists up to eight fields, and EZFORMAT displays as many screens as necessary to list all the modifiable fields in the screen image. There are four columns of information. In the two columns at the left, each field is identified by the row and column position of its first (leftmost) character on the screen.

The two remaining columns, labeled *Source* and *Object*, contain fields in which to enter the field names from the control file. Instead of pseudoblanks, these fields, when the screen first appears, contain dummy names generated by EZFORMAT. (There is an exception in the case of previously created COBOL files.[1]) The dummy names remind you whether a given field is numeric or alphanumeric (e.g., ALF001, NUM002), and otherwise consist simply of consecutive numbers indicating the position in which the field occurs on the screen. (In the COBOL version of the screen, the dummy field names are longer, e.g., ALF-001.)

Your task is to enter the name of a field from the control file in at least one of these columns for each field listed. You can accept the default names only if they correspond to identically named fields defined in the control file.

The *Source* field tells the data entry program where to obtain input for a field on the screen. If you enter the name of a field from the control file, when you run the program and retrieve a record for modification, the value from this field in the record is displayed as a default value in the corresponding field on the screen (i.e., the field identified by row and column coordinates at the left).

The *Object* field tells the data entry program where to direct output from a modifiable field on the screen. If you enter the name of a field from the control file, when you run the program and modify a record, the value entered in this field on the screen (i.e., the field identified by row and column coordinates at the left) is written to the field in the record whose name you entered in the Object field.

You can leave either the source or the object field (but not both) blank. The consequences for each of these options are as follows:

- If you leave the object field blank, the data entry program displays the value from the field named in the source field, but it accepts no input and does not update the record. This is a way to display nonmodifiable fields on your screen if you want to do so.

[1] When you are modifying a data entry program that EZFORMAT previously created in COBOL, the program obtains the field assignments from the field names file (provided this file is still in its original location), and displays them in the correct places on the Source and Object Definitions screen. Dummy names appear only where you have added new fields to the screen image.

- If you leave the source field blank, the program displays a field filled with pseudoblanks. The user can enter information which will in due course replace the value in the specified field in the data record. However, the user has no way of knowing the original value, nor will the program be able to display the modified value when the record is retrieved. If the data entry program will be used to update existing records, not merely to enter new ones, this option is of limited use.

For most fields in most data entry programs, it is appropriate to enter the control field name in both the Source and the Object fields.

The following list summarizes the restrictions that apply to the definition of source and object fields:

- Any data name entered in either the source or the object field must be the name of a field defined in the specified control file.

- For each field listed on the Source and Object Definition screens, either the source or the object field must be specified. Both fields may not be left blank.

- Source fields need not be unique; i.e., the same control field name may be assigned as the source for more than one field on the screen. When the data entry program retrieves a record, the value from this field in the record will be displayed as the default in all of these fields. (You can specify different fields in the record to receive values entered in these fields, as long as you do not use the same field for output more than once -- see the next restriction.)

- Object fields must be unique; i.e., a given control field may not be assigned as object to more than one field on the screen. The data entry program cannot be required to write values from several fields on the screen to the same field in the record.

- When you specify both source and object fields for a given field on the screen, be sure to enter names in both fields. If you enter the control field name in only one field (source or object), EZFORMAT accepts the default name for the other field, with a resultant error when the program is compiled or run.

- If only the source field is specified, the field is displayed but cannot be modified (i.e., it becomes a nonmodifiable field, regardless of how it is defined in the control file).

- If only the object field is specified, the field is modifiable but cannot display a default value.

Since the options available from the Source and Object Definitions screen differ according to the source language, they are described separately in the two sections that follow.

COBOL Version

The Source and Object Definitions screen offers the options listed below when the source language is COBOL.

PF Key	Option and Description
ENTER	**Continue** -- If there is more than one Source and Object Definitions screen (i.e., if there are more than eight fields in your screen design), the ENTER key moves you to the next screen, and from the last screen back to the first. If there is only one screen, this key has no effect. When ENTER is pressed, however, EZFORMAT checks for errors such as field names that do not appear in the control file and displays error messages as appropriate. You cannot proceed until the errors are corrected.
1	**Return to input options** -- This option returns you to the Function Specification screen (Figure 4-1) without generating the source code or the data entry program.
2	**First** -- Return to the first Source and Object Definitions screen.
4	**Previous** -- Return to the previous Source and Object Definitions screen.
6	**Undo** -- This option deletes all your entries in the fields on the currently displayed screen and restores the default values.
13	**Information** -- Display a screen summarizing the information presented in this section.
14	**Screen** -- Display the screen image as it will appear in the data entry program.
15	**Control file field names** -- Display the Control Field IDs screen. By using PF14 and PF15, you can refresh your memory about where the fields appear on the screen and what control fields you want to assign to them.
16	**Generate code** -- When you are satisfied with the field assignments and have no more entries to make from any Source and Object Definitions screen, press PF16 to complete this part of the process and go on to the next part -- specifying the source file (see Section 4.12.7).

RPG II Version

The Source and Object Definitions screen offers the options listed below when the source language is RPG II.

PF Key	Option and Description
ENTER	**Continue** -- If there is more than one Source and Object Definitions screen (i.e., if there are more than eight fields in your screen design), the ENTER key moves you to the next screen, and from the last screen back to the first. If there is only one screen, this key has no effect. When ENTER is pressed, however, EZFORMAT checks for errors such as field names that do not appear in the control file and displays error messages as appropriate. You cannot proceed until the errors are corrected.
1	**Restart definitions** -- This option deletes all your entries in the fields on the currently displayed screen and restores the default values.
14	**Screen** -- Display the screen image as it will appear in the data entry program.
15	**Control file field names** -- Display the Control Field IDs screen. By using PF14 and PF15, you can refresh your memory about where the fields appear on the screen and what control fields you want to assign to them.
16	**Generate code** -- When you are satisfied with the field assignments and have no more entries to make from any Source and Object Definitions screen, press PF16 to complete this part of the process and go on to the next part -- specifying the source file (see Section 4.12.7).
32	**Terminate** -- This option terminates EZFORMAT processing and returns you to the Command Processor (or to the program or procedure from which EZFORMAT was called) without generating the source code or the data entry program.

Whichever source language you use, these are the fundamental commands:

ENTER -- To move to the next screen (after all entries on the current screen are complete and correct), if more than one screen is required to display all of your fields.

PF14 -- To display the screen image you designed, so that you can check the locations, lengths, and labels of the fields.

PF15 -- To display the names of the fields in the control file.

PF16 -- To continue with the code generation process when all field definitions are completed. (Note that ENTER is not used for this purpose.)

4.12.7 Specifying the Source File

When you press PF16 (Generate code) from the Source and Object Definitions screen (Figure 4-14), the Data Entry Source File Specification screen (not shown) appears. This screen contains three fields in which to enter the specifications for the file that will contain the COBOL or RPG II source code for the data entry program.

FILE -- The default value is the name you supplied for the control file.

LIBRARY -- For the COBOL version, EZFORMAT uses as a default the value in the OUTLIB field of your usage constants, if this has been set. For the RPG II version, EZFORMAT concatenates your user ID with the string *COPY* (e.g., USRCOPY).

VOLUME -- EZFORMAT uses as a default the value in the OUTVOL field of your usage constants, if this has been set.

Enter file information in these fields as necessary and press ENTER to continue.

4.12.8 Specifying the Object File

When you press ENTER from the Source File Specification screen, the Object File Specification screen (not shown) appears. This screen has three fields in which to enter specifications for the file that will contain the executable object code for the data entry program.

FILE -- The default value is the name you supplied for the control file.

LIBRARY -- EZFORMAT uses as a default the value in the OUTLIB field of your usage constants, if this has been set.

VOLUME -- EZFORMAT uses as a default the value in the OUTVOL field of your usage constants, if this has been set.

Enter file information in these fields as necessary. (Note that, if you accepted your OUTLIB and OUTVOL defaults for the source file, you cannot accept the same defaults for the object file without having to scratch the source file.) When your specifications are complete, press ENTER. EZFORMAT creates the source and object files for the data entry program and terminates processing, returning you to the Command Processor or to the program or procedure from which the utility was called.

4.12.9 Running the Data Entry Program

To run a data entry program generated by EZFORMAT, press PF1 (Run program or procedure) from the Command Processor menu. Specify the name of the data entry object file and the library and volume where it resides; press ENTER. This section provides a brief summary of the way the program operates.

1. When you invoke the data entry program, a screen appears that prompts you to specify the data file the program is designed to interact with. The data file's records must be described in the control file you specified while creating the data entry program. You must specify an existing data file; the data entry program cannot create one. (If such a file does not exist, however, you can use the DATENTRY utility to create it without records; you can then use the new program to place records in the file.)

2. After you specify an existing data file, a Find Mode screen appears, from which you are prompted to specify a particular record for modification. (The means of specification vary according to the file structure. They are essentially the same as the methods described for the DATENTRY utility in Section 3.4.2.)

3. If you specify a particular record, the data entry screen you designed using EZFORMAT appears, with the appropriate value from the data record displayed in each field. The screen is labeled "Display Mode," and the fields do not accept data. However, you have these options from the display screen:

PF Key	Options
PF1	Return to the Find Mode screen.
PF2	Display the first record in the file.
PF5	Display the next record.
PF9	Modify the record. When you press PF9, the modifiable fields are highlighted and will accept data. You can enter new values and press ENTER to update the record.
PF12	Delete the record.

4. Besides locating records from the Find Mode screen, you can switch to Add Mode by pressing PF11. Your data entry screen is displayed with blank fields. You can enter data in the fields and press ENTER to store the record and clear the fields for the the next record. You can add records indefinitely, or return to the Find Mode screen at any time by pressing PF1.

5. To terminate processing, press PF16 from the Find Mode screen.

CHAPTER 5
INQUIRY

5.1 INTRODUCTION

The INQUIRY utility provides a quick, informal, and interactive means of interrogating a data file. (For a more formal and more powerful means of retrieving data, use the REPORT utility, described in Chapter 6.) To retrieve information from a data file through INQUIRY, you need only formulate a query in language that is conveniently close to conversational English. INQUIRY then finds all the records in the data file that meet the criteria specified in the query. What it does with these records depends on which of the two program functions you have specified:

- You can have INQUIRY display on the workstation screen any combination of fields from the records it has retrieved. (If you choose this function, you specify the fields in your query.)

- You can have the retrieved records written into a new data file, extracting a subset of the original data file that consists only of records that meet the specified criteria. (Each record is written in its entirety; this option does not permit individual fields to be selected.)

As an example, consider a data file containing employees' weekly payroll records (like the one used for an example in the preceding chapters). The records in this file contain such information as each employee's first and last names, employee number, job classification, supervisor's name, regular hours and pay for a particular week, overtime hours and pay, and so on.

- You could use INQUIRY's Display function to select, for example, all the records for employees who worked more than four hours of overtime and display, for each employee in this category, whatever information you choose: name, employee number, supervisor, pay rate -- any combination of fields you care to specify from the data record.

- You could use INQUIRY's Extract function to find, for example, the records of all employees in a particular job classification, or all employees whose total pay for the week was greater than $500, or both -- any combination of criteria that can be read from the fields in the data record. INQUIRY writes all the records that meet your specifications into a new data file for further sorting and analysis.

5.1.1 Overview

INQUIRY processing has three parts:

1. From the initial program screen, you specify the data file to be interrogated and select either Display or Extract as the program function.

2. INQUIRY then displays a blank screen on which you formulate your query.

3. When the query is entered, the interrogation takes place, and it is necessary to dispose of the program's output. The first step differs according to the function specified:

   Display -- INQUIRY links to the DISPLAY utility and shows the results of the interrogation on your workstation screen. (Through one of DISPLAY's options you can print a hard copy of these results.)

   Extract -- INQUIRY prompts you to specify a name and location for the output file.

   When you have reviewed or printed the results produced through the Display function, or specified the output file created through the Extract function, INQUIRY offers certain options, including exiting the program or restarting it so that you can formulate another query. However, you can also save your query so that you can recall it during a later INQUIRY session without having to reenter the specifications, and create a Report Definition file (RDF) that can be used with the REPORT utility to extract the same information from the data file.

5.1.2 Software Requirements

For INQUIRY to function properly, the DISPLAY utility must reside either in the system library or in the same library as INQUIRY.

INQUIRY requires any data file it interrogates to have an associated control file containing a complete description of the data record and each of its fields. For information about control files and the way the CONTROL utility is used to create them, see Chapter 2.

5.1.3 Running INQUIRY

You can run the INQUIRY utility by pressing PF1 (Run Program or Procedure) from the Command Processor menu or by pressing PF↑10 (PF26) from the CONTROL Utility menu (see Section 2.7).

The following sections describe INQUIRY processing:

Section	Process
5.2	Specifying the Data File and the INQUIRY Function
5.3	Specifying the Query
5.3.1	Using INQUIRY Syntax To Formulate the Query
5.3.2	Sample Queries
5.3.3	Entering the Query
5.4	Managing INQUIRY Output
5.4.1	Display Function Output
5.4.2	Extract Function Output
5.4.3	Final INQUIRY Options

5.2 SPECIFYING THE DATA FILE AND THE INQUIRY FUNCTION

When INQUIRY processing begins, the Input Options screen (Figure 5-1) appears first.

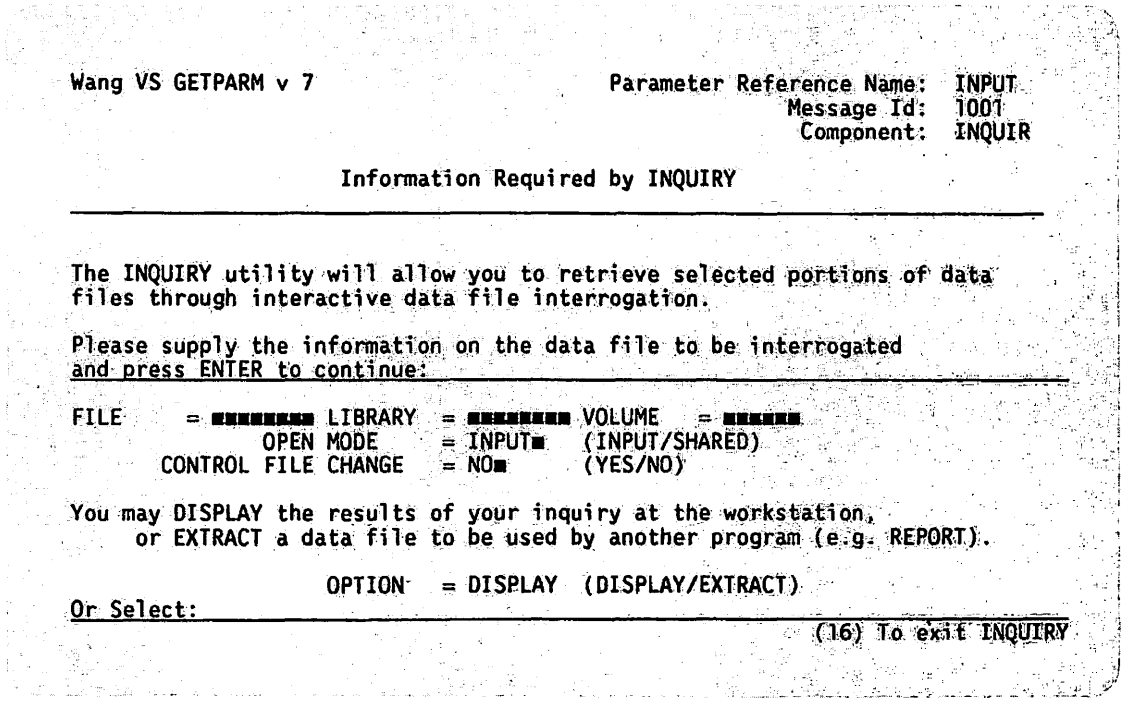


Figure 5-1. INQUIRY Input Options Screen

Use this screen to specify the data file you want INQUIRY to interrogate and to choose the program function. You can specify any data file that has an associated control file. INQUIRY relies on the control file for descriptions of the data record and its component fields. Besides specifying the name and location of the data file, you can tell INQUIRY to open it in Input or Shared mode, and indicate whether or not the control file has the expected name and location.

The last field on the screen is used to specify the program function. The display function causes INQUIRY to list the results of the interrogation on the workstation screen. Only the fields you specify in your query are listed. The extract function causes INQUIRY to write all the records that meet the specifications in your query to a new output file. (Each record is written in full; fields cannot be selected with this option.)

Enter information in the fields as follows:

FILE -- Specify the name of the data file you want to interrogate.

LIBRARY -- If the INLIB field of your usage constants is set, the value from that field appears as a default.

VOLUME -- INQUIRY takes the default value from the INVOL field of your usage constants, if it is set.

OPEN MODE -- INQUIRY supports two open modes for data files: Input (the default) and Shared. Indexed data files may be opened in either mode, but consecutive and relative files must be opened in Input mode.

Input mode -- Allows the records of a file to be read but not to be modified. This is the way INQUIRY always treats data files -- it reads but does not update them. If you select this mode, INQUIRY can open the file only if no other program is currently *updating* it. Specifying Input mode thus insures that the contents of the data file do not change during processing. (More than one user or program can have a file open simultaneously in Input mode, since neither user nor program can change the file while it is open.)

Shared mode -- For indexed files only, allows records to be read and modified by more than one user or program. (A given record, however, is accessible to only one user or program at a time.) If you select Shared mode, INQUIRY can open the indexed file even if another program such as DATENTRY has already opened it in the same mode. Although INQUIRY does not update records, the other program may do so during INQUIRY processing, with the result that two identical queries processed at separate times during the same session may have different results.

CONTROL FILE CHANGE -- INQUIRY starts with the assumption that the control file associated with the specified data file has the same file name and resides on the same volume, in a library whose name is formed by concatenating your user ID with the string *CTL* (e.g., USRCTL). If this assumption is false, an error results and you are prompted for the correct control file specification. However, you can change the default NO in this field to YES to notify INQUIRY that the name or location of the control file does not accord with its assumption. You can then supply file information from the Control File Specification screen (not shown), which appears when you press ENTER from this screen. (By making it unnecessary to specify a differently named control file interactively, this field facilitates running INQUIRY through a procedure.)

OPTION -- Specify either DISPLAY (the default) or EXTRACT as the program function.

- When you select *DISPLAY*, your query must specify the fields you want listed as well as the selection criteria that determine which records are retrieved. INQUIRY retrieves all records that meet the criteria and uses the DISPLAY utility to list the specified fields from each record on the workstation screen. (Records are listed in consecutive or primary key order, depending on the file organization -- it is not possible to sort on alternate keys during INQUIRY processing.)

- When you specify *EXTRACT*, your query needs to specify only the selection criteria. When INQUIRY retrieves all the records that meet these criteria, it creates a new output file and copies the selected records into it. (You are prompted to specify this output file at a later stage in INQUIRY processing -- see Section 5.4.2.)

You can override any of the defaults in these fields. Add or change information as necessary, and press ENTER to continue, or, if you decide instead to terminate INQUIRY processing without interrogating a data file, press PF16.

5.3 SPECIFYING THE QUERY

When you have specified the data file and the program function (and, if necessary, the control file), INQUIRY displays the Query Specification screen (Figure 5-2).

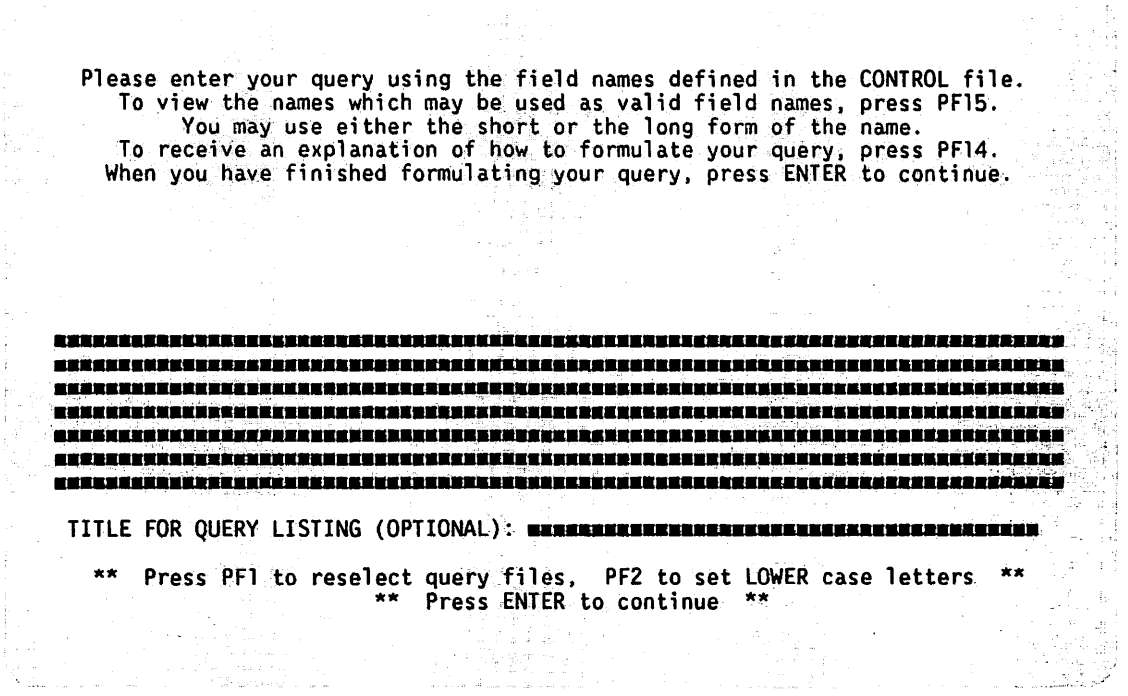


Figure 5-2. INQUIRY Query Specification Screen

This screen serves as a work slate on which to enter your query, which you formulate according to the rules described in the next section.

Note: *If you enter a quote string as part of your query, it must start and end on the same line. If it does not, you will receive an error message.*

For the display function, you can also enter a title to appear at the head of the listing INQUIRY produces on your workstation screen. The field for this title is at the bottom of the screen following the label "TITLE FOR QUERY LISTING (OPTIONAL)."

You have the following options from the INQUIRY Query Specification screen:

PF Key	Option and Description
1	**Reselect query files** -- Press PF1 to return to the Input Options screen (Figure 5-1), from which you can select another data file to interrogate or terminate INQUIRY processing.
2	**Set lowercase letters** -- In its standard input mode, INQUIRY converts all letters entered from the Query Specification screen to uppercase. This is required for field names, but when you enter specific values for certain data fields it may be necessary to express them in lowercase letters. Pressing PF2 changes the input mode so that letters can be entered in lowercase. (In this mode the SHIFT key enables you to enter uppercase letters.) Press PF2 again to return to standard (uppercase) input mode.
14	**Receive information** -- Press PF14 to display two screens that summarize the information about query syntax presented in the next section.
15	**View valid field names** -- When you press PF15, INQUIRY displays a list of the fields in the data file. If aliases were specified for any of these fields when they were being defined through the CONTROL utility (see Section 2.3.2), these are listed also. (The field names appear in a column headed *Short Form,* and the corresponding aliases in a column headed *Long Form.*) Your query must use the correct field name or alias to reference any field. This option helps you recall the valid names while you are composing the query.

When you finish formulating your query, press ENTER. INQUIRY reformulates the query and displays it in the upper part of the Query Specification screen. Your options at this point are described in Section 5.3.3.

5.3.1 Using INQUIRY Syntax to Formulate The Query

Note: *Most examples in this and the following sections use fields defined for the data file PAYREC, used as an example throughout Part I of this manual. Chapter 2 contains information about the field definitions.*

To be comprehensible to INQUIRY, a query must be stated according to a particular syntax, but the syntax INQUIRY uses is constructed so that you can ask your question in as English-like a way as possible. The format for a standard query specifies two parts (only one of which, however, is needed for the extract function). These parts are the list clause and the relation clause.

- *The list clause* specifies the fields whose values are to be listed in the Display function. It has no meaning to the Extract function, for which fields are not specified.

- *The relation clause* indicates the criteria by which records are to be selected from the data file. In the Display function, the values from the specified fields in these records are then listed on the workstation screen. In the Extract function, the selected records are written in their entirety to an output file.

The next two sections describe the list and relation clauses in detail. Section 5.3.2 presents and discusses some sample queries.

The List Clause

Note: *Because entire records are written to the extracted file, the list clause has no meaning for the extract function. If your query is written for this function, therefore, the list clause can be omitted. If a list clause is included in a query written for the extract function, INQUIRY ignores the clause.*

The list clause specifies the fields whose values the Display function will list. For each record that satisfies the conditions stated in the relation clause, INQUIRY displays the value from each of these fields (as well as each field named in the relation clause) on the workstation screen.

The list clause consists of a *list verb* followed by one or more *field names*.

List verbs -- INQUIRY recognizes the following as list verbs: *LIST, DISPLAY, FIND, WRITE, PRINT, SHOW,* and *GIVE*. As far as INQUIRY is concerned, these verbs are exact synonyms. Since the program does not associate a distinct meaning with each verb, the user can choose whichever verb best fits the particular query.

Field names -- A field name in a list clause can be the name of any field defined in the control file, or, provided one has been specified, the *field alias*.

A field alias is specified when the field is defined or modified through the CONTROL utility. If a field already has such an alias, it appears after the field name (in the Long Form column) when you press PF15 to list the fields in the control file. You can provide an alias for a field that lacks one by running the CONTROL utility, selecting the field for modification, and entering a name up to 31 characters long in the *Field Alias field* of the Field Specification screen (Figure 2-5). See the description of this field in Section 2.3.2 for more information.

To construct a list clause, simply follow any list verb with the names or aliases of those fields whose values you want the display to include. For example:

List Verb	**Field Names**
LIST	ABLE BAKER DOG FOX
PRINT	LASTN FIRSTN EMPNUM JOBCLASS PAYGRADE
SHOW	REGULAR HOURS OVERTIME HOURS TOTAL HOURS

The last example uses aliases rather than control file field names.

INQUIRY also displays the values of all fields mentioned in the relation clause. However, a field mentioned in both list and relation clauses is displayed only once, in the position it occupies in the list clause.

The Relation Clause

The relation clause expresses the conditions that the records in the data file must meet in order to be included in the results of the query. INQUIRY locates all records that meet these conditions and either displays the listed fields (if the Display function has been specified) or writes the records to an output file (if the Extract function has been specified).

The simple relation clause consists of a *field name* followed by a *relational operator* followed by a *reference value*. The relation clause of a query may consist of one simple relation clause, or it may be compounded of multiple simple relation clauses joined together with *connectors*.

Field name -- A field name is the name or alias of a control file field, exactly as it is for the list clause.

Relational operator -- INQUIRY recognizes six relational operators. Each one can be expressed in a two-letter code, a mathematical symbol or pair of symbols, or an English phrase. Table 5-1 lists the six relational operators.

Table 5-1. INQUIRY Relational Operators

Letter Code	Symbol(s)	Phrase
EQ	=	Equal to
NE	NOT =	Not equal to
LT	<	Less than
GT	>	Greater than
LE	<=	Less than or equal to
GE	>=	Greater than or equal to

Reference value -- The contents of the field named at the head of the relation clause are compared to a value specified in this part of the clause. This reference or comparison value can be expressed in one of four forms:

Numeric literal -- a digit or sequence of digits that may contain a sign, a decimal point, or both. *Only a numeric field (i.e., a field whose type is Zoned, Packed, or Unsigned) can be compared to a numeric literal.* Example:

```
PAYRATE > 7.50
```

This relational clause selects all records in which the value in the PAYRATE field exceeds ($)7.50.

Character literal -- Any displayable character or string. Character literals must be enclosed in quotation marks. (These may be either single or double, but they cannot be mixed -- the leading quotation mark must be the same as the trailing one). *Only an alphanumeric field (i.e., a field whose type is Character or Binary) can be compared to a character literal.* Example:

```
DATESTMP LT '01/01/88'
```

This relational clause selects all records whose date stamp field has a value less than 01/01/88, i.e., all records not updated since the beginning of 1988. (Note that, for this example to work, the DATESTMP field must be alphanumeric and of type 1. See the description of the Date Stamp field of the Field Specification screen in Section 2.3.2 for detailed information.)

Hexadecimal literal -- Any value expressed in the form *X'hh...h'* where *h* is one of an even number of hexadecimal digits; valid digits include 0 to 9 and A to F. For example, X'0A' and X'1C489F0DC6' are valid hexadecimal literals. X'1AB' is invalid because the number of digits is odd (it could be corrected by changing it to X'01AB'), and X"005E" is invalid because the quotation marks are double. *Only an alphanumeric field (i.e., a field whose type is Character or Binary) can be compared to a hexadecimal literal.* Example:

```
FOO GREATER THAN OR EQUAL TO X'00'
```

This relational clause selects all records in which the value of the field FOO is greater than or equal to hexadecimal 0 -- that is, it selects all records.

Field name -- The name of any other field of the same type. For each record, INQUIRY compares the value in the field specified at the head of the relational clause with the value in this field, according to the relational operator between the two field names. *Numeric fields can be compared only with other numeric fields, and alphanumeric fields only with other alphanumeric fields.* Example:

```
OTPAY >= REGPAY
```

This relational clause selects all employee pay records in which the amount of overtime pay is greater than or equal to the amount of regular pay.

Connector -- To express multiple selection criteria, multiple simple relation clauses (each having the form *FIELD NAME - RELATIONAL OPERATOR - REFERENCE VALUE*) are joined together with one of the two connectors INQUIRY recognizes -- AND and OR. When the connector is AND, the relations expressed in the clauses that precede and follow it must *both* be true for the record to be selected. When the connector is OR, INQUIRY selects the record if *either* relation is true.

In a compound relation clause involving several simple clauses and connectors, all AND relations are evaluated first. Each AND result is then treated as a single criterion to be used by the OR operators. Consider the following example, in which each bold letter represents a simple relation clause such as FIELD1 > FIELD2, FIELD3 LE 500, etc.:

Original formulation: **A AND B AND C OR C AND D OR E**

After evaluation of ANDs: **ABC OR CD OR E**

For this example, INQUIRY selects

1. All records in which clauses A, B, and C are all true
2. All records in which clauses C and D are both true
3. All records in which clause E is true

The format of the simple relation clause -- FIELD NAME - RELATIONAL OPERATOR - REFERENCE VALUE -- is invariable. INQUIRY identifies a relation clause solely by the presence of a field name (or alias) prior to a relational operator. Any field names that come before a field name in this position are considered part of the list clause, if one is present. When a query does not begin with a list verb, multiple field names preceding a relational operator cause an error.

This syntactic rule makes it impossible to specify a criterion for more than one field in a simple relation clause. A compound relation clause joining multiple simple relation clauses is therefore the only way to compare multiple fields to the same reference value: e.g., FIELD1 GE 100 AND FIELD2 GE 100 AND FIELD3 GE 100, etc.

For the display function, if the complete relation clause mentions any fields not mentioned in the list clause, these fields are displayed after the fields named in the list clause, in order of their occurrence in the relation clause.

5.3.2 Sample Queries

The following list of sample queries illustrates the result of each query for both the display and the extract options.

```
LIST EMPNUM FIRSTN LASTN GE 'M'
```

Display -- The values of the Employee Number, First Name, and Last Name fields are displayed for every employee whose last name begins with *M* or a subsequent letter.

Extract -- All records for employees whose last name begins with *M* or a subsequent letter are written to a file.

```
SHOW PRNTCHAR WHERE PRNTCHAR >= X'20' AND PRNTCHAR <= X'FF'
```

Display -- All values for the PRNTCHAR field from hexadecimal 20 to hexadecimal FF (i.e., all values equivalent to printable ASCII characters) are displayed. (PRNTCHAR is a 1-character field.)

Extract -- All records for which the PRNTCHAR field contains a value within the specified range are written to a file.

```
PAYRATE LESS THAN 7.50
```

Display -- Since there is no list clause, this query would cause an error.

Extract -- The records for all employees paid less than $7.50 per hour are written to a file.

```
PRINT EMPNUM LASTN SUPRVISR PAYRATE GE 7.50 AND OTPAY EQ 0
```

Display -- The employee number, last name, supervisor, pay rate, and overtime pay are displayed for all employees paid $7.50 per hour or more who earned no overtime pay.

Extract -- The records for all employees paid $7.50 per hour or more who earned no overtime pay are written to a file.

```
FIND LASTN FIRSTN EMPNUM SUPRVISR = "WILSON" AND OTPAY > 200 OR
SUPRVISR = "WILSON" AND OTHRS > 20 OR SUPRVISR = "WILSON" AND
TOTPAY > 500
```

Display -- The fields LASTN, FIRSTN, EMPNUM, SUPRVISR, OTPAY, OTHRS, and TOTPAY are displayed for each record of an employee whose supervisor is Wilson and who, for the week covered by the record, also satisfies any one of the following three criteria:

- Earned more than $200 in overtime pay
- Worked more than 20 hours of overtime
- Earned more than $500 in total pay

Extract -- The records of all employees who work for supervisor Wilson and who also satisfy any one of the three listed criteria are written to a file.

In the last example, note that the relation clause containing the value for the field SUPRVISR must be repeated three times, once for each of the other criteria. If it appeared only the first time, INQUIRY would retrieve the records for employees in Wilson's group who earned more than $200 in overtime pay, but it would retrieve the records for *all* employees who worked more than 20 overtime hours or earned more than $500 total, whether or not Wilson was their supervisor.

The descriptions and examples above establish only the essential components of a query. Since INQUIRY ignores words it does not recognize or consider necessary, you can formulate the query in a readable English sentence, using whatever punctuation seems appropriate to you. The following queries produce the same results as those in the list of examples:

```
PLEASE LIST THE LASTN, FIRSTN, AND EMPNUM FIELDS FOR ALL RECORDS FOR
WHICH THE LASTN FIELD IS GREATER THAN OR EQUAL TO "M".  THANK YOU.

DISPLAY THE PRINTCHAR FIELD FOR VALUES OF PRINTCHAR > X'20' AND
PRINTCHAR <= X'FF'.

EXTRACT THOSE RECORDS FOR EMPLOYEES WHO HAVE AN HOURLY PAYRATE LESS
THAN 7.50 PER HOUR.

LIST THE EMPNUM, LASTN, AND SUPRVISR FIELDS FOR ALL EMPLOYEES WHOSE
HOURLY PAYRATE IS GREATER THAN 7.50 AND WHOSE OTHRS WERE GREATER
THAN 0.
```

AT YOUR EARLIEST CONVENIENCE, PRINT LASTN, FIRSTN, AND EMPNUM FOR ALL EMPLOYEES WHOSE SUPRVISR = "WILSON" AND WHOSE OTPAY IS GREATER THAN 200 DOLLARS, OR WHOSE SUPRVISR = "WILSON" AND WHOSE OTHRS ARE GREATER THAN 20, OR WHOSE SUPRVISR = "WILSON" AND WHOSE TOTPAY IS GREATER THAN 500 DOLLARS.

You could make the query even more English-like if you assigned aliases to the field definitions so that these could be used in place of field names, for example, PLEASE LIST THE *LAST NAME, FIRST NAME,* AND *EMPLOYEE NUMBER* FOR ALL EMPLOYEES WHOSE *LAST NAME* BEGINS WITH A LETTER GREATER THAN OR EQUAL TO "M" (aliases shown in italics).

Nevertheless, the necessity of observing INQUIRY's rules of syntax impose certain constraints that may prevent you from phrasing the query in a completely natural English style. You could not, for example, state the condition in the same example as WHOSE LAST NAME BEGINS WITH M OR A SUBSEQUENT LETTER, since INQUIRY must see either the words *GREATER THAN OR EQUAL TO,* the code *GE,* or the symbols >= in order to understand the clause, and would not recognize M as the comparison value unless it was enclosed in quotation marks. Similarly, in the last example above, you could not say WHOSE SUPERVISOR IS 'WILSON' (even if you defined SUPERVISOR as an alias for SUPRVISR and put WILSON in quotation marks), since INQUIRY cannot recognize IS as the equivalent of IS EQUAL TO.

Given these constraints, however, INQUIRY allows the user a good deal of freedom in formulating queries, and when it finds a mistake that prevents it from interrogating the file, it displays an error message to explain the difficulty.

This concludes the description of INQUIRY syntax. Section 5.3.3 describes the process of entering your query when it has been formulated.

5.3.3 Entering the Query

When you finish entering your query in the seven lines provided on the Query Specification screen (Figure 5-2), press ENTER. If INQUIRY cannot recognize a syntactically valid query in what you have entered, it displays an appropriate error message. If the query appears valid, however, INQUIRY displays the Query Confirmation screen. Figure 5-3 shows an example of this screen.

```
The QUERY which you have entered has been reduced to the form displayed below.
          Press PF16 if this is equivalent to what you desire.
     Press PF14 for query instructions, Press PF15 to view field names.

LIST EMPNUM LASTN SUPRVISR ; PAYRATE > 7.50 AND OTHRS > 0




LIST THE EMPNUM, LASTN, AND SUPRVISR FIELDS FOR ALL EMPLOYEES WHOSE HOURLY
PAYRATE IS GREATER THAN 7.50 AND WHOSE OTHRS WERE GREATER THAN 0

TITLE FOR QUERY LISTING (OPTIONAL): RATE $7.50 OR OVER; NO OVERTIME

  **  Press PF1 to reselect query files,  PF2 to set LOWER case letters  **
                      **  Press ENTER to respecify **
                      **  Press PF16  to continue  **
```

Figure 5-3. A Sample INQUIRY Query Confirmation Screen

The Query Confirmation screen is quite similar to the Query Specification screen. At the top of the screen, however, INQUIRY summarizes your query in a version of its syntax that reduces the query to essentials, omitting all nonessential words. INQUIRY translates all list verbs to LIST and marks the separation between the list and relation clauses with a semicolon. It also translates all relational operators to the appropriate mathematical symbols.

The purpose of this screen is to allow you to see how INQUIRY understands your query so that you can change the query if necessary before final entry. You have the same options as from the Query Specification screen (see Section 5.3), but a new one is added: if you are satisfied that the query is correctly formulated, press PF16 to begin the interrogation of the data file.

If you make any changes in the formulation of your query, however, press ENTER instead of PF16. The Query Confirmation screen reappears, with INQUIRY's interpretation of the newest version displayed at the top. You can make as many changes in this manner as you wish, always pressing ENTER to make sure that INQUIRY understands the query the way you meant it.

Note: *Do not press PF16 until what you see at the top of the screen satisfies you that INQUIRY has understood your query and will return the information you are seeking. It is this listing that determines how INQUIRY will interrogate the data file, so if you make a change in the modifiable field below and press PF16 instead of ENTER, the change will have no effect on the interrogation. In order to make a change effective, you must press ENTER and confirm that the change appears in the expression at the top of the screen. Only when there are no further changes to make should you press PF16.*

A query may contain errors that INQUIRY cannot detect until it begins the interrogation -- when, for example, a numeric field is compared to a character literal. Since INQUIRY does not check field types until it accesses the file, errors of this kind remain hidden during the syntax check that takes place earlier in the process. If it detects an error at this point, the program redisplays the Query Specification screen (Figure 5-2) with an appropriate error message.

If the query contains no errors, INQUIRY selects records from the data file according to your specifications. Their disposition depends on the program function. The next section describes this part of the process.

5.4 MANAGING INQUIRY OUTPUT

5.4.1 Display Function Output

When you have specified your query and corrected any errors, INQUIRY begins the interrogation process as soon as you press PF16 from the Query Confirmation screen. If you specified the display function, INQUIRY links to the DISPLAY utility and begins listing the specified fields on the workstation screen in a conveniently readable format. Figure 5-4 shows an example of an INQUIRY Display screen.

```
(1)Menu          (3)Position          (5)Next          (7)Up (8)Find
                                                    (15)Print (16)Exit

Column    1                                                  Column   80
     11/04/88     EMPLOYEE OVERTIME HOURS                 PAGE 0001

      LASTN        EMPNUM      OTHRS

SMITH             011010001   00012.00
WILLIAMS          116566202   00008.00
BERLINER          179872254   00012.00
WINDHAM           179879865   00008.00
KIM               193445022   00012.00
MASSON            242565655   00008.00
BRICKER           265376625   00006.00
FOSTER            278398701   00008.00
PHELAN            286993799   00005.00
RAMIREZ           290054323   00008.00
FLYNN             297286742   00008.00
SATTERLEY         298378972   00008.00
FLANAGAN          376928376   00006.00
MICHALSKI         379200166   00012.00
SCIBELLI          379289871   00008.00
WOYCZEK           448296389   00008.00
```

Figure 5-4. A Sample INQUIRY Display Screen

Each page of the listing is headed with the title you supplied from the Query Specification screen, or, if you did not supply a title, with the default title "RESULTS OF INQUIRY." You can use the standard DISPLAY utility commands for moving through the listing; for details, see the description of DISPLAY in Chapter 11.

As the illustration shows, records from an indexed file are listed in primary key order (here the key is EMPNUM or employee number) for indexed files. Records from consecutive files are listed consecutively from the beginning of the file to the end. INQUIRY selects records, but cannot sort them. If it is important that you see a sorted INQUIRY listing, you can use the SORT utility, described in Chapter 14, to create one. SORT can take any data file as input and produce a new file that contains the same records sorted in order of the values in one or more fields that you specify.

For large quantities of complex data, it would be preferable to use the REPORT utility (described in Chapter 6), which can both select and sort records at the same time. However, it is not difficult to presort records for informal interrogation by INQUIRY by means of the following steps:

1. First, run the INQUIRY utility, specifying the extract function in order to select all the records you want to list and put them into a smaller file. This makes the SORT processing faster and more efficient than it would be if you sorted the entire data file.

   Example: To select the records of all employees who worked overtime, you can use the relation clause of the query that produced the listing in Figure 5-4: *OTHRS* > *0*.

2. In order to sort data records by value, SORT needs the exact position, length, and data type of each field to be sorted on. You can provide this information easily if you run the CONTROL utility before you run SORT. Specify the control file associated with the original data file, since it also describes the records in the extracted file. Display a listing of the fields, and write down the position, length, and data type of each field you intend to specify a sort on.

   Example: By examining the control file PAYREC, you can determine the position, length, and type of the fields OTHRS, LASTN, and FIRSTN.

3. Run the SORT utility, specifying as the input data file the file extracted from the original data file by INQUIRY, which contains all and only the records you want to examine. Specify the fields on which you want these records to be sorted and the order (ascending or descending) of each sort.

   Example: One way to sort the records extracted from PAYREC is to specify the following criteria:

   1) A descending sort on OTHRS
   2) An ascending sort on LASTN
   3) An ascending sort on FIRSTN

   These specifications cause the records to be sorted first in order of the amount of overtime (largest amounts first), then alphabetically by last name, and finally by first name.

4. SORT's output is a consecutive file that contains the same records as the input file, but the records are placed in order according to your specifications. Using this file as input, you can run INQUIRY, this time specifying the display function. The records are displayed in the order in which they occur in the file, i.e., the order into which you have presorted them.

   If your original data file was not a consecutive file, an intermediate step is necessary. INQUIRY requires a control file as well as a data file specification, but it cannot accept a control file that describes an indexed or relative data file, since this does not match the organization of the consecutive file you are interrogating. You must therefore (1) use the COPY utility to make a copy of the original control file, (2) run the CONTROL utility on the copied file to change the file organization it describes to consecutive, and (3) specify this modified control file when you run the INQUIRY utility.

   Example: Since the original PAYREC is an indexed file, COPY and then CONTROL must be run to create a new control file that describes a consecutive data file which is identical in every other detail of its organization to PAYREC. You can then run INQUIRY on the sorted output file, specifying the new control file, the display function, and the original query: *LIST LASTN FIRSTN EMPNUM; OTHRS > 0*. This query causes all the records in the file to be displayed, since they were preselected to satisfy the relation clause. However, instead of appearing in the order illustrated in Figure 5-4, the records -- because this is their consecutive order in the file -- appear in order of overtime hours worked, from the greatest to the least number. Where overtime hours are equal, the records are listed in alphabetical order of last and first names.

The Display screen shows a maximum of 20 records at a time, but the listing is formatted into pages containing (if the fields you are listing will all fit on one line) a maximum of 54 records each. The date and heading are repeated at the top of each page. From any Display screen, you can press PF15 to send the entire listing to the printer and have it printed out in the same format.

When you are through displaying or printing the result of your query, press PF16 to return control from the DISPLAY utility to INQUIRY. The End of Query menu appears, as described in Section 5.4.3.

5.4.2 Extract Function Output

When you have specified your query and corrected any errors, INQUIRY begins the interrogation process as soon as you press PF16 from the Query Confirmation screen. If you specified the Extract function, the INQUIRY Output File Specification screen appears. Figure 5-5 shows an example of this screen.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  OUTPUT
                                                      Message Id:  1001
                                                       Component:  INQUIR

                    Information Required by INQUIRY

Please fill in the blank information for the file to be created
by this utility, and press ENTER.

     FILE    = ■■■■■■■■ in LIBRARY  = USRDATA■ on VOLUME   = VOL111
     RECORDS = 0002075

     The output file may be of the same file organization as the input file.
     If this is not necessary, processing time can be saved by creating a
        consecutive file.  To create a consecutive file enter CONSEC = YES.

                              CONSEC   = NO*
```

Figure 5-5. A Sample INQUIRY Output File Specification Screen

INQUIRY cannot create a file in which to store the extracted records until you specify a name and location for the file. That is the purpose of the Output File Specification screen. Enter the requested information in the fields as follows:

FILE -- Provide a name for the output data file.

LIBRARY -- If the OUTLIB field of your usage constants is set, the value appears as the default in this field.

VOLUME -- If the OUTVOL field of your usage constants is set, the value appears as the default in this field.

RECORDS -- The size of a new data file is set in advance by specifying, when the file is created, the maximum number of records it can contain. As a default value, INQUIRY uses the number of records in the file it is interrogating, since this is by definition the maximum number that any query can extract. However, if you know that your query will extract a significantly smaller number of records, you can save storage by modifying this value appropriately.

CONSEC -- INQUIRY ordinarily creates an output data file with the same organization as the input file. For indexed or relative data files, however, you can change the default NO in this field to YES and cause INQUIRY to create a consecutive output file. This option saves storage, since a consecutive file requires less space to create, but you should not select it if you intend further processing for the output file that requires indexed or relative file organization.

When you finish specifying the output file, press ENTER. The End of Query menu appears, as described in the next section.

5.4.3 Final INQUIRY Options

When you press PF16 to terminate the Display function or ENTER to specify the output file for the Extract function, the INQUIRY End of Query menu (Figure 5-6) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  EOJ
                                                          Message Id:  0012
                                                           Component:  INQUIR

                        Information Required by INQUIRY

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

  PFKEY        ACTION

    1          Formulate another query
    2          Save the previously completed query
    3          Create a REPORT DEFINITION FILE to be used by
               the REPORT utility
   16          Exit from INQUIRY utility
```

Figure 5-6. INQUIRY End of Query Menu

If you have just completed the Extract function, a message appears at the bottom of this menu announcing that the output file has been created, e.g., "File OUTFILE in library USRLIB on volume VOL111 created with 0000517 records." Otherwise the menu is identical for the Display and Extract functions.

You have the following options from the End of Query menu:

PF Key	Option and Description
1	**Formulate another query** -- If you press PF1, INQUIRY redisplays the Input Options screen (Figure 5-1). The last entries you made from this screen appear as default values. Whether or not you change these values in order to specify a different function or input file, the previous query remains as a default on the Query Specification screen when that screen next appears. You can accept or redefine the query before you continue. This option makes it possible to interrogate several data files in a single INQUIRY session; the defaults facilitate the formulation of related queries. If you have chosen the Extract function, you must specify an output file for each query.
2	**Save the previously completed query** -- Pressing PF2 causes INQUIRY to generate a procedure capable of reproducing the results of the query just completed. This procedure runs INQUIRY, specifies the query, and displays or extracts the query results. The next section, "Reproducing a Query," provides further information about this option.
3	**Create a REPORT DEFINITION FILE to be used by the REPORT utility** -- The REPORT utility (described in Chapter 6) retrieves data from data files by a process more elaborate and also more powerful than INQUIRY. REPORT uses a Report Definition file (RDF) to define the various options and criteria by which it selects and presents data. This INQUIRY option provides an automatic method of generating an RDF based on the query results. You can use this RDF without modification to generate a report (although it usually cannot fully duplicate the INQUIRY query results), or you can use it as the basis for a more selective or comprehensive report by modifying the default values generated by INQUIRY. The section "Creating a Report Definition File" contains more information about this option.
16	**Exit from INQUIRY utility** -- Press PF16 to terminate INQUIRY processing. You are returned to the Command Processor menu or to the program or procedure from which INQUIRY was called.

Reproducing a Query

When you select the option to save the completed query (PF2) from the End of Query menu (Figure 5-6), the INQUIRY utility creates a procedure capable of reproducing the query you have just completed. (Whether the results will be the same depends, of course, on whether or not the data file is updated in the meantime.) When the procedure is run, it runs INQUIRY, specifies the same files and options you specified from the Input Options screen, the Control File Specification screen (if any), and the Query Specification screen during the session just completed. Depending on the function, the query results are either displayed or written to an output file that has the same name and location as the one you just specified.

As soon as you press PF2 from the End of Query menu, the INQUIRY Procedure Specification screen (Figure 5-7) appears.

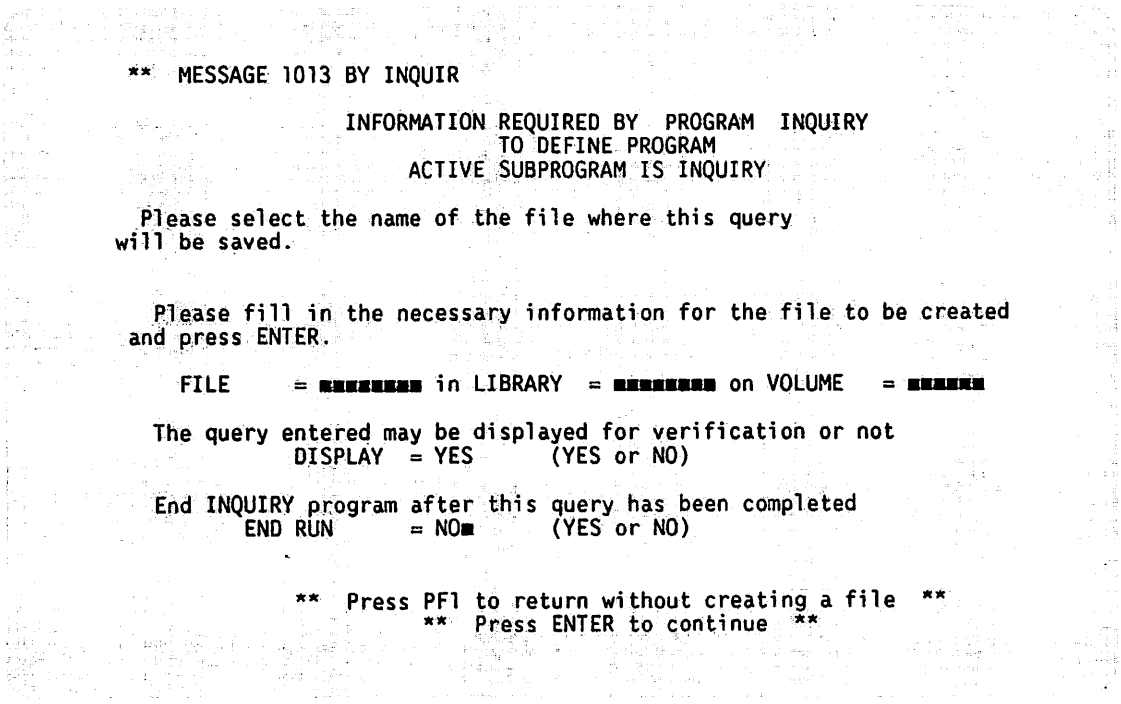


Figure 5-7. INQUIRY Procedure Specification Screen

The Procedure Specification screen solicits information needed to create the procedure. Specify the fields as follows:

FILE -- Supply a name for the procedure file.

LIBRARY -- The default value is taken from the OUTLIB field of your usage constants, if this is set.

VOLUME -- The default value is taken from the OUTVOL field of your usage constants, if this is set.

DISPLAY -- If you accept the default YES in this field, the procedure displays the Query Specification screen, which contains the saved query as the default, and INQUIRY processing continues in the standard interactive manner from that point. You can, at your option, modify the query before you proceed. If you change the value to NO, the saved query is accepted automatically and used to interrogate the data file; the query result is then either displayed or written to a file depending on the function.

END RUN -- If you accept the default NO in this field, the End of Query screen appears when INQUIRY processing is completed. (For the Extract function, this point comes when the output file is created; for the Display function, it comes when PF16 is pressed to end the display of the query results.

Enter or change values in these fields as appropriate and press ENTER. The End of Query menu reappears with a message indicating that the procedure file has been created (e.g., "File FOOPROC in library USRPROC on volume VOL111" created"). If you decide instead to return to the End of Query menu without generating a procedure, press PF1.

To run the procedure, enter the file, library, and volume names from the Run screen of the VS Command Processor, or in a RUN statement in another procedure.

If you have accepted the default value for either the DISPLAY or the END RUN field, INQUIRY processing will require some interaction. If you intend a procedure to be run in background mode (as may be appropriate for the extract function), DISPLAY must be set to NO and END RUN to YES.

Note: *If a procedure repeats a query that extracts records to an output file, the procedure specifies the same output file name and location as the query. If an existing file has the same specifications, the user is prompted to choose between pressing PF3 to scratch the original file, or supplying different specifications for the new file. If you intend such a procedure to run without interaction, therefore, you must either scratch the original output file or change its specifications before the procedure is run.*

Creating a Report Definition File

When you select the option to create a Report Definition file (PF3) from the End of Query menu, INQUIRY creates a file capable of reproducing certain results of the query just completed through the REPORT utility (described in Chapter 6). REPORT uses a Report Definition file (RDF) to define the various options and criteria by which it selects and presents data. You can use an RDF created through this INQUIRY option to produce a report, but in most cases it cannot fully duplicate the results of the INQUIRY query.

REPORT both selects and sorts records, but its means of selection are not compatible with INQUIRY's. It cannot use an RDF generated by INQUIRY, therefore, to select the same set of records INQUIRY selected in the query on which the RDF is based. If the query involved the Display function, the RDF produces a report that contains the same fields as the query (i.e., every field mentioned in the list and relation clauses), but it includes *all* the records in the data file, being unable to select. If the query involved the Extract function, the RDF produces a report that essentially prints out the whole data file it is interrogating, since the report includes all fields from every record in the file.

The principal use of an INQUIRY-generated RDF, however, is not to produce reports directly, but to serve as the basis for a modified RDF that takes advantage of the REPORT utility's power. By generating an RDF based on an INQUIRY query, and later specifying it when you run REPORT, you can save steps and time in the report generation process described in Chapter 6. You can use the methods explained there to restore and extend the selectivity of the query, as well as to improve the presentation of the data in several ways.

When you press PF3 from the End Query screen, the RDF Specification screen (not shown) appears. This screen has the usual three fields in which to enter specifications for the Report Definition file:

FILE -- Supply a name for the Report Definition file.

LIBRARY -- INQUIRY supplies a default value for this field by concatenating your user ID with the string *RPT*, e.g., USRRPT. Since REPORT automatically seeks RDFs in a library with this name, it makes sense to accept the default unless there is a compelling reason not to do so.

VOLUME -- Default value is taken from the OUTVOL field of your usage constants, if this is set.

Enter or change the specifications in these fields as appropriate, and press ENTER. The End of Query screen reappears with a message indicating that the RDF has been created (e.g., "File FOORDF in library USRRPT on volume VOL111 created"). If you decide instead to return to the End of Query menu without generating an RDF, press PF1.

CHAPTER 6
REPORT

6.1 INTRODUCTION

The REPORT utility provides a method of creating, modifying, and printing reports on data files. Designed especially for nontechnical users, REPORT features a step-by-step approach that leads a user through the process of defining a report. REPORT also allows modification of the report definition. The Modify option uses, to a large extent, the same displays as the report definition phase and makes it easy for the operator to modify an existing report.

In addition to producing formal reports, this utility also enables the user to obtain listings of data that fall within specified limits, for example, a listing of the salesmen selling more than $10,000 during a specified month. INQUIRY, which is discussed in Chapter 5, provides a more elaborate way of extracting data based on user-specified criteria.

The process of producing a report involves the following steps, documented in the indicated sections.

- The report format and content must be defined. For details, refer to Section 6.2.
- An existing report format and content can be modified. For details, refer to Section 6.3.
- Once the format and content have been defined, the report is printed. The steps involved in printing a report are discussed in Section 6.4.
- The report can be further customized through a User Exit subroutine, discussed in Section 6.5.
- The amount of workstation interaction involved in producing a report can be reduced through the VS Procedure Language, as described in Section 6.6.

An overview of REPORT processing is provided in Figure 6-1.

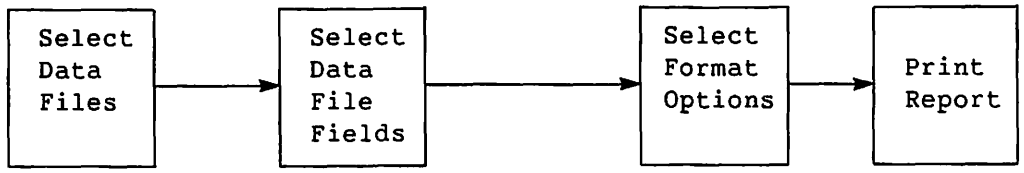


Figure 6-1. REPORT Processing

REPORT can be invoked directly from the Command Processor or from the CONTROL File Utility menu. The first REPORT screen, referred to as the REPORT Utility menu, offers the following options:

PF Key	Option
2	Create a Report Definition
3	Modify Report Attributes
4	Print a Report
16	End Job

6.2 CREATING A REPORT DEFINITION FILE

The first step in producing a report is to create a Report Definition file (RDF). Creating the RDF can be simple or complex, depending on the requirements of the user. Almost all screens have default field values to be used if the user does not wish to modify them. REPORT creates the RDF by combining the default values with the values entered by the operator. Once an RDF has been created, the report can be run repeatedly by specifying its Report ID in the utility's print option.

6.2.1 Report Definition File Creation Options

The Report Definition file can be defined interactively when PF2 from the REPORT Utility menu is pressed. The Creation Options screen is displayed when the RDF Creation option is selected. Through the Creation Options screen, the user names the RDF and selects additional REPORT functions.

The Report ID is the name used by REPORT to identify the RDF. The value specified for the Report ID is the VS file name; a default library name is constructed for the RDF by concatenating the user ID with RPT. If a default OUTVOL has been set by the user through the Command Processor or a procedure, the RDF file parameters are the Report ID, default library, and default volume names. If no default OUTVOL has been specified, the user is asked to select an output volume following the field selection phase of report processing. (Refer to Section 6.2.3.) The Report ID and default library name can be changed when the output volume is specified. Wherever possible, the Report Definition file should have the same name as its associated data and control files.

The Creation Options screen also allows the user to select one or two REPORT options. The user can choose to report on more than one data file to invoke a User Exit program for additional report format control. The default selection response for each option is NO, so that the user need not specify a response for any option unless a response is desired. If a secondary DMS file is to be used in the report, the file *must* be indexed because it is chained to the primary file through its primary key field. User Exit program processing is described in Section 6.5.

6.2.2 Data File Selection

The user must specify the name of the primary data file on the Primary Data File Selection screen. The primary data file on which the report is to be based is specified by providing its file name and, optionally, the library and volume names. If the library and/or volume are not specified, their names are obtained when the report is printed, from either user input or system defaults specified by the user. The primary data file is the only file on which a sort can be performed.

REPORT assumes that the data file has a corresponding control file with the same file name in the default control file library. Unless the default INVOL has been set, the user must specify the volume location of the control file and/or supply the actual file and library names of the control file if they differ from the default values. If a second data file is to be used for the report, the file, library, and volume names are requested following primary data file field selection. (Refer to Section 6.2.3, Field Selection.) On the Secondary Data File Selection screen, the user must provide the name of the field in the primary file to be used for chaining the primary file. REPORT assumes that the secondary data file has an associated control file with the same file name; the control file parameters must be specified for the secondary data file just as for the primary data file. The user can then select fields from the secondary data file, as described in Section 6.2.3.

6.2.3 Field Selection

The user must select the fields needed for report preparation from the list of all the control file's fields presented on the Primary File Field Selection screen. Fields used merely for record selection or sorting purposes should be selected as well as fields to be printed in the report. It may be useful to select extra fields in anticipation of future requirements. (These extra fields can be assigned a 99 sequence number and need not be used to report production. Refer to Field Sequence, in Section 6.2.4, for additional information.)

If a second file is to be used for the report, its fields are presented for user selection on the Secondary File Field Selection screen following secondary data file selection. The fields are selected by the same process as the fields in the primary file.

Following field selection for the primary and/or secondary files, the user can specify new fields to be used in the report. New fields comprise either existing numeric fields or constants whose values are added, subtracted, multiplied, or divided with other numeric fields and/or constants, or existing character fields or literals whose values are concatenated with other character fields and/or literal values. The arithmetic operations used in creating new fields are performed in the order in which they are specified. If new fields are desired, they are defined on subsequent screens by giving the field IDs, field types, lengths (i.e., the number of characters), decimal positions (if field type is numeric), source fields, and the arithmetic operations to be performed.

The sum of the primary, secondary, and new fields selected cannot exceed 80. The total width of all fields appearing on any one print line in the report cannot exceed 132 bytes; the total width for all print lines cannot exceed 396 bytes. The width of a field is determined by the width of its header or the width of its contents, whichever is greater.

6.2.4 Report Definition Options

After all fields have been selected for the report, 13 options become available and are displayed on the Report Definition Options menu. These options determine the report format and are described in detail in this section. Pressing ENTER from the Report Definition Options menu enables the user to proceed sequentially through each option screen, specifying the desired format for each option. Default values are provided for all options, so that in many cases the user can quickly define the report format. After each screen has been filled, pressing ENTER displays the next option in sequence, through the PF11 option. The functions associated with PF keys 12 and 16 are available only from the Report Definition Options menu.

PF Key	Action	Default Action
1	Display the Report Definition Menu	None
2	Report Title Information	Report ID
3	Column Headings	Field IDs
4	Spacing Between Fields	2 spaces
5	Field Sequence on Report	Alphabetic order
6	External Field Size	Per control file
7	Edit Options	Per control file
8	Data Limits for Record Selection	None
9	Sort Fields	None
10	Control Fields	None
11	Report Summary Options	None
12	Print Headings and Dummy Detail Lines	
16	Exit Report Definition Phase	

The fields selected may result in a print line exceeding 132 characters. In that event, a subset of the options is provided, so that the user can reduce the number of characters in that print line. The user cannot proceed with report definition until the print line size is diminished by (1) reducing the external field size, the spacing between fields, or the length of one or more column headings; (2) redefining which fields are to be included in the report; and (3) excluding one or more fields from the report or respecifying the lines on which the fields are to appear through the Field Sequence option.

Report Title and Headings Information

Three types of title may be specified for a report: report title, page heading, and/or control break descriptors.

Report Title -- The report title may be up to three lines long, is required, and always appears on the first page.

Page Headings -- If page headings are selected, they appear on all pages except the first page. If page headings are not selected, the report title appears on all pages. The report date (selected at report production time) and page numbers appear at the top of every page.

Control Break Descriptors -- A control break descriptor is a user-supplied subtitle appearing within the body of a report whenever the value of a control field changes (e.g., at each control break). Control break descriptors appear on the same line as the associated total and in the selected column. (The default is column 1.) (Refer to Control Fields and Report Summary Options, within this section.)

Column Headings

The field ID (field name) is used as the fields column heading unless a different heading is provided. The space allocated for the field on the report is the column heading length or external field size, whichever is greater. The column heading can be blank. User-selected column subheadings may also be defined at this time.

Note: *Lines for column headings for print lines 2 and 3 are reserved, even if no column heading for fields appearing on these lines is specified. If column headings for fields on print lines 2 and 3 are specified, these headings are printed above their corresponding fields, on their respective lines.*

Spaces Before Fields

Two spaces normally are placed before each field on the report. The user may change the number of spaces preceding the first character of each field and its column heading. If the total number of spaces specified for any print line exceeds 132, the Print Line Reduction screen appears.

Field Sequence

With the Field Sequence option, the user can specify both the line number (1 - 3) and the sequence number (1 - 80) for each field to be printed in the report. Fields given a sequence number of 99 do not appear on the report, but may be used in the preparation of the report. For example, if it is desirable to sort data by the field DEPT but not to print the values for DEPT, DEPT is given a sequence number of 99. The field names are presented to the user in sorted (i.e., alphabetic or numeric) order, with all fields assigned to line 1; unless modified, these values are the defaults. The user can modify both the line numbers and sequence numbers by typing over the displayed values.

When determining the sequence of a large number of fields, skipping numbers is useful (e.g., numbering by 5s). Skipping numbers allows fields to be easily inserted between two formerly adjoining fields. Also, by skipping numbers, more than one field may be given the same sequence number. In this case, the first field of a given sequence number is placed before subsequent fields of the same sequence number, etc. The fields then are automatically renumbered in increments of 1 and rearranged by sequence number within line number when ENTER is pressed. This duplicate numbering option facilitates ordering of large numbers of fields.

External Field Size

The External Field Size option is used to specify the number of character positions each field is to occupy on the report. The default size indicated is the number of character positions specified in CONTROL. Fields can be increased in size by padding with blank characters, or they can be truncated. Character fields are left-justified and are truncated from the right, while numeric fields are right-justified and are truncated from the left. Adding dollar signs, commas, or other edit characters through the data edit options increases the size of the field, and must be considered by the user to avoid unwanted truncation. (Refer to Data Edit Options in this Section.) The external field size may not be 0.

There are three primary reasons to alter the external field sizes for a report.

Reduced Size -- To delete undesired or irrelevant data by truncation

Increased Size -- To allow space for a field that has been expanded due to special character insertion (", ", "$", "/", "-")

Total Values -- To allow space for a numeric field's total value (created through the Report Summary options) when the total value exceeds the field's length

Data File Options

The user has the option of altering the data format for any numeric field (packed, zoned, unsigned, or binary if the Binary Edit Code = 1) scheduled to appear on the report. The Data Edit Options screen presents a "picture" of each field, and requests the user to specify if any given field is to be modified. If modification is chosen, an edit screen appears, requesting the user to select which modifications are to be performed. The current format is displayed as the default. Edit options for numeric fields, with explanations of the entry codes, are listed as follows. Entry codes can also be displayed by pressing PF14.

- Suppress zeroes

  ZN -- Zero-suppress leading 0s except for rightmost N positions.
  **N* -- Asterisk-protect leading 0s except for rightmost N positions.
  Blank -- No zero suppression.

- Sign control

  CR -- Print trailing "CR" if the field is negative.

  DB -- Print trailing "DB" if the field is negative.

  9- -- Print 1 trailing blank if the field is positive, trailing "-" if negative.

  Blank -- Print no signs.

- Decimal carry

  .N -- Print out N characters to the right of the decimal point.
  Blank -- Print all decimal positions, but no decimal point.

- Special insertion characters

  *N** -- Insert an "*" to the right of the Nth character (leftmost = 1).

  N- -- Insert a "-" to the right of the Nth character.

  N, -- Insert a "," to the right of the Nth character.

  N/ -- Insert a "/" to the right of the Nth character.

  N. -- Insert a "." to the right of the Nth character.

  Blank -- No special characters.

Note: *The character designated as N cannot be 0. Also, a leading dollar sign is not included in determining N.*

- Dollar sign

  $9 -- Insert a "$" in the leftmost field position.
  $$ -- Float the dollar sign.
  Blank -- No dollar sign.

- Commas

  Y -- Insert a comma after every third integer from the decimal point.

  N -- No comma insertion.

Data Limits

The Data Limits option allows the user to select records for printing by defining limits on the values of up to ten fields within the records. The legal operators are as follows:

- GT Greater than
- GE Greater than or equal to
- LT Less than
- LE Less than or equal to
- EQ Equal to
- NE Not equal to

Up to ten of these conditions can be specified for each field, connected with AND/OR operators. The AND operators are evaluated first. Consequently, a set of limits is defined for record selection for use in report printing. If two or more conditions are connected with AND, the record is selected only if both (or all) conditions hold true. If the OR connector is used, the record is selected if at least one of the conditions is true.

For example, suppose each record consists of the following fields: employee's name (NAME), social security number (IDNUMBER), department (DEPT), hours worked during preceding week (HOURS), and hourly wage rate (WRATE). The users desire a separate report on all employees who worked fewer than 40 hours and whose wage rates exceed $4.00. They therefore define the limit on the HOURS field as LT 40, the limit on WRATE as GT $4.00, and specify AND as the connector. Any record with values outside these limits is not printed.

File Sort

The File Sort option allows the user to define the order of records to be printed on the report. Up to eight fields from the primary file can be sorted in either ascending or descending order.

The user assigns the highest level to the primary field on which the sort is based. The next level would be assigned to the secondary field. For example, if the file is first sorted, according to department, DEPT will be assigned Level 1. If, within each department, the file is to be sorted alphabetically according to the employee's family names, NAME will be assigned to Level 2.

Control Fields

Control fields govern the totalling and printing of subtotals at report time, and provide the capacity to include breaks within the report with either spacing or page ejections. Page ejection facilitates sending selected portions of a report to different departments. The specified break occurs whenever the value of a control field changes.

In the printed report, a subtotal line for all fields selected for total values (through the Report Summary Options screen) is printed following the printed field values of the control field section. If a control field descriptor is specified, it also appears on this subtotal line. Note that careful placement of the descriptor through the Spaces Before Fields screen is necessary to avoid overprinting of descriptors and data. If no totals are specified in the Report Summary Options screen, this line displays only the descriptor, if selected, or nothing if no descriptor and no totals are specified.

Each control field must be assigned a level number. The level number is used to associate each total line (if specified through the Report Summary Options screen, or blank line if not), with a control break descriptor (if specified through the Report Title Information screen, or blank if not). Level 1 control breaks occur most frequently, and are associated with Level 1 control break descriptors.

Report Summary Options

Report Summary options are as follows:

- TOTAL
- MAXIMUM
- MINIMUM
- AVERAGE

The Report Summary information is provided at the end of the report for the values contained in each numeric field selected. If control fields have been defined, subtotals are provided at control breaks for all numeric fields (if any) for which totals are requested. (Refer to Control Fields in this section.)

Print Headings and Dummy Detail Lines

The Print Headings and Dummy Detail Lines option prints out a simulated run of the report, including the title lines, heading lines and one detail line. The detail line uses Xs for character fields, 9s for numeric fields, Fs for binary hexadecimal fields, plus any editing character selected (such as $) to indicate the formats and positions of the fields.

6.3 MODIFYING A REPORT DEFINITION FILE

The Report Modification option of the REPORT utility is invoked by selecting PF3 from the REPORT Utility menu. The Report Definition file (RDF) to be modified must first be identified. If the RDF does not reside in the default library and/or if the user has no default INVOL, REPORT requests the file, library, and volume names of the RDF immediately after receiving the Report ID. The control and data files to be used in the report can be respecified on a screen that appears when PF8 is pressed from the Report Modification screen. In this way, the user can change the data and/or control files used in preparing the report. In addition, the User Exit option can be added or deleted on the same screen as control and data file respecification. If the user specifies the file, library, and volume names of a User Exit subroutine, the report format is controlled by the specified User Exit subroutine. Similarly, the User Exit option can be deleted from the RDF by erasing the values specified for file, library, and volume names.

The next screen enables additional primary data file, secondary data file, or new fields to be included in the report if the user overrides the default NO response. If new fields are defined, the user first defines the field name, data type, and length, and then defines the new field's components by choosing to modify them. All added fields are given a 99 sequence number by default; this value may be changed at a later stage of report modification. All options available on the Report Definition Options menu then become available for report modification, and are discussed in the following sections. The only restriction on modification is that the data fields previously specified for the report may not be changed.

6.3.1 Report Title Information

Report titles, page headings, and control break descriptors may be modified by the user. Refer to "Report Title and Headings Information" in Section 6.2.4 for additional information.

6.3.2 Column Headings

Column headings and subheadings (maximum length 25 characters) may be modified by the user for each field. Refer to Column Headings in Section 6.2.4 for additional information.

6.3.3 Spacing Before Fields

The spacing before fields on the report may be modified by the user. For additional information, refer to "Spaces Before Fields" in Section 6.2.4.

6.3.4 Field Sequence on Report

The user can modify the sequence and line numbers of the fields on the report. A field also can be given a sequence number of 99, which means that the field can be used in report preparation (e.g., for data limit purposes), but the field does not appear on the report. For additional information, refer to "Field Sequence" in Section 6.2.4.

6.3.5 External Field Size

The user can alter the external field size of any field to add blanks to the field or to truncate part of the field. For additional information, refer to "External Field Size" in Section 6.2.4.

6.3.6 Edit Options

The user can alter the data edit format options selected in the report definition. For additional information, refer to "Data Edit Options" in Section 6.2.4.

6.3.7 Data Limits for Record Selection

The user can alter or delete the limits to be imposed on a field to determine which records should be printed on the report. Up to 10 fields may be defined with limits, and up to 20 AND/OR connectors and operands may be used per field with each set of limits. For additional information, refer to "Data Limits" in Section 6.2.4.

6.3.8 Sort Fields

The sort fields selected at the report definitions stage can be altered by the user. The records to be used in the report can be sorted on up to eight fields, with any combination of ascending or descending order on the individual fields. However, the fields to be sorted must all reside in the primary file. For additional information, refer to "File Sort" in Section 6.2.4.

6.3.9 Control Fields

The control fields selected during the report definition stage may be altered. Up to five levels of control fields may be selected for the report. For additional information, refer to "Control Fields" in Section 6.2.4.

6.3.10 Report Summaries

Summary information to be printed at the end of the report may be altered by the user. Summary information can be printed for any and all numeric fields in the report and includes totals, maximum values, minimum values, and averages. For details, refer to "Report Summary Options" in Section 6.2.4.

6.3.11 Print Headings and Dummy Detail Lines

This option enables the user to print the title lines, heading lines, and a detail line of the defined report. In this way, the user can review any modifications made through the Report Modification option of the REPORT utility. Refer to Section 6.2.4 for further information.

6.4 PRINTING A REPORT

When PF4 is pressed from the REPORT Utility menu, the Print Report screen is displayed. From the Print Report screen the user can specify certain parameters for the printing of previously defined reports. These parameters include the report data, output device, data file changes, record count, lines per page, print lines to be printed, and the order of the print lines. If the library and volume names of the primary and/or secondary data files have not been previously specified, their values must be supplied on the screen that appears following the RDF definition. The library and volume names can also be given immediately following the Print Report screen, if no additional file parameters were required for the RDF. The Print Report options are discussed in detail in the following subsections.

6.4.1 Report ID

The Report ID field specifies the name of the Report Definition file to be used in printing the report. The next screen requests the file parameters of the RDF if the default values are incorrect or if the default INVOL has not been specified.

6.4.2 Report Date

The selected report date appears on the top of every page. The default for the report date is the current date.

6.4.3 Output Device

The user can choose either to display the report on the screen (DISPLAY) or to print it (PRINTER). The default is PRINTER.

6.4.4 Change Data Files

The user can elect to change the data file(s) to be used for the report. The response is YES or NO; the default is NO. The affirmative response produces a screen for data file respecification.

6.4.5 Count Option

The Count option allows the user to select a number of records to be used in the report. This option can be used either to give a count of the records processed or to limit the number of records to be listed (as in a "TOP TEN" report).

6.4.6 Sum Only

The Sum Only option provides the capability to obtain a report of total information (i.e., no detail lines). Only control break and total lines are printed. Totals appear only if they were specified in the report file specifications.

6.4.7 Lines Per Page

The Lines per Page option permits the user to specify any number from 05 to 99 as the maximum number of lines printed per page.

6.4.8 Select Lines

The Select Lines option allows the user to specify the order of the three print lines (and their corresponding column heading lines, if any) by specifying the digits 1, 2, and 3 in the desired order. Since a user can enter a leading or trailing space in place of a number, the user also selects which print lines are printed. Two examples of the Select Lines option are as follows:

```
Select Lines = 123      Select Lines = 32
prints as:              prints as:

LINE 1
                        LINE 3
LINE 2                  LINE 2
LINE 3
```

6.4.9 Print Line Spacing

The Print Line Spacing option permits specification of the number of blank lines following each print line, in the order specified in the Select Lines option. Up to 5 blank lines can be specified after each print line. Print line spacing is illustrated as follows:

```
Select Lines = 213      Select Lines = 123 (only 1 data line specified)
Print Line              Print Line
Spacing      = 031      Spacing      = 000

LINE 2                  LINE 1(A)
LINE 1                  LINE 1(B)
blank line              LINE 1(C)
blank line
blank line
LINE 3
blank line
```

6.5 INVOKING A USER EXIT SUBROUTINE

The report format and content defined in REPORT can be overridden by the user on a record-by-record basis through a User Exit subroutine. Through a User Exit subroutine, the user can omit a line from the report, print additional lines, or alter the contents of a line constructed by REPORT. Invoking a User Exit subroutine involves the following steps:

- The Report Definition file must indicate User Exit processing and contain the file parameters of the appropriate subroutine. Refer to Section 6.5.1 for details.
- A compiled or assembled User Exit subroutine must be constructed. Refer to Section 6.5.2 for details.
- The report is printed with the subroutine manipulating the format and content. Refer to Section 6.5.3 for details.

6.5.1 Report Definition File Requirements

For the User Exit subroutine to function, the RDF must have the User Exit option specified and must contain the file parameters of the User Exit subroutine object code. If an RDF is created with the anticipation of User Exit processing, the User Exit option on the Creation Options screen is set to YES. REPORT then requests the User Exit subroutine object code file parameters following field selection. If an existing RDF is modified to support User Exit processing, the User Exit option is specified during Report Modification. To specify the User Exit option, press PF8 from the Report ID Specification screen and supply the file parameters of the User Exit program on the resulting screen. User Exit processing can be removed from an RDF by deleting the User Exit file name during Report Modification.

Note: *A Report Definition file constructed for User Exit processing cannot be used to print a report without the User Exit. If the user wishes to print the report with and without User Exit manipulations, a separate, additional RDF, which does not contain the User Exit file parameters, should also be constructed.*

6.5.2 Constructing the User Exit Subroutine

REPORT supports User Exit subroutines written in the VS Assembler, BASIC, COBOL, and FORTRAN languages. The program must conform to the specific language requirements for an external subroutine, as described in the *VS Assembler Language Reference*, *VS BASIC Language Reference*, VS COBOL Reference, or the *VS FORTRAN Language Reference*.

The User Exit subroutine does not directly manage the report print file, but controls the printing of each line through the arguments passed to REPORT. The subroutine can alter both the report format and content. Some arguments are passed to the subroutine only for informational purposes; other arguments control the print line when returned to REPORT. These arguments may have any name in the subroutine, provided that they are returned in the order required by REPORT. The arguments passed to and returned from the User Exit subroutine are summarized as follows, in the order in which they must appear in an argument list:

Argument	Length	Description
Line	4 bytes	Each byte in the argument provides a particular description of the line about to be printed. Versions 3.2.2 and later of REPORT access all four bytes; earlier versions access only byte 1. User Exit subroutines written for earlier versions of REPORT are compatible with later versions. Each byte and its meaning are summarized as follows; each byte contains a character value:

Byte	Description
1	Indicates the type of the line about to be printed. The line type is supplied to the subroutine by REPORT. Any changes made to the line type within the subroutine are ignored by REPORT. Byte 1 has one of the following character values indicating a particular line type.

Value	Line Type
P	Page heading or report title.
C	Column heading or subheading.
D	Detail line containing information from the data file record. Formatted according to RDF specifications.

Argument	Length	Description
Line (continued)	4 bytes	(continued)

Byte	Description
	Value **Line Type**
	1 - 5 Control break line. The number indicates the control break level encountered as defined on the Control Fields Specification screen.
	T Control break totals line at the end of the report.
	S Report summary option line, including summary option heading.
	E Print line does not originate from REPORT. The E line type indicates that all print lines from REPORT have already been printed. The E line type can arise only if the user has set byte 4 of this argument to 1.
2	Indicates which line number, of a maximum of three, is being printed for the current record. In a report where only one line is printed per record, the value of this field is 1. In a report with more than one print line per record, the value can be 1, 2, or 3. When the line type is a report title or page heading (P), this field does not contain a meaningful value. The value indicating the multiple line number is passed to the subroutine by REPORT; any changes made to the multiple line number value by the User Exit subroutine are ignored by REPORT.

Argument	Length	Description
Line (continued)	4 bytes	(continued)

Byte	Description
3	Indicates the end of data from the data file(s). The end of data field has a value of 0 until the last record in the input file has been read. A value of 1 indicates that the last data file record has been read. The end of data field value is passed to the subroutine by REPORT; REPORT ignores the value returned by the subroutine.
4	Indicates whether or not control should pass to the User Exit subroutine after the last report line has been printed. The default value of 0 returns control to REPORT; if the byte is set to 1 by the subroutine, control returns to the subroutine after each print line until the subroutine resets the byte to 0. If the byte is set to 1, the User Exit subroutine can print additional information at the *end* of the report. Unless this byte is set to 1, there is no way of inserting additional information or comments at the end of a report, as control automatically returns to REPORT when the last line has been printed. The Print Switch value (Argument 3 in the argument list) continues to influence whether or not page or column headings are printed when a top-of-form is reached. A Print Switch value of P prints the headings; a value of R suppresses them. The User Exit subroutine defines a line to be printed and then sends it to REPORT for printing. Processing continues on a line-by-line basis until the subroutine resets the byte value to 0.

Argument	Length	Description
Print Line Count	2 bytes	Indicates the number of lines currently on the page, including the current line if it has been generated by the REPORT utility. This byte controls top-of-form processing. Thus, the User Exit subroutine *must* update the print line count whenever lines are added, deleted, or affected by a change in the control character of the print line. Failure to update the print line count results in irregular top-of-form processing. The argument data type is BINARY. Because BASIC has 4-byte integers, BASIC programs must ensure compatibility.
Print Switch	1 byte	Indicates the action REPORT should take with the print line. The value is set by the subroutine; each of the following values has the indicated meaning. REPORT sets a default value of P. The Print Switch argument has the CHARACTER data type.

Value	Line Type
O	Omit printing the line.
P	Print line and continue with report.
R	Print line and return to the subroutine before processing the next print line. In this way, the user can insert comments or additional information in the middle of the report. The user must reset the Print Switch to P when printing the last of the inserted lines.

Print	134 bytes	Contains the 134-character current print line. The first two print line characters are the print control characters, described in the *VS Principles of Operation*. The User Exit subroutine can affect the line content by changing the value of this argument. In addition, blank lines can be inserted through the print control characters. If blank lines are inserted, the print line count must be updated.

Argument	Length	Description
Primary Record Area	Record length of primary data file	Contains the current record from the primary data file or view. Any fields on which reporting is not allowed contain blanks. This argument is passed to the subroutine by REPORT; changes made to the record by the subroutine do not affect either the record in the data file or the record viewed by REPORT. The field definition and record length in the subroutine should be consistent with those of the primary data file. In COBOL programs, the record description can be obtained from the Copylib function of CONTROL. (Refer to Chapter 2, CONTROL.) Because this argument corresponds to the primary data file rather than the report print file, the value of this argument does not necessarily change with every call to the User Exit subroutine. The value can be identical to that of a previous call if a multiple print line was defined in the RDF or if the current print line does not involve the data file, such as page or column heading lines or lines inserted into the report by the User Exit subroutine. This argument allows the user to use report field values in calculations.
Secondary Record Area	Record length of secondary data file	Contains the current record from the secondary data file or view if the report is based on two data files. This argument is passed to the subroutine by REPORT only if the RDF specifies a second data file. The argument description is identical to that of the primary record area in all other respects.

A sample COBOL User Exit subroutine follows that illustrates the use of the subroutine arguments to alter the format and content of a report. The subroutine monitors the print file generated by REPORT that contains the department, name, address, hourly rate, and number of hours worked for an employee data file. The User Exit subroutine omits entries for employees not in department 50, prints a message on the print lines corresponding to those entries with an hourly rate exceeding $20, and prints a message at the end of the report.

```
000100    IDENTIFICATION DIVISION.
000200    PROGRAM-ID. FIXRPT.
000300*
000400*     THIS IS A SAMPLE USER EXIT PROGRAM FOR THE REPORT UTILITY.
000500*     THIS USEREXIT MODIFIES THE PRINTED REPORT IN THE FOLLOWING
000600*     MANNER.
000700*       IF A PERSON IN THE PRINTED REPORT IS NOT IN DEPARTMENT 50
000800*         THEN THE PRINTLINE IS OMITTED.
000900*       IF A PERSON IN DEPARTMENT 50 EARNS MORE THAN $20 PER HOUR,
001000*         THEN A MESSAGE IS PLACED ON THE PRINT LINE BEFORE IT
001100*         IS PRINTED.
001110*       AT THE END OF THE REPORT, THE USER EXIT PROGRAM ADDS A FEW
001120*         ADDITIONAL LINES OF TEXT.
001200*
001300    ENVIRONMENT DIVISION.
001400    CONFIGURATION SECTION.
001500    SOURCE-COMPUTER. VS2200.
001600    OBJECT-COMPUTER. VS2200.
001700*
001800    DATA DIVISION.
001900    WORKING-STORAGE SECTION.
002000    77  FIRST-TIME-INDICATOR PIC X VALUE "0".
002100        88  FIRST-TIME VALUE "0".
002200    77  NOT-FIRST-TIME PIC X VALUE "1".
002300    77  END-LINE-COUNT PIC 99.
002400*
002500    LINKAGE SECTION.
002600    01  LINE-INFO.
002700        02  LINE-TYPE   PIC X.
002800        02  LINE-NUMBER PIC X.
003000        02  END-OF-DATA PIC X.
003100            88  MORE-DATA VALUE "0".
03150     02  RETURN-AFTER-LAST PIC X.
03200 01  PRINT-LINE-COUNT USAGE IS BINARY
03300 01  PRINT-SW PIC X.
03400 01  PRINT-LINE.
003500        02  PRINT-LINE-CC USAGE IS BINARY.
003600        02  PRINT-LINE-TEXT.
003700            03  FILLER PIC X(97).
003800            03  PRINT-LINE-MESSAGE-AREA PIC X(25).
003900            03  FILLER PIC X(10).
004000    01  RECORD-AREA.
004100        02  DEPARTMENT   PIC 99.
004200        02  NAME         PIC X(20).
004300        02  ADDRESS      PIC X(25).
004400        02  HOURLY-RATE  PIC 99V99.
004500        02  HOURS-WORKED PIC 99.
004600*
004700    PROCEDURE DIVISION USING LINE-INFO, PRINT-LINE COUNT, PRINT-SW,
004800        PRINT-LINE, RECORD-AREA.
004900*
```

```
005000     PROCESS-A-PRINT-LINE.
005100         IF FIRST-TIME THEN PERFORM FIRST-TIME-PROCESSING.
005200         IF MORE-DATA AND LINE-TYPE = "D" THEN
005300             IF DEPARTMENT NOT - 50
005400                 THEN PERFORM OMIT-THE-LINE,
005500             ELSE IF HOURLY-RATE > 20.00
005600                 THEN PERFORM CHANGE-THE-PRINT-LINE.
005700         IF LINE-TYPE = "E" THEN PERFORM END-THE-REPORT.
005800         EXIT PROGRAM.
005900*
006000     FIRST-TIME-PROCESSING.
006100         MOVE NOT-FIRST-TIME TO FIRST-TIME-INDICATOR.
006200         MOVE "1" TO RETURN-AFTER-LAST.
006300         MOVE 1 TO END-LINE-COUNT.
006400*
006500     OMIT-THE-LINE.
006600         MOVE "0" TO PRINT-SW.
006700         SUBTRACT PRINT-LINE-CC FROM PRINT-LINE-COUNT.
006800*
006900     CHANGE-THE-PRINT-LINE.
007000         MOVE "THIS EMPLOYEE IS OVERPAID" TO PRINT-LINE-MESSAGE-AREA.
007100*
007200     END-THE-REPORT.
007300         IF END-LINE-COUNT = 1 THEN
007400             MOVE "THIS IS THE END OF THE REPORT" TO PRINT-LINE-TEXT,
007500             MOVE "R" TO PRINT-SW.
007600         ELSE
007700             MOVE "****************************" TO PRINT-LINE-TEXT,
007800             MOVE "P" TO PRINT-SW.
007900             MOVE "0" TO RETURN-AFTER-LAST.
```

6.5.3 REPORT Processing With a User Exit Subroutine

Once the RDF has been supplied with the file parameters of an existing compiled or assembled User Exit subroutine, the report can be printed through PF4 of the REPORT Utility menu. The report is printed according to the report format and content as modified by the subroutine. It is not necessary to link the User Exit subroutine to REPORT through the LINKER utility; REPORT automatically links to the subroutine without user interaction.

When the report is printed, the REPORT utility constructs each print line as defined in the RDF and passes the print line (along with the other arguments described in Section 6.5.2, Constructing the User Exit Subroutine), to the User Exit subroutine for further processing. After the subroutine returns the optionally modified print line, REPORT prints the line and constructs the next print line. All printing is carried out through the REPORT utility. When the last line has been placed in the print file, processing terminates and the print file is available. The immediate disposition of the print file is dependent on the user's print mode defaults. Refer to the *VS Programmer's Introduction* for a discussion of print mode defaults and the print queue.

6.6 A SAMPLE REPORT PROCEDURE

Although report definition and modification are necessarily interactive processes, REPORT function selection, report printing, and much of the file specification in the definition and modification phases can be automated through the VS Procedure Language. A complete list of REPORT GETPARMs is given in Appendix A, File Management Utility GETPARMs; consult the *VS Procedure Language Reference* for details concerning procedure syntax.

For report definition and modification, REPORT does not request file parameters through GETPARMs unless default values are incorrect or nonexistent. However, since only the file names of the primary and secondary data files need be specified, the library and volume locations can be specified at print time through a procedure. The volume locations and any altering of default values for the data files' corresponding control files must be specified following file name specification. The specification of the volume name and alteration of any default value for the RDF can be supplied through a procedure immediately following field definition for report creation and immediately following report ID specification for report printing and modification.

The following procedure initiates report creation and prints the report. The procedure supplies the volume locations for the primary and secondary control files and overrides the default library for the secondary control file; supplies the volume name of the RDF whenever required (accepting the REPORT default library); and specifies the library and volume names of the primary and secondary data files. The user need only specify the file names of the primary and secondary data files, define the fields for the report, and supply the appropriate report definition options.

```
PROCEDURE
RUN REPORT
ENTER FUNCTION 2
ENTER CONTROL VOLUME=SYSTEM
ENTER CONTROL2 LIBRARY=MLHCTL1, VOLUME=SYSTEM
ENTER RPTDEF VOLUME=SYSTEM
ENTER FUNCTION 4
ENTER OPTIONS ID=LEELA, PAGE=40
ENTER RPTDEF VOLUME=SYSTEM
ENTER INPUT1 LIBRARY=MLHLIB, VOLUME=SYSTEM
ENTER INPUT2 LIBRARY=MLHLIB, VOLUME=SYSTEM
ENTER FUNCTION 16
RETURN
```

CHAPTER 7
A SAMPLE APPLICATION USING CONTROL, DATENTRY, AND REPORT

7.1 INTRODUCTION

This chapter describes a sample data management application that illustrates the combined use of three Data File Management utilities: CONTROL, DATENTRY, and REPORT. The example represents a payroll application for an imaginary small business, and includes the following three processes:

- Designing a data file
- Creating the data file and entering the data
- Designing and producing a monthly report based on the data

The data is recorded on forms in the following format:

EMPLOYEE'S ID NMBER	SSN	EMPLOYEE'S NAME	REG. HOURS WORKED	OVERTIME WORKED	VACATION EARNED	HOURLY WAGE
12345	555668888	Doe, John Q.	120	15	2.5	$6.25

The application developer uses this information to design and implement a data entry and reporting system in three basic operations. They consist of:

- Using CONTROL to create the control file
- Using DATENTRY to create and build the data file
- Using REPORT to create a Report Definition file (RDF)

These three operations correspond to the three processes listed above.

7.2 CREATING THE CONTROL FILE

7.2.1 Planning the Data File

Since the control file determines the characteristics of the data file, it is important to decide how the data file will be laid out before beginning to create the control file. Careful planning at this stage helps to avoid tedious modification of the control file after it is created.

For this application, the layout of the data file depends largely on the design and requirements of the monthly report to be produced. The developer therefore considers the information that will appear on the report when selecting and naming the fields in the records that will make up the data file.

In the example described, the report must present for each employee

- A table of regular hours worked each month
- Overtime hours worked each month
- Vacation hours accrued each month
- Hours worked year-to-date
- Wage rate and salary for the reporting month

The data for this report is to be listed by employee number. Since the report is run monthly in a year-to-date format, the file must keep track of the employee's hours for each month as well as information needed only once, such as wage rate.

Taking the requirements into consideration, it is clear that the data file must contain such information as the social security number, employee name, regular and overtime hours for each month, vacation accrued each month, total hours worked year-to-date, and wage rate.

Selecting the Fields

The application developer's first step is determining which of the available fields of information will be in the data file. In this case, all the available fields are needed. The developer decides to expand the field, Employee's Name, into two separate fields, Last Name and First Name/Middle Initial. The cumulative field YTD is also created to represent the sum of the values of all 12 months of the Regular Hours and Overtime Hours fields. For ease of data entry, the developer needs a way to keep the regular and overtime hours for each month together. He does this by defining an additional field (MONTHLY), which contains both regular and overtime hours. Each field is assigned a field name of eight or fewer characters.

This set of fields is compatible with both the original data and the requirements of the report.

The easiest and clearest method for designing the data file is to develop a table of field information, tabularizing the data file specifications as they are selected. The first step is to list the names of the fields:

Original Field	New Field Name
Employee's ID number	EMPNUM
Social security number	SSN
Employee's name	LASTNAME
	FRSTNAME
Monthly (contains both REGHRS/OTHRS)	MONTHLY
Regular hours worked each month	REGHRS
Overtime worked each month	OTHRS
Hours worked year-to-date	YTD
Hourly wage	PAYRATE
Vacation time accrued each month	VACATION

Specifying the Internal Format of the Fields

After selecting and naming the data fields, the application developer specifies an internal format for each field. It is important to understand the distinction between external and internal formats:

- The *external format* is the way the data in a field appears to the user. There are two types of external format, *alphanumeric* and *numeric*.

- The *internal format* determines the manner in which the data in a field is stored internally by the system. The internal format is transparent to the user. There are five types of internal format (four of which are used in this example).

The five types of internal format are

- *Character* (C), used for all alphanumeric fields

- *Zoned* (Z), used for numeric fields

- *Unsigned* (U), used for numeric fields

- *Packed* (P), used for numeric fields

- *Binary* (B), the binary format, is not discussed here; see the description of the CONTROL utility in Chapter 2.

Since all alphanumeric fields have the internal format C, they share the same type of internal representation: each alphanumeric character requires one byte of storage. For example, the string ABC1 is represented internally in four bytes.

In this example, numeric data will be represented internally in three ways. These three internal formats -- zoned, unsigned, and packed -- differ in the number of bytes they require to represent a given piece of data.

Zoned format requires a full byte for each digit, exclusive of sign and decimal point. The sign is stored in the high-order bits of the last byte (see Figure 7-1). For example, *+1234* or *-1234* requires four bytes of storage in zoned format.

Unsigned format also stores one digit per byte, but does not store signs. (If a sign is added through the CONTROL data edit options, the sign is stored in a full byte by itself.) For example, *1234* requires four bytes of storage. A field in unsigned format cannot have validation tables or be an accumulator field.

Packed format requires only a half-byte of storage per digit, plus one additional half-byte for the sign. For example, *+1234* or *-1234* requires three bytes of storage.

Figure 7-1 illustrates data storage in formats C, Z, U, and P:

Data Type	External Format	Internal Format (Graphic Representation)	Internal Format (Hex Representation)
Character	ABC1	A B C 1	4 1 4 2 4 3 3 1
Zoned	-1234	1 2 3 -4	3 1 3 2 3 3 D 4
Unsigned	1234	1 2 3 4	3 1 3 2 3 3 3 4
Packed	-1234	0 1 2 3 4 -	0 1 2 3 4 D

Figure 7-1. Data Storage According to Type

The format chosen for a numeric field is important because it can affect file space requirements and CPU time as well as screen representation of the field length. The choice of format should depend on field length and usage. For fields more than two bytes long, packed format requires less storage space than zoned or unsigned format. Moreover, packed format requires less CPU time than the other formats for fields used in arithmetic operations, since all arithmetic operations are done in packed format. Zoned and unsigned data must therefore be converted internally to and from packed format in order to be used in such operations.

For the reasons given, it is best to use the following criteria when selecting the format for a numeric field:

Format	Suggested Criteria for Selection
Z	Signed numeric fields of one or two digits on which no arithmetic operations will be done
U	Unsigned numeric fields on which no arithmetic operations will be done
P	Signed numeric fields of three or more digits, as well as all fields on which arithmetic operations will be done

The Data File Management Utilities restrict the sizes for all three formats to no more than 15 digits without a decimal point, and 14 digits with one.

Using these criteria, the application developer assigns internal formats to the fields as shown below. The fields EMPNUM, MONTHLY, REGHRS, and OTHRS are designated as unsigned fields because these fields never require a sign and therefore do not require the extra pseudoblank automatically provided by packed format for data entry. Using unsigned instead of packed format eliminates this extra pseudoblank, and with it some potential data entry confusion. Name fields are specified as character fields. YTD and PAYRATE are defined as packed fields. Although these fields do not require a sign, the amount of data entry required is minimal to none, and therefore the risk of data entry errors is not a factor. VACATION has been defined as zoned to demonstrate this field type. Because it may require a sign (if vacation time was taken and the employee uses more days than accrued for the month), it could also be defined as packed. Unsigned would not be a satisfactory data type for this field. Refer to Figure 7-2 for the list of field names and their format types.

Specifying Field Lengths

Since the internal length must be specified for each field, it is good practice to tabulate the number of characters or digits as a working value from which the internal field length can be determined. The developer examines the data to find out how many characters or digits each field requires.

For example, values for the two HRS fields range from 1 to 99 and thus require two digits. PAYRATE, which includes values such as $11.25, requires four digits. (The dollar sign and decimal point are edit characters that can be added when the Report Definition file is created.)

The internal field length is the number of bytes needed to represent the specified number of characters or digits in the data record. The sum of the internal lengths is the total record length for the data file. From the internal format and the number of characters or digits in each field, the application developer determines the internal field lengths from the algorithms listed below. These calculations, as well as those for external length, allow for the required place for a sign in both zone and packed formats.

The external field length is the number of pseudoblanks appearing on the data entry screen. It is unnecessary to specify external field lengths in creating a control file, since the CONTROL utility calculates these automatically from the internal lengths, using the algorithms shown below. However, since some applications require the external field length to be adjusted, it is useful to be familiar with these calculation algorithms.

Internal Format	**Internal Field Length**	**External Field Length**
C	Number of characters required	Internal length
U	Number of digits required	Internal length
Z DEC = 0	Number of digits required	Internal length + 1
Z DEC > 0	Number of digits required	Internal length + 2
P DEC = 0	Number of digits / 2 + 1	(2 x internal length)
P DEC > 0	Number of digits / 2 + 1	(2 x internal length) + 1

Refer to Figure 7-2 for the list of field names and their lengths.

Specifying Validation Methods

The developer's next task is deciding which fields, if any, will be subject to data validation, and whether tables, ranges, or user exit subroutines will be appropriate for this purpose. Based on examination of the data, a range of values is specified for PAYRATE to make sure payments to employees fall within the accepted limits of $5.95 to $15.00 an hour. These validation methods are summarized below.

Additional Characteristics

Before field specifications are complete, the developer notes any characteristics peculiar to a field, for example:

- An update code of 1 for MONTHLY makes the field nonmodifiable. The field occurring 12 times means this field can be used as a complex table. REGHRS and OTHRS are defined as occupying the same space as the first occurrence of the table, making them components of the complex table. (Refer to Section 2.3.4 for details.)

- A display code of 1 for PAYRATE causes the present hourly wage to remain displayed for each data entry until another hourly wage is entered.

- Two decimal positions and a dollar sign are inserted for PAYRATE. (These will not affect internal data storage, but will be displayed when DATENTRY is used to update the file.)

- YTD is specified as a cumulative field for REGHRS and OTHRS.

Specifying the File Organization

Besides laying out the data fields, the application developer must specify a record type for the file. The choice of a record type is based on whether the file is to be indexed and whether the data is to be modified.

The data file in the example will contain one payroll record for each employee, and the records will be modified monthly. Since it is useful to access the data records through particular fields, the file will be indexed. For this example, it is convenient to access records by employee number, employee name, and social security number.

Since each employee has a unique employee number, EMPNUM can be designated the primary key (for which duplicate values cannot be allowed). LASTNAME and SSN can be designated as alternate keys. Since social security numbers are unique, they are not allowed duplicate values whereas many persons may have the same last name and therefore the LASTNAME field is allowed to have duplicate values.

The choice of fixed-length or variable-length records is not crucial for an indexed file, but a consecutive file must have fixed-length records if the data is to be modified.

Preparing the Specifications for CONTROL

Now that the information necessary to define the fields is gathered together, it can be reorganized into a tabular form that facilitates the creation of a control file through the CONTROL utility. The developer moves columns to match the order of specifications from CONTROL screens, and arranges the fields in the order in which they are specified. YTD must be defined before REGHRS and OTHRS, because CONTROL requires an accumulator field to be defined before its source fields. The data formats and internal lengths are also checked for accuracy at this time to prevent time-consuming errors in the control file. Figure 7-2 is an example of a table that can be used easily for entering CONTROL specifications.

Field Name	Data Format	Index Keys	Int. Format	Int. Length	Ext. Length	Dec. Positions	Occurs	Special Features	Validation Methods
EMPNUM	U	U	U	5	5	0	01		
SSN	P	A1	P	5	9	0	01		
LASTNAME	C	A2	C	15	15	--	01		
FRSTNAME	C		C	15	15	--	01	Cum Fld for REGHRS & OTHRS	
MONTHLY	U		U	4	4	0	12	Update code = 1	
REGHRS	U		U	2	2	0	01	Cum Fld = YTD	
OTHRS	U		U	2	2	0	01	Cum Fld = YTD	
YTD	P		P	3	5	0	01	2 decimal pos	Range 5.95
PAYRATE	P		P	3	7	2	01	Dollar/comma = 2 1 decimal pos	to 15.00
VACATION	Z		Z	3	5	1	12		

Figure 7-2. CONTROL File Data Specifications

7.2.2 Entering the File and Field Information for CONTROL

When the planning stage is finished, the application developer runs the CONTROL utility to create the control file. The first step is to supply a name, library, and volume for this file, then to press PF2 to indicate that a new control file is being created. The developer chooses to name the file PAYROLL, since it will house weekly payroll information. CONTROL supplies a default "control library" by appending CTL to the user's ID, e.g., USRCTL.

The next screen requests header information for the data file. The specifications shown in Figure 7-3 are entered in the table of field information.

```
***  MESSAGE 0001 BY CONTRO

                  INFORMATION REQUIRED BY PROGRAM CONTROL
                              TO DEFINE HEADER
                          ACTIVE SUBPROGRAM IS CONTROL

Creation of file header information.  Enter parameters for the data file.

RECLEN   = 0200          (2024 var/comp; 2040 fixed/relative; 2048 consec)
FILETYPE = F             (F-Fixed, V-Variable, C-Compressed records)
FILEORG  = I             (C-Consecutive, I-Indexed, R-Relative)
KEYFIELD = EMPNUM        (specify if FILEORG is Indexed)
USEREXIT =               (user supplied, if any: "USER1" -- "USER10")
UPDATE   = 0             (0-record update allowed, 1-not allowed)
DELETE   = 0             (0-record deletions allowed, 1-not allowed)
REPORT   = 0             (0-report allowed, 1-not allowed)
OPENSHAR = Y             (Shared mode for DATENTRY, Y-Yes, N-No)
TIMEOUT  = 000           (Timeout for DATENTRY, 000 - 255 seconds)
DEFAULT DATA   FILE      = PAYROLL  LIBRARY  = USRDATA  VOLUME   = VOL111
COMMENT1 = THIS CONTROL FILE DESCRIBES THE DATA FILE PAYROLL WHICH
COMMENT2 = CONTAINS MONTHLY PAYROLL INFORMATION.
COMMENT3 =
(ENTER) Continue header creation OR          (16) Return to file specification
```

Figure 7-3. Sample File Header Specifications for CONTROL

For a description of the screen and the specification process, see Section 2.3.1. The specifications shown in Figure 7-2 include the following:

- Data file record length, set to 200 to allow some extra space if additional fields need to be added.
- File type: fixed-length records.
- File organization: indexed.
- Primary key: EMPNUM.
- User exit subroutine: none specified.
- Update, delete, and report codes:
  - The DATENTRY utility can modify the data records.
  - DATENTRY can delete records.
  - The REPORT utility can access the file to extract information.
- Shared mode and timeout codes:
  - DATENTRY can open the file in shared mode.
  - No timeout. DATENTRY will wait indefinitely for the file.
- Default name and location for the data file: PAYROLL in library USRDATA on volume VOL111.
- Comments that identify the purpose of the control file.

The codes that govern updating, deleting, and reporting on the data file apply to the Data File Management Utilities only; they do not control access to the data file by other means. The default name selected for the data file is the same as that for the control file, a recommended procedure. (The two files must, of course, be housed in separate libraries.) The comments in the three Comment fields appear only when the CONTROL utility is used to list header information. For more detailed information on the contents of a control file header record, see Section 2.1, Section 2.3.1, and Appendix C.

The next screen (which appears because an indexed file organization was specified) prompts for the alternate keys, if any, to be listed. Since this file is to have two alternate keys, the developer enters their field names: SSN and LASTNAME, and for each field specifies that duplicate values are allowed. ("Defining Alternate Keys" in Section 2.3.1 describes this process.)

After alternate keys are specified, the first Field Specification screen appears. Specifications for the first field in the record, EMPNUM, are entered as shown in Figure 7-4.

```
                        FIELD SPECIFICATION SCREEN

                      Fill in the appropriate information

Field name   : EMPNUM
Start loc    : 0001      (Starting location within the data record)
Int. format  : U         (P-packed, B-binary, C-character, Z-zoned, U-Unsigned)
Int. length  : 005       (Maximum values: P-8, B-4, C-132, Z-15, U-15)
Ext. length  :           (only if not calculated from internal length)
Decimal pos  : 0         (0-9, specify for numeric fields only)
Occurrences  : 01        (number of times this field repeated)
Report code  : 0         (0-report on this field, 1-no report allowed)
Update code  : 0         (0-modifiable, 1-nonmodifiable, no DATENTRY processing
Display code: 0          (0-blank field, 1-display last value, 2-display only)
0-suppress   : 0         (0-no zero suppress, 1-zero suppress, 2-* protect)
Sign control: 0          (0-no sign, 1-trailing minus, 2-CR on -, 3- DB on -)
Dollar/comma: 0          (0-none  1-comma, 2-dollar, 3-comma and dollar)
Binary edit  : 0         (0-hex, 1-decimal, Binary format only)
Stamp field  : 0         (0-no, 1-Date(MMDDYY), 2-Date(YYMMDD), 3-Day, 4-Time)
Cum. field   :           (cumulative field name)
Field alias  :                                  (alternate name used by INQUIRY)

(ENTER) Continue field addition OR
                                                  (16) Return to Central Menu
```

Figure 7-4. Sample Field Specifications for CONTROL

The default starting location, byte 1, is correct for EMPNUM. (CONTROL uses the field length to compute the default starting location for the next field, and for other fields as they are added.) As decided in the planning stage, an internal format of U and an internal length of 5 are specified. No external length is specified, however, since the CONTROL utility will compute that internally.

Other specifications indicate that this field occurs only once in the record and that reporting and modification via the REPORT and DATENTRY utilities are permitted. For EMPNUM, default values are accepted in all the fields of the Field Specifications screen after Internal Length, but, as the previous section indicates, this is not true for all eight fields being specified. See Section 2.3.2 for detailed instructions.

When specifications have been completed and entered for any data field less than 16 bytes in length, CONTROL displays a screen from which table or range validation can be specified for that field. The developer uses this screen to specify range checking for PAYRATE.

For PAYRATE, the developer specifies the lower and upper limits of the range, namely 5.95 and 15.00, from the same screen. For other fields, ENTER is pressed from the blank screen to decline all validation options. See Section 2.3.3 for detailed instructions.

Once all fields, the table, and all ranges have been specified, CONTROL creates the control file. Its header and field information can be examined by the pressing of PF6 from the CONTROL central menu. The control file can be modified by means of the various options described in Section 2.4. If all specifications are correct, it is ready to be used, together with the DATENTRY utility, to create a data file.

7.3 CREATING THE DATA FILE

The DATENTRY utility (described in Chapter 3) can be run either from the CONTROL central menu (PF9) or the Command Processor menu. The developer's first step in creating a data file is to specify its name and location. It is recommended that the data file have the same name as the control file. (The control file must also be specified, but if DATENTRY is run from the CONTROL central menu, the file information for the currently specified control file appears as default values in the appropriate file, library, and volume fields of the DATENTRY screen. This screen and its fields are described in Section 3.2.)

When data file and control file information has been specified, the developer presses PF2 to create a new data file, specifying file attributes such as the size (in number of records). A version of the DATENTRY central menu then appears, displaying the options available for a data file that, as yet, contains no records. The developer presses PF3, Add Records.

A data entry screen (Figure 7-5) appears, on which DATENTRY displays the name of each modifiable field in the data record, followed by a field of pseudoblanks into which a value can be entered. The modifiable fields appear in the order in which they occur in the control file (See Section 2.4.7 for detailed instructions.)

```
VS Data Entry Utility  File: PAYROLL  Library: USRDATA  Volume: VOL111
EMPNUM 05005 SSN 555668888 LASTNAME FRANK■■■■■■■■■■ FRSTNAME ANN■■■■■■■■■■■■
REGHRS (01) 160 OTHRS (01) 05 REGHRS (02) 160 OTHRS (02) 20 REGHRS (03) ■■■
OTHRS (03) ■■ REGHRS (04) ■■■ OTHRS (04) ■■ REGHRS (05) ■■■ OTHRS (05) ■■
REGHRS (06) ■■■ OTHRS (06) ■■ REGHRS (07) ■■■ OTHRS (07) ■■ REGHRS (08) ■■■
OTHRS (08) ■■ REGHRS (09) ■■■ OTHRS (09) ■■ REGHRS (10) ■■■ OTHRS (10) ■■
REGHRS (11) ■■■ OTHRS (11) ■■ REGHRS (12) ■■■ OTHRS (12) ■■ YTD ■■■■■■■
PAYRATE 6.25■■■■ VACATION (01) 2.5■■ VACATION (02) -.5■■ VACATION (03) ■■■■■
VACATION (04) ■■■■■ VACATION (05) ■■■■■ VACATION (06) ■■■■■ VACATION (07) ■■■■■
VACATION (08) ■■■■■ VACATION (09) ■■■■■ VACATION (10) ■■■■■ VACATION (11) ■■■■■
VACATION (12) ■■■■■
(ENTER) Add record  (9) Exit to Modify a record  (12) Table/Range     (16) Exit
```

Figure 7-5. Sample Data Entry Screen

In the example, the field YTD appears but is nonmodifiable. YTD receives its value from the entries in the REGHRS and OTHRS fields, which it sums. The data is entered in the appropriate fields.

Figure 7-5 shows the data for the first record entered in the appropriate fields. The tab key is used for moving between fields. When ENTER is pressed, the record is placed in the data file, and all fields (except PAYRATE, whose display code is 1) are cleared for the entry of the next record.

When all the data has been entered, PF16 permits a return to the DATENTRY central menu, from which it is possible to exit from the program.

Once the payroll data has been stored in the new data file, the REPORT utility can be run on it to extract, combine, and rearrange data in various forms. In successive months, the records can be recalled and modified through DATENTRY. Any change made in the values of the accumulator source fields REGHRS and OTHRS automatically changes the value of the accumulator field YTD, which always contains the sum of the values in the source fields. The data records can be accessed for review or modification by any of the keys specified for the data file, and, within a key path, by sequence of records or by key value.

7.4 CREATING THE REPORT DEFINITION FILE

7.4.1 Introduction

The REPORT utility permits the user to design a report to be run on one or two data files (each must have a corresponding control file). REPORT saves the specifications for a report in a Report Definition file (RDF), which can be used repeatedly to produce reports on the specified data file or files. In the example, for instance, an RDF could be used to run a monthly report on the payroll data as soon as the data file PAYROLL has been updated for that month.

7.4.2 Planning the Report Format

As in control file creation, the first step in creating an RDF is to plan the report format before entering any specifications. The application developer begins with a rough draft of the report format, which will be developed into a detailed design. This draft reflects the eventual format the developer desires, rather than the details of content. It serves as a basis on which to create the structure of the report.

In the example, the report is to present the regular and overtime hours worked for each month, total vacation accumulated for each month, and total hours worked as well as the wage rate and total pay for each employee. This data is to be organized by employee number.

Designing a Draft

It is easiest to begin designing the draft format by considering the columns. The information presented for each employee will be tabulated in columns corresponding to the items presented for each employee: SSN; employee name (last, first); regular, overtime, and total hours; wage rate; and salary for the reporting month. Starting from this information, the developer assigns column headings corresponding to the contents, and also selects a one-line report title, *EMPLOYEE WAGES*. With Xs used to represent the data, the rough draft is laid out as indicated in Figure 7-6.

EMPNUM	SSN	EMPLOYEE NAME	EMPLOYEE WAGES											
	PAY RATE	HOURS WORKED YTD	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
	MONTHLY SALARY	PAY THIS MONTH												
XXXXX	XXXXXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	XXX	XXX	XXX	XXX	XXX	XXX	XXX	XXX	XXX	XXX	XXX	XXX
	XX.XX	XXXXX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX	XX
	XXXX.XX		XX.X	XX.X	XX.X	XX.X	XX.X	XX.X	XX.X	XX.X	XX.X	XX.X	XX.X	XX.X

Figure 7-6. A Sample Draft

Using this draft, the developer can begin to specify the structure and details of the report:

- Fields (including new fields, sort fields, and control break fields) must be selected, organized, and designated for specific uses within the report.

- Titles and headings (such as control break descriptors) must be assigned to specific report components.

- Appearance attributes (like the spacing between fields and sections) must be worked out in detail.

Once all these design decisions are made, the report specifications can be entered through the REPORT utility.

Selecting Fields

The developer selects the fields needed to print a report by examining the column headings in the rough draft. For this report, it is clear that all fields are needed. Although the draft specifies a column for monthly salary, no such data field exists. A new field must be created for this purpose through REPORT. The developer names this field SALARY, and defines it as a numeric field whose value is the sum of

- REGHRS (XX) multiplied by PAYRATE

- OTHRS (XX) multiplied by 1.5 times PAYRATE (since overtime pays time and a half)

(XX) represents the reporting month. SALARY is a dollars field that may run into four figures; it will also have two decimal places, a dollar sign, and a comma, giving it a total external length of eight. REPORT automatically adds two places, one for the decimal point and one for the sign, for a final total external length of ten. (This external length is discussed further in the section entitled "Selecting Spacing.")

For some reports there may be fields in the data file that never need to be accessed, let alone printed (such as MONTHLY in our example). These fields need not be selected. Other fields may have to be used (for instance in calculating the value of a new field), even though they are not themselves printed. REPORT allows such fields to be selected, but marked in such a way that they are not printed, even though they can be accessed and used in other ways. A field in this category is also available for future use if the RDF is modified.

Once all the fields have been selected, the developer is ready to work out their sequence in the report and assign a sequence number to each field. The order of fields is determined by examining the data columns from left to right. Fields that are selected but are not to be printed are assigned a sequence number of 99. Such fields are accessible although they do not appear in the final report. For this example, sequence numbers are assigned as follows:

Field ID	Seq	Field ID	Seq	Field ID	Seq	Field ID	Seq
EMPNUM	1 01	SSN	1 02	FRSTNAME	1 03	LASTNAME	1 04
REGHRS (01)	1 05	REGHRS (02)	1 06	REGHRS (03)	1 07	REGHRS (04)	1 08
REGHRS (05)	1 09	REGHRS (06)	1 10	REGHRS (07)	1 11	REGHRS (08)	1 12
REGHRS (09)	1 13	REGHRS (10)	1 14	REGHRS (11)	1 15	REGHRS (12)	1 12
PAYRATE	2 01	YTD	2 02	OTHRS (01)	2 01	OTHRS (02)	2 02
OTHRS (03)	2 03	OTHRS (04)	2 04	OTHRS (05)	2 05	OTHRS (06)	2 06

ETC....

Selecting Titles and Headings

Having finished selecting and organizing the fields to be used in the report, the developer takes up the question of titles and headings. The main title for a report can be up to three lines long and 60 characters wide. The title for this report, however, is simply *EMPLOYEE WAGES* on one line, as specified in the rough draft. REPORT will automatically center this title on the top line of the report.

The rough draft also contains column headings. Since there is only one heading for the two columns FRSTNAME and LASTNAME, the word EMPLOYEE is put to the far right of the FRSTNAME field and the word NAME is put to far left of the LASTNAME field.

Since a report may contain more than one subdivision, REPORT requires that a level be specified for each control field and control break descriptor. (In a larger company, for example, a payroll report might subdivide the data by department and within departments by section. DEPT would be the level 1 control field and SECTION the level 2 control field. The corresponding control break descriptors (for instance, *DEPT. WAGES:* and *SECTION WAGES:* would have to be specified with a level corresponding to that of the appropriate control field.

Selecting Spacing

The placement of column headings and data, accomplished by means of spacing, is important in creating an attractive and legible report. The developer must consider the data, the column headings, and the width of the form or paper on which the report is to be printed. (This last item is specified, in characters, as the print line length.)

The first consideration in report spacing is whether or not all the data and headings can fit within the specified print line length. 132 characters is the usual print line length for standard 11 by 14 inch computer forms. To determine how many print lines will be required, the developer lays out a diagram of the column headings and data fields with their print lengths. (When modifying an existing RDF through REPORT, the user can use an option, Print Headings and Dummy Detail Lines to view a simulated report in draft format. Since this RDF has not yet been created, however, the developer uses pencil and paper for this purpose.) Using Xs and 9s to represent character and numeric data, the developer sketches out columns across the page. The total width for a given column is its heading or field length, whichever is greater. (The employee number column heading, for example, is six characters long, while the data is only five; this column is therefore six characters wide.)

In our example, the print line length poses a problem for this report. It must be reduced before report definition can proceed. The easiest way to reduce the print line length here is to shorten fields or column headings that are longer than necessary or arrange the data on the second or third print line; eliminating fields is drastic and unnecessary.

To eliminate excess space within fields, the developer compares the actual data to the allotted external field lengths to determine if all the allotted spaces are actually needed for data. Any external field length that is found to be unnecessarily large can be truncated in the report to eliminate wasted space and to allow greater flexibility in positioning the columns. These not only extend the print line unnecessarily; they also prevent two columns from being placed closer together, and make the report appear discontinuous and unattractive.

Examining the data, the developer learns that the first name field has a greater external length than is required by its data. The longest entry for FRSTNAME has only nine characters. This reduces the column, and therefore the print line length. Printing the last and first names of the employees closer together also improves the appearance of the report.

The print line is still too long. We also have the problem of keeping the monthly columns neatly stacked one above the other. By coming up with a standard column header for all three lines, this can be simplified and the line lengths shortened. Each column represents a month; therefore, monthly abbreviations are used.

The next task for the developer is to center the column headings within their respective columns. Column headings are positioned by the user; REPORT automatically centers each column of data under its heading. In this report, EMPLOYEE is the only heading shorter than the data below it. The other headings do not have to be centered, since each one runs all the way across its column.

Since the column heading EMPNUM is wider than its 5-digit data, REPORT will center the data under its heading with a space after it. The data for REGHRS, OTHRS, VACATION, and PAYRATE will be centered in the same way, with REPORT calculating the number of spaces to place before and after the data in order to center it under the heading. (For PAYRATE, REPORT adds one place to the original external length of 4 to allow space for the dollar sign specified in the control file. With the decimal point included, PAYRATE has an external length of 6 in the report.)

The calculations for SALARY are somewhat more involved than for the other fields. The developer originally specifies the external length for the new field SALARY as 8, to allow for its six digits, dollar sign, and comma. REPORT automatically adds two places (for a total of 10): 1 for a decimal point and 1 for a sign. However, since a sign is not appropriate for this report, the developer intends to specify that it be deleted when the report is printed. That reduces the external length of SALARY to 9, which, since it is the same length as the heading, requires no centering spaces to be added.

7.4.3 Specifying the Report Format

Once the design of the report is finished, the application developer can enter the specifications for creating an RDF. The REPORT utility can be run from the Command Processor menu or by pressing PF10 from the CONTROL central menu. From the REPORT central menu, the developer presses PF2, Create a Report Definition.

Source Files

The first screen to appear after the Create a Report Definition screen, requests the ID of the report being created. This will be the file name of the RDF. To maintain consistency with the control and data files, the developer names the report PAYROLL. This screen also asks whether an additional data file will be used; in this example, the default value NO is accepted.

The next screen requests the name and location of the primary data file. (When this is supplied, a third screen may appear to request the same information for the corresponding control file, if this cannot be determined from such defaults as the data file name, the user's control library, e.g., USRCTL, and the INVOL value from the user's usage constants.)

Selecting Fields

Once the source files have been specified, REPORT prompts for the selection of the fields to be used in the report, first from the primary data file, and then from the secondary file, if any. REPORT displays a list of the fields in the data file, and the user selects individual fields by entering an X next to each field chosen. In the example, the developer marks all the fields in the data file PAYROLL, except the fields MONTHLY (xx), and presses ENTER.

New Fields

When fields have been selected from the data file or files, REPORT displays a screen that asks whether new fields are to be defined. In the example, the developer responds YES. The next screen requests the names, types (numeric or character), lengths, and (for numeric fields only) number of decimal positions for up to 10 new fields to be defined in the RDF. Specifications for SALARY are entered as indicated in Figure 7-7.

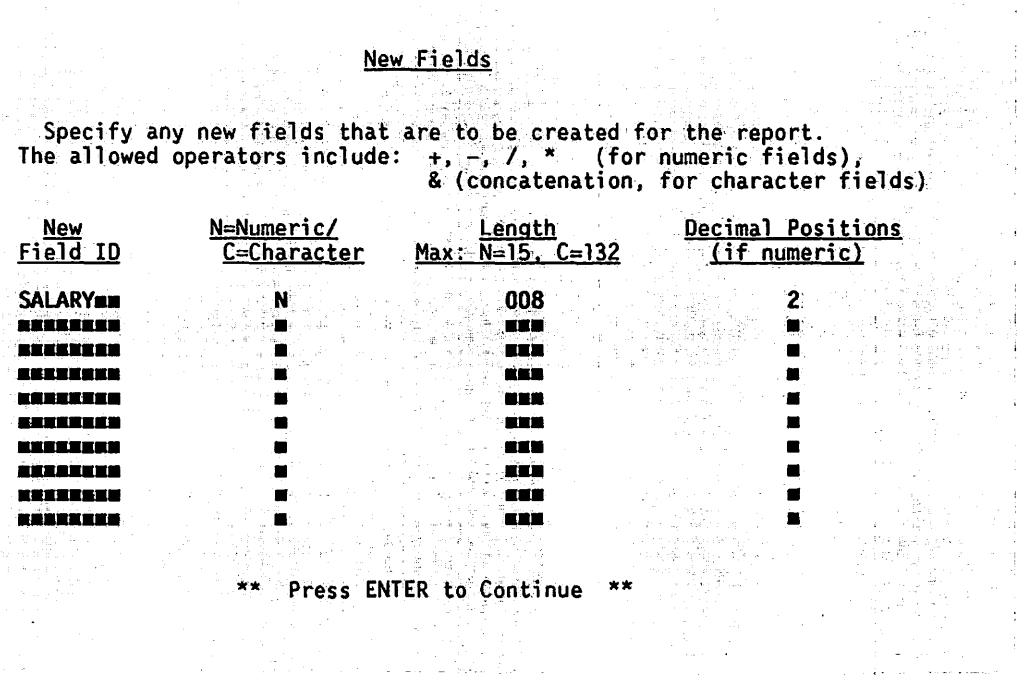


Figure 7-7. New Field Specification for REPORT

SALARY, a numeric field, represents a dollar value that may run to four figures, plus two decimal positions (six characters in all). Adding one character for a comma and one for the dollar sign, the developer specifies its length as eight bytes. The last column gives the number of decimal positions, which must be included in the specified length.

Since SALARY is the only new field, the developer presses ENTER after these specifications are complete. A second New Fields screen appears, prompting the user to specify the way the new field named at the top of the screen is to be defined. (If multiple new screens were specified, this screen appears once for each of them.)

Since REPORT has no access to data from outside the input data file or files, the only way it can obtain values for a new field is by manipulating the data in the fields of this file or files. (In the example, only one data file is being used for input.) Values can be derived by combining existing fields with each other and/or with constant or literal values in any of several ways: Numeric fields and/or constants can be added, multiplied, divided, or subtracted. Character fields can be concatenated with each other or with literals. A sequence of such operations can be specified to obtain a value for a particular field. The value of SALARY is defined as indicated in Figure 7-8.

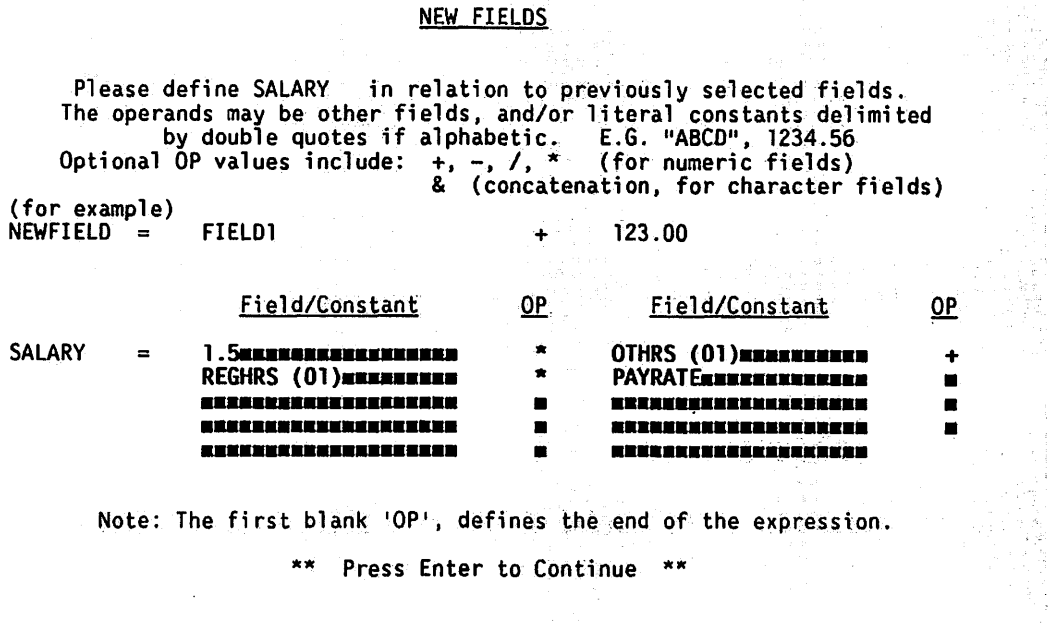


Figure 7-8. New Field Definition for REPORT

The value for SALARY is derived by multiplying OTHRS (01) by 1.5, adding the result to REGHRS (01), and multiplying the total by PAYRATE. This definition is entered as shown in Figure 7-8.

Supplying the definition through which REPORT can obtain values for the SALARY field completes the process of new field specification. The developer goes on to specify the format of the report through the options available from the REPORT Options menu. These options can be accessed directly by means of the PF keys indicated on the menu. After the user selects an option, however, REPORT automatically moves from one to the next in sequence as ENTER is pressed from each screen.

Titles and Headings

When the new field has been defined, the REPORT Options menu appears. The developer selects the first available option, PF2 (Report Title Information). The first screen to appear after all new fields are defined requests the main title of the report, the page heading (which will appear at the head of each page after the first), and the control break descriptors with their starting column positions.

The report title, as determined earlier, is EMPLOYEE WAGES. Since the report is unlikely to be long enough to require a second page, the developer does not enter a page heading; the report title will be used as a default page heading if the report should require it.

Column Headings

The next screen shows all the fields selected for the report in alphabetical order and prompts for the column headings and subheadings. The name of each field is shown as a default heading (see Figure 7-9).

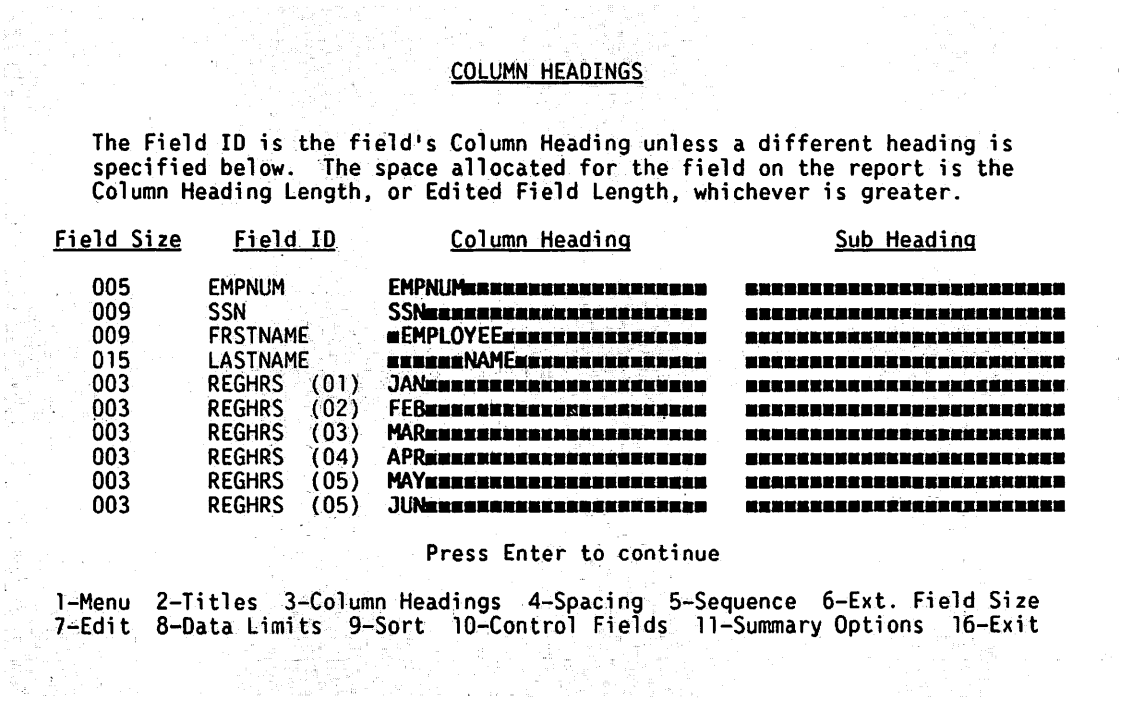


Figure 7-9. Column Heading Specifications for REPORT

Field Spacing

The next screen requests the number of spaces to be printed before each field in the report. A default value of 2 is displayed for each field. Once again, all fields selected are shown in alphabetical order. The developer enters 0 in place of the default value for EMPNUM (since it is to appear at the left margin, no spaces should be printed before it). SALARY, YTD, REGHRS (01), OTHRS (01), and VACATION (01) must be increased to align the data under the monthly column headers. All other OTHRS fields retain the default value of 2. All other VACATION and REGHRS fields are decreased to 1 to allow alignment under the column headers.

Field Sequence

The next format option after column spacing is the sequence in which the columns will be printed across the page. As a default, REPORT lists the fields alphabetically and assigns each one a sequence number corresponding to that order. The screen also prompts the user to specify the data line (1, 2, or 3) on which the field value is to be printed. The report in the example uses all three lines for each record. This example shows the sequence numbers after their proper assignment.

```
                                FIELD SEQUENCE

  Fields selected for the report are listed below.  Indicate the sequence in
  which the fields are to appear on the report by specifying the line (1,2,or3)
  and the position (1-80) on that line.  Fields with a '99' position do not
  appear on the report, but can be used in the preparation of the report.

  Field ID        Seq     Field ID        Seq     Field ID        Seq     Field ID        Seq

EMPNUM            1 01    SSN             1 02    FRSTNAME        1 03    LASTNAME        1 04
REGHRS  (01)      1 05    REGHRS  (02)    1 06    REGHRS  (03)    1 07    REGHRS  (04)    1 080
REGHRS  (05)      1 09    REGHRS  (06)    1 10    REGHRS  (07)    1 11    REGHRS  (08)    1 12
REGHRS  (09)      1 13    REGHRS  (10)    1 14    REGHRS  (11)    1 15    REGHRS  (12)    1 16
PAYRATE           2 01    YTD             2 02    OTHRS   (01)    2 03    OTHRS   (02)    2 04 0
OTHRS   (03)      2 05    OTHRS   (04)    2 06    OTHRS   (05)    2 07    OTHRS   (06)    2 09
OTHRS   (07)      2 10    OTHRS   (08)    2 11    OTHRS   (09)    2 12    OTHRS   (10)    2 13
OTHRS   (11)      2 14    OTHRS   (12)    2 15    SALARY          3 01    VACATION(01)    3 02
VACATION(02)      3 03    VACATION(03)    3 04    VACATION(04)    3 05    VACATION(05)    3 06

                          ** Press ENTER to Continue **

  1-Menu  2-Titles  3-Column Headings  4-Spacing  5-Sequence  6-Ext. Field Size
  7-Edit  8-Data Limits  9-Sort  10-Control Fields  11-Summary Options  16-Exit
```

Figure 7-10. Field Sequence Specifications for REPORT

External Field Size

The External Field Size screen, which follows the Field Sequence screen, permits the user to modify the external lengths of the fields. A default value is shown for each field. The developer uses this screen to truncate FRSTNAME from 15 characters to 9, as worked out in the planning stage. SALARY is also truncated from 10 characters to 9 to eliminate the place created for its sign. PAYRATE (for reasons internal to the CONTROL utility which it is unnecessary to consider here) has a default length of 11 characters, far more than it needs, since it does not have to record a dollar value greater than $99.99. PAYRATE is therefore truncated to 6 to accommodate the four digits, decimal point, and dollar sign. The other default values are left alone.

Data Edit Options

After the External Field Size screen, the Data Edit Options screens appear. These screens enable the user to modify the way numeric fields are represented externally. The first screen displays the current representations for all numeric fields (e.g., 99999.99- for SALARY) and asks the user to mark those fields whose representations need changing. The developer types an X next to each of the field names PAYRATE, REGHRS, OTHRS, TOTHRS, and SALARY, and presses ENTER.

The second Data Edit Options screen lists the selected fields and prompts the user for edit specifications for each (see Figure 7-11).

OUTPUT DATA EDIT OPTIONS

The nominal data edit formats are displayed for each field that is to be modified. For an explanation of the codes, press PF14.

Field ID	(*N,ZN) Suppress Zeroes	(CR,DB,9-) Sign Control	(.N) Decimal Carry	N(.,/*-) Special Insertion	($$,$9) Dollar Sign	(Y or N) Commas
REGHRS	Z1	■■	.0	■■ ■■ ■■	■■	N
OTHRS	Z1	■■	.0	■■ ■■ ■■	■■	N
YTD	Z1	■■	.0	■■ ■■ ■■	■■	N
PAYRATE	Z3	■■	.2	■■ ■■ ■■	$$	N
SALARY	Z3	■■	.2	■■ ■■ ■■	$$	Y

** Press ENTER to Continue **

1-Menu 2-Titles 3-Column Headings 4-Spacing 5-Sequence 6-Ext. Field Size
7-Edit 8-Data Limits 9-Sort 10-Control Fields 11-Summary Options 16-Exit

Figure 7-11. Numeric Field Edit Specifications for REPORT

The developer wants zero suppression for REGHRS, OTHRS, and YTD, and therefore enters Z1 in the Suppress Zeroes column for each of these fields. This entry suppresses 0s up to but not including the rightmost column; a single 0 (for no hours) can be printed in this column, but eight hours will appear as *8* instead of *08*. Zero suppression for PAYRATE and SALARY is handled by entering Z3 to suppress 0s up to the rightmost *dollar* position: *$1.23* instead of *$001.23*, but *$0.03* instead of *$.03*.

Since REPORT has included a sign for the new field SALARY, the Sign Control column contains a default value of -9 for this field. The developer types blanks over this value to eliminate the sign from the printed report.

To precede the values in PAYRATE and SALARY with a floating dollar sign, the developer enters $$ in the Dollar Sign column for each of these fields.

The Decimal Carry column needs no modification, since it already indicates that two decimal places to the right of the decimal point will be printed for PAYRATE and SALARY, and no decimal places for any other field.

Finally, to insert a comma into a 4-figure dollar value for SALARY, the default value N in the Commas column is changed to Y.

Note that no editing, not even zero suppression, was specified for the numeric field EMPNUM. This is because employee numbers should be represented in their entirety.

Data Limits

The Data Limits screen, which appears next, allows the user to specify limits for the values of report fields. The application developer could, for instance, use this screen to specify that only the records of employees who worked overtime, or who worked 40 regular hours, or who earned between $10.00 and $11.99 per hour, be included. All records that fail to meet the specified conditions are rejected; only the records that qualify are printed in the report.

In this example, however, the developer intends all the records in the data file to be printed, and therefore does not alter this screen.

File Sort

The File Sort screen follows the Data Limits screen, and requests the names of the fields on which the data is to be sorted. The level (indicating precedence among the sort fields) and the sort order (ascending or descending) must also be specified for each field. Because our report is in employee number order and the file is indexed by employee number, we do not need to specify file sorting.

Control Fields

The Control Fields screen follows the File Sort screen, and requests the names of any Control fields used for control breaks. Since we do not require control breaks we continue to the next screen.

Report Summary Options

The last screen for REPORT options lists all the numeric fields and asks the user to indicate which, if any, summary values are to be printed on the summary lines after each control break and at the end of the report. The choices are total, maximum, minimum, and average; these values can be printed for any or all fields, even in different combinations for different fields, if desired. In the example, only the total value of SALARY is to be printed, so the developer types an X in the Total column for this field, and leaves the rest of the screen unmarked. When ENTER is pressed, the Format Options menu appears, and the developer presses PF16 to return to the REPORT central menu.

From the central menu, the developer can press PF3 to modify any of the format options (through the Format Options menu), or PF4 to print the report as defined in the RDF (Report Definition file). This file is automatically given the same name as the data file and deposited in the user's "Report Library." The report library is named by concatenating the string *RPT* with the user's ID (e.g., USRRPT). (The default volume is taken from the OUTVOL field of the user's usage constants.)

7.4.4 Printing the Report

Since it is often best to examine a report on the workstation screen before printing it, the developer presses PF4 (Print a Report) from the REPORT central menu. The screen that appears requests the report ID (PAYROLL), the report date (used if a date is specified for the report), and the output device. The default value for this last item is PRINTER, but the developer changes it to DISPLAY to examine the report. All other default values on this screen are accepted, and the developer presses ENTER. REPORT links to the DISPLAY utility, and the first part of the report appears on the screen. The standard DISPLAY commands (documented in Chapter 11) can be used to move around in the report and examine it in detail.

If the displayed report is satisfactory, the developer can return to the central menu by pressing PF16. The developer then presses PF4 again, changes DISPLAY to PRINTER in the Output Device field, and presses ENTER to begin printing.

If anything needs modification, the developer can press PF3 (Modify Report Attributes) instead of PF4 from the central menu. The Format Options menu appears, and any of the formatting options can be selected and respecified. After being modified, the report can be examined again as described above. Figure 7-12 shows the sample report in printed form.

EMPNUM	SSN	EMPLOYEE NAME	EMPLOYEE WAGES											
	PAY RATE	HOURS WORKED YTD	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
	MONTHLY SALARY	PAY THIS MONTH												
02341	555668888	JOHN J. DOE	160	0	0	0	0	0	0	0	0	0	0	0
	$6.25	180	20	0	0	0	0	0	0	0	0	0	0	0
	$1187.50		2.5	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0
56957	768349738	SALLY M. SMITH	160	0	0	0	0	0	0	0	0	0	0	0
	$11.00	195	30	0	0	0	0	0	0	0	0	0	0	0
	$2255.00		3.5	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0

Figure 7-12. A Sample Report

Part II

GENERAL FILE MANAGEMENT UTILITIES

CHAPTER 8
COMPARE

8.1 INTRODUCTION

The COMPARE utility enables you to compare the contents of two files, libraries, or volumes.

COMPARE offers three types of file comparison:

- A rapid survey of two files, ending with the message that they are or are not identical
- A split-screen display that shows 60 characters of each record in both files and marks all mismatched records
- For program source files only, a listing file containing every line that was deleted from or inserted in the earlier file in order to produce the later one

Library comparisons can show lists of the files that appear in either library exclusively or the files that appear in both libraries.

- From the display of common files, you can position the cursor on any file name and have the program compare the identically named files. They are first checked for type, record size, and number of records, and then, if identical in these respects, compared byte by byte. Identically named program source files (i.e., consecutive files with 80-byte records) can be compared record by record and the differences recorded in a listing file.
- Library comparisons can also include automatic file comparison. If you specify the appropriate options before the libraries are compared, any of the types of comparison you can specify for individual files can be done automatically as part of the library comparison. The screen that identifies files common to both libraries summarizes the results of these file comparisons.

Volume comparisons can show you lists of the libraries that appear exclusively in either volume or the libraries that appear in both. From the latter display (which indicates whether or not the duplicate libraries contain equal numbers of files) you can position the cursor on any library name and initiate a library comparison.

COMPARE's several functions are thus all available no matter which level you choose to begin with: volume, library, or file comparison.

Note: *COMPARE is subject to the limitation that it cannot compare more than 1,000 files in a library or 1,000 libraries on a volume.*

8.1.1 Running COMPARE

Run the COMPARE program from the Command Processor as you run any other program or utility. Enter COMPARE into the program name field along with the appropriate library and volume names.

The following sections describe COMPARE processing:

Section	Process
8.2	Initial Specifications
8.3	Comparing Files
8.3.1	File Comparison Display Mode
8.3.2	File Comparison Source Difference Mode
8.3.3	File Comparison Summary Mode
8.4	Comparing Libraries
8.5	Comparing Volumes
8.6	On-Line Information

8.2 INITIAL SPECIFICATIONS

When COMPARE processing begins, the Input Specification screen (Figure 8-1) appears. This screen has three functions:

- By entering information in the appropriate fields, you specify the files, libraries, or volumes to be compared.
- By choosing to leave certain fields blank, you can specify a volume or library comparison instead of a file comparison.
- When you are comparing files, you use this screen to specify one of the three available comparison modes.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  INPUT
                                                          Message Id:  000
                                                           Component:  COMPARE

                        Information Required by COMPARE

        VS File Comparison Utility - Version  x.xx.xx (c) Copr. Wang 1988

 Please assign the input DISK files and press ENTER to continue:

 Input 1:    FILE1    = ________   LIBRARY1 = ________   VOLUME1  = ______
 Input 2:    FILE2    = ________   LIBRARY2 = ________   VOLUME2  = ______

 Comparison  MODE     = DISPLAY    (DISPLAY / SUMMARY / SRCDIFF)


 Or Select:
                                    (13) Information  (16) To exit COMPARE
```

Figure 8-1. COMPARE Input Specification Screen

Enter information in the fields of the Input Specification screen as follows:

FILE1 -- The name of the first file to be compared. If you are requesting a source-difference listing (MODE = SRCDIFF), FILE1 should be the "older" file. Leave this field blank if you want a library or volume comparison.

LIBRARY1 -- The name of the first library to be compared. Specify this for both file and library comparisons. If you are doing a library comparison and intend to request source-difference listings later in the process, LIBRARY1 should be the library that contains the "older" source files.

For a volume comparison, this field must be blank. When the screen appears, however, the field contains the default, i.e., the INLIB value previously set through the SET USAGE CONSTANTS function of the Command Processor. You must therefore blank out the field if you want a volume comparison.

VOLUME1 -- The name of the first volume to be compared. This must always be specified. The default is the INVOL value previously set through the SET USAGE CONSTANTS function of the Command Processor.

FILE2 -- The name of the second file to be compared. This should be the "newer" file if MODE = SRCDIFF. Leave this field blank for a library or volume comparison.

LIBRARY2 -- The name of the second library to be compared. Specify this for both file and library comparisons. If you are comparing libraries and intend to request source difference listings, this library should contain the "newer" source files.

LIBRARY2 has the same default value as LIBRARY1. To do a volume comparison, you must blank out this field. (See LIBRARY1.)

VOLUME2 -- The name of the second volume to be compared. This field must always be specified. It has the same default value as VOLUME1.

MODE -- The method to be used for file comparison, DISPLAY, Source Difference (SRCDIFF), or SUMMARY. You do not have to enter the entire mode name: D, SR, or SU is sufficient.

DISPLAY is the only acceptable value if you are comparing libraries or volumes. (If you combine file comparison with either of these processes, you can specify the mode for the latter before it begins.)

Library and volume comparisons are described in Section 8.4, "Comparing Libraries," and Section 8.5, "Comparing Volumes." The following section describes the file comparison process.

8.3 COMPARING FILES

COMPARE offers three modes or methods of file comparison. You specify your choice in the MODE field of the Input Specification screen. Briefly described, they have the following features:

Display -- Two files appear on a split-screen display where differing records are highlighted and the position of the first mismatched character is given. By using PF keys, you can examine and align the two files in the screen windows. Section 8.3.1 describes this file comparison mode.

Source Difference (SRCDIFF) -- This option is intended only for source files (consecutive files with 80-byte records). It provides a difference listing showing the lines that have been deleted from and inserted in the original file. The latter should be FILE1 and the new file should be FILE2. (If their positions are reversed, the deleted lines are labeled insertions and the inserted lines are labeled deletions.) Section 8.3.2 describes this mode of file comparison.

Summary -- The two files are simply compared. At the first sign of inequality (including file type, size, or record count), the comparison is terminated and you are informed that the files are not equal. A return code indicates the nature of the inequality. For a full description of this mode of file comparison, see Section 8.3.3.

8.3.1 File Comparison -- Display Mode

When you specify Display mode for a file comparison from the Input Specification screen, the File Options screen appears. Figure 8-2 shows an example of this screen.

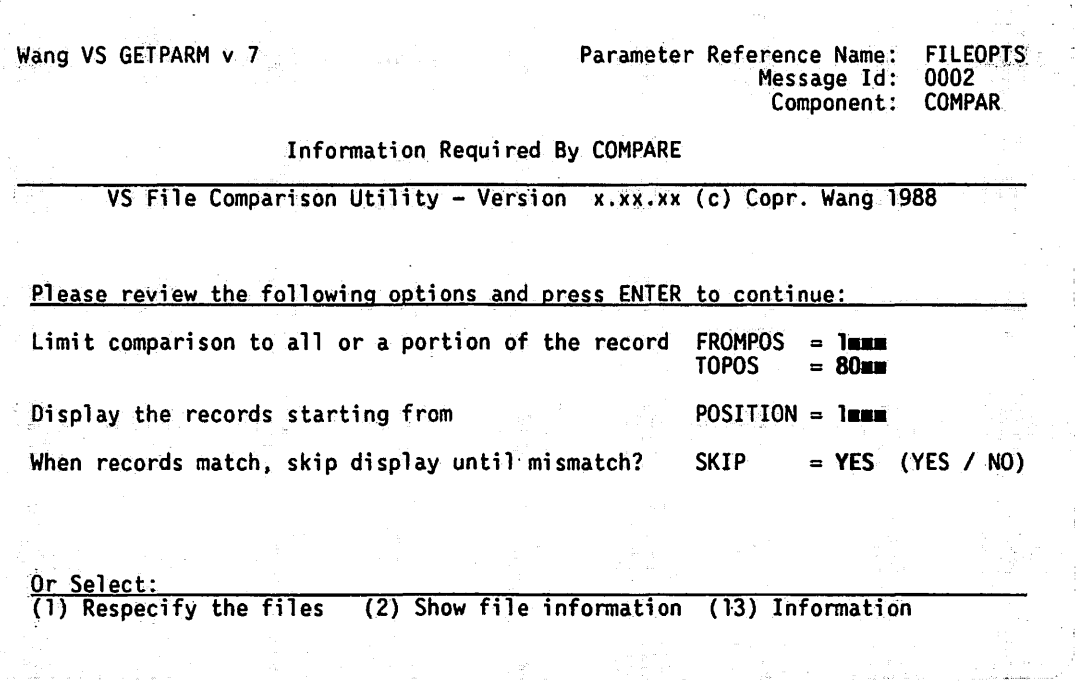


Figure 8-2. A Sample COMPARE File Options Screen

Note: *This screen also appears when you initiate a file comparison in Display mode from the Matching Files screen (Figure 8-11) during a library comparison. See Section 8.4.*

Review the default values shown in the fields of the File Options screen and enter any changes you want to make:

FROMPOS/TOPOS -- You can limit the comparison to a portion of each record in the files by specifying a starting and ending position in the record. The default is to compare the entire record, with FROMPOS set to 1 and TOPOS set to the maximum record size of FILE1. These values appear in the fields when the screen appears; you can change them at will. The program compares only the portion from FROMPOS through TOPOS in each record, according to the values specified.

The value of FROMPOS cannot be greater than the maximum record size of FILE1 or the value of TOPOS. The value of TOPOS cannot be greater than the maximum record size of FILE1 or less than the value of FROMPOS.

Note: *The two files you compare need not have the same record size, but the verifications for FROMPOS, TOPOS, and POSITION are based on the specifications for FILE1 only.*

POSITION -- This field allows you to specify the initial location of the display window in each file. The default is to display each file starting at record position 1.

SKIP -- Leaving the default YES in this field causes both files to be read without any display until a mismatch occurs. The files are then displayed in the windows starting with the first mismatched records. If the files are identical, the window shows only the last record and the message "End of file has been reached." If SKIP is set to NO, both files are displayed from the first specified records onward, even if all records match.

If you want to go back to the Input Specification screen to specify different files or another type of comparison, press PF1. You can also press PF13 for information about the options specified from this screen or PF2 to display information about each file.

If you press PF2 from the File Options screen, the File Information screen appears. Figure 8-3 shows an example of this screen.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  FILES
                                                          Message Id:  0003
                                                           Component:  COMPAR

                        Information Required by COMPARE

      VS File Comparison Utility - Version  x.xx.xx (c) Copr. Wang 1988

File #1    LXTEST     Library   CFJSRC     Volume    WORK       File Type    C
Rec Size     80       Records        77    Owner ID  CFJ        Rec Type     C
Crea Date  12/04/86   Mod Date  01/09/87   Exp Date  12/04/86   File Class
Blks Alloc     2      Blks Used      2     Extents      1

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

File #2    LXTEST     Library   CFJJUNK    Volume    WORK       File Type    C
Rec Size     80       Records        75    Owner ID  CFJ        Rec Type     C
Crea Date  12/27/86   Mod Date  01/09/87   Exp Date  12/27/86   File Class
Blks Alloc     2      Blks Used      2     Extents      1

                          Press ENTER to continue
```

Figure 8-3. A Sample COMPARE File Information Screen

Note: *This screen also appears if you press PF17 from the File Display screen (Figure 8-4) or PF2 from the Source Difference Options screen (Figure 8-7).*

The screen provides the following information about each of the files being compared:

- Name, library, and volume.
- File type:

C	Consecutive
I	Indexed
I+	Indexed plus
A	Alternate indexed
A+	Alternate indexed plus
O	Object
L	Log
R	Relative
P	Print
W	WP document

- Record size
- Number of records
- Record type:

F	Fixed length
V	Variable length
C	Compressed

- Creation, last modification, and expiration dates
- File protection class
- Blocks allocated to the file
- Blocks actually used by the file
- Extents used by the file

For an indexed file, the key position and length are also shown, and for an alternate indexed file, the number of alternate keys is shown.

After reviewing the information, press ENTER to return to the File Options screen.

When you press ENTER from the File Options screen, the File Display screen (Figure 8-4) appears. (Figure 8-4 shows an example of this screen.) It displays FILE1 in its upper window and FILE2 in its lower window.

```
      VS File Comparison Utility - Version  x.xx.xx (c) Copr. Wang 1986
File1:
Record   MM Col   1.........11........21........31........41........51........
    12        1   dcl readvtoc_message   (0:9)   char(8);
    13        5   DCL J                          CHAR(8);
    14        5   DCL Q(MAX_Q)                   CHAR(8);
    15        5   DCL SNUM                       CHAR(9)             STATIC;
    16        5   DCL CNUM                       CHAR(15)            STATIC;
    17        1   DCL REORG_MSG2                 CHAR(73)            INIT("***
-------------------------------------------------------------------------------
File2:
Record   MM Col   1.........11........21........31........41........51........
    12        1   DCL J                          CHAR(8);
    13        5   DCL Q(MAX_Q)                   CHAR(8);
    14        5   DCL SNUM                       CHAR(9)             STATIC;
    15        5   DCL CNUM                       CHAR(15)            STATIC;
    16        5   DCL REORG_MSG2                 CHAR(73)            INIT("***
    17        1       "NOT PROPERLY CLOSED - MAY REQUIRE REORGANIZATION ON RES

Press ENTER to compare the next six records in each file or select:
(1/↑1) Opts/Files  (5/↑5) Next      (9/↑9)   L Margin    (13) Information
(2/↑2) First       (6/↑6) Down      (10/↑10) R Margin    (14) Display in HEX
(3/↑3) Last        (7/↑7) Up        (11/↑11) Left  30    (15) Print Screen
(4/↑4) Prev        (8/↑8) Find      (12/↑12) Right 30    (16) Exit
```

Figure 8-4. A Sample COMPARE File Display Screen -- ASCII Mode

If you specified SKIP = YES from the File Options screen, the windows show the first mismatched records in both files, unless the files are identical; in this case the last record in each file is displayed. If you specified SKIP = NO, however, the windows show the first records in each file, whether or not these contain any mismatches.

Each window displays six lines, each of which shows 60 bytes of one record from the file. Two columns of information precede each line. The first column identifies each record by number; if the record does not match its counterpart in the other file, this number is highlighted. The second column shows the position of the first mismatched character in the record. (If the records match, this column contains no entry.)

All characters with hexadecimal values between X'80' and X'FF' are converted to periods in the ASCII display.

At the bottom of the screen appears a menu showing the functions you can command by using the ENTER and PF keys. They are described in the following list.

Note: *In the list below, the movement commands (PF keys 2 through 12) apply to the file in the top window if the SHIFT key is held down when the PF key is pressed, and to the file in the bottom window if it is not held down. You can thus move the two file displays independently.*

Because of the way the files are constructed, PF keys 3, 4, and 6 cannot be used when indexed or alternate indexed files are being displayed.

Key	Command
ENTER	If SKIP = YES, pressing ENTER causes both files to be read, starting *immediately after* the records displayed in the window, until the next mismatch is found. The latter is displayed at the top of each window, followed by the next five records of each file, if there are that many. If SKIP = NO, pressing ENTER moves the display to the next six records in each file.
PF1	Returns you to the File Options screen (Figure 8-2) so that you can change your selection of options. If you change them, the File Display screen, when you return to it, is updated accordingly, though no new records are read. This is a convenient way to move the display a long horizontal distance. For example, to see position 1000 of a 2000-byte record, set POSITION to 1000 on the File Options screen, and press ENTER. The File Display screen returns, with both files now positioned at column 1000. (This method is much easier than using PF12 to move to that position.) PF1 shifted (i.e., PF17) redisplays the File Information screen (Figure 8-3).
PF2	Displays the first six records of the file (if shifted, FILE1; otherwise FILE2; see the note above).
PF3	Displays the last six records of FILE1 (if shifted) or FILE2.
PF4	Displays the previous six records of either file.
PF5	Displays the next six records of either file.
PF6	Rolls the display of either file back one record.
PF7	Rolls the display of either file forward one record.
PF8	Enters Find mode (described below) for either file.
PF9	Positions the window to display the first 60 characters of the record for either file.

Key	Command
PF10	Positions the window to display the last 60 characters of the record for either file.
PF11	Moves the window 30 characters to the left in the record for either file (15 characters in hexadecimal Display mode).
PF12	Moves the window 30 characters to the right in the record for either file (15 characters in hexadecimal Display mode).
PF13	Displays information.
PF14	Switches between ASCII and hexadecimal Display modes. (The former is illustrated in Figure 8-4, the latter in Figure 8-5.) You can place the cursor under a character in either window, and, when you press PF14, the cursor is positioned under the same character in the alternate Display mode (if it is still on the screen, since the hexadecimal display shows only half the number of characters shown in the ASCII display).
PF15	Prints a copy of the current screen.
PF16	Terminates the file comparison display and presents the File Display End-of-Job (EOJ) screen.

```
      VS File Comparison Utility - Version  x.xx.xx (c) Copr. Wang 1986
File1:
Record   MM Col   1 . . . . 6 . . . . 11. . . . 16. . . . 21. . . . 26. . . .
    12       1   64636C2072656164746F635F6D65737361676520202028303A39292020
    13       5   44434C204A2020202020202020202020202020202020202020202020202020
    14       5   44434C2051284D41585F5129202020202020202020202020202020202020
    15       5   44434C20534E554D2020202020202020202020202020202020202020202020
    16       5   44434C20434E554D2020202020202020202020202020202020202020202020
    17       1   44434C2052454F52475F4D5347322020202020202020202020202020202020
-------------------------------------------------------------------------------
File2:
Record   MM Col   1 . . . . 6 . . . . 11. . . . 16. . . . 21. . . . 26. . . .
    12       1   44434C204A2020202020202020202020202020202020202020202020202020
    13       5   44434C2051284D41585F5129202020202020202020202020202020202020
    14       5   44434C20534E554D2020202020202020202020202020202020202020202020
    15       5   44434C20434E554D2020202020202020202020202020202020202020202020
    16       5   44434C2052454F52475F4D5347322020202020202020202020202020202020
    17       1   20202020224E4F542050524F5045524C5920434C4F534544202D204D4159

Press ENTER to compare the next six records in each file or select:
(1/†1) Opts/Files  (5/†5) Next       (9/†9)   L Margin    (13) Information
(2/†2) First       (6/†6) Down       (10/†10) R Margin    (14) Display in ASCII
(3/†3) Last        (7/†7) Up         (11/†11) Left  15    (15) Print Screen
(4/†4) Prev        (8/†8) Find       (12/†12) Right 15    (16) Exit
```

Figure 8-5. A Sample COMPARE File Display Screen -- Hexadecimal Mode

File Display Find Mode

When you press PF8, the display changes to Find mode. The bottom of the screen changes to display three search parameter fields in place of the menu. Figure 8-6 shows an example of such a screen.

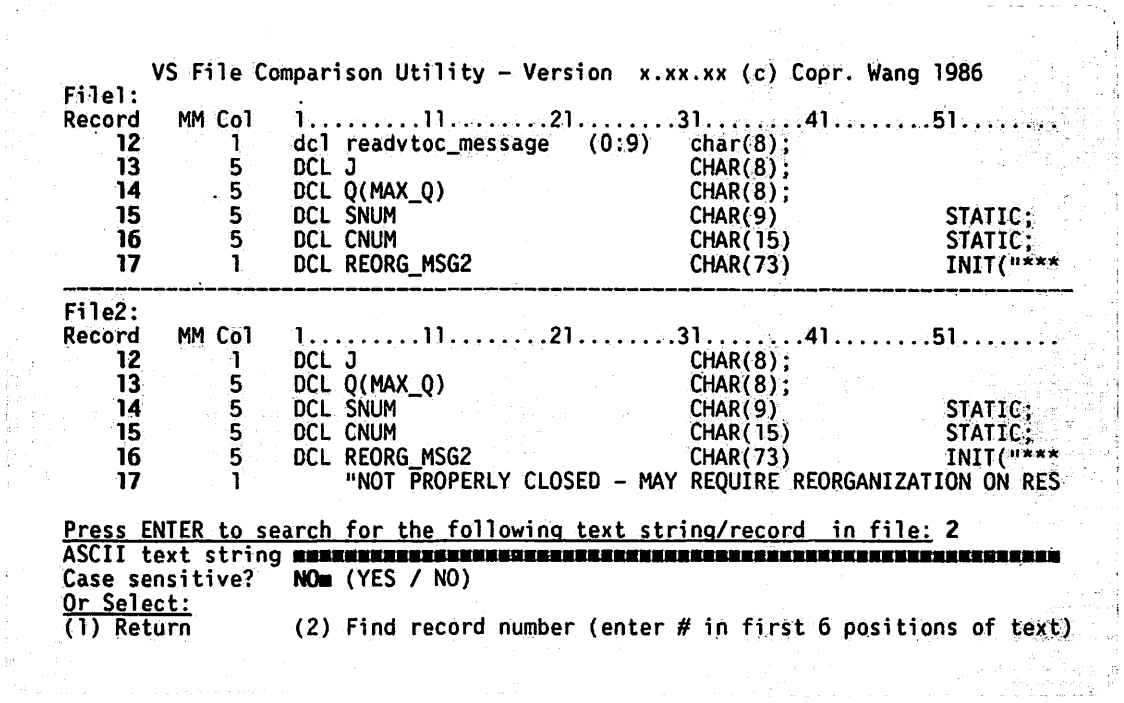


Figure 8-6. A Sample COMPARE File Display Screen -- Find Mode (ASCII)

If PF8 is shifted when you select Find Mode, the screen is set up for a search of File 1; if not, for a search of File 2. The number of the file you are searching appears in the field at the end of the first line below the windows. You can switch between File 1 and File 2 at will by changing this number.

To find a text string, enter it in the field labeled "ASCII text string." (If you pressed PF8 while in hexadecimal display mode, you are prompted to enter a string of hexadecimal values instead of ASCII characters.)

The third and last field determines case sensitivity for the search. The default is NO; that is, an uppercase letter is considered to match the same lowercase letter. (For example, *Smith*, *SMITH*, and *smith* are considered identical.) If you set this parameter to YES, two strings are considered mismatched unless corresponding letters are in the same case. (*Smith*, *SMITH*, and *smith* are considered different.)

After entering the text string (and other parameters if necessary), press ENTER. If the text is found, the appropriate record is positioned at the top of its display window and left-justified.

When one search is finished, you can press ENTER to find the next occurrence of the same string, or change any of the three search parameters (file, search string, and case sensitivity) and press ENTER to do a new search.

You also have these options from the File Display screen in Search mode:

PF Key	Option and Description
1	**Return** -- Cancels Find mode and returns the screen to its regular Display mode, with the original menu at the bottom, as in Figure 8-4.
2	**Find record number** -- To move the display directly to a different record in the same file, enter its number in the text string field (within the first six pseudoblanks) and press PF2.

Note: *PF2 is disabled so that you cannot move the display at will to a different record when you are searching (1) an indexed or alternate indexed file (in either Display mode), or (2) a file of any type in hexadecimal Display mode.*

When you finish displaying the files, press PF16 from the File Display screen. The File Display End-of-Job (EOJ) screen (not shown) appears. This screen identifies the two files that were compared and shows the record size and number of records for each.

- If you began by specifying a file comparison, you have two options: Pressing ENTER or PF16 to terminate COMPARE processing, or pressing PF1, which returns you to the Input Specification screen to begin a new file, library, or volume comparison.

- If, however, you initiated this file comparison from the Matching Files screen (Figure 8-11) when you were comparing libraries, pressing any of these keys returns you to that screen, which may list other files you want to compare.

8.3.2 File Comparison -- Source Difference Mode

The Source Difference mode of file comparison generates a difference listing file. This file contains all the lines that appear only in the newer file, labeled as insertions, and all the lines that appear only in the older file, labeled as deletions. The Source Difference mode is available only with files that contain 80-byte consecutive records -- provided that each file contains at least one and no more than 20,479 records.

When you specify this mode from the Input Specification screen (Figure 8-1), the Source Difference Options screen appears. Figure 8-7 shows an example of this screen.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  DIFFOPTS
                                                          Message Id:  0002
                                                           Component:  COMPAR

                        Information Required by COMPARE

        VS File Comparison Utility - Version  x.xx.xx (c) Copr. Wang 1988

 Please review the following options and press ENTER to continue:

 Select the language of the input file(s)          LANGUAGE = PL
 (ASSEMBLER, BASIC, CC, COBOL, FORTRAN, PLI, PROCEDURE, RPG)
 Or enter LANGUAGE = X to select your own
    start and end points for the comparison        FROMPOS  = 1
    (values automatically set for other languages) TOPOS    = 72

 Place comparison listings in LIBRARY  = #CFJDIFF on VOLUME   = WORK
 Replace duplicate names in listing library?       REPLACE  = YES  (YES / NO)
 List changed lines only (NO = all lines)?         LISTOPT  = YES  (YES / NO)

 Or select:
 (1) Respecify the files     (2) Show file information  (13) Information
```

Figure 8-7. A Sample COMPARE Source Difference Options Screen

Note: *This screen also appears immediately after the Library Options screen (Figure 8-9) if you have entered YES in the SRCDIFF field on that screen. See Section 8.2.3.*

Enter or change the values in the fields as appropriate.

LANGUAGE -- This option automatically sets the positions to be compared in the two source files; if you specify a language in this field, your entries in the FROMPOS and TOPOS fields are ignored. If you want to specify a different range than any of those shown in Table 8-1, enter X in this field and the first and last positions in the FROMPOS and TOPOS fields. Table 8-1 shows how to specify each supported language as well as the default settings of FROMPOS and TOPOS associated with each.

Table 8-1. COMPARE Language Specifications for Source Difference File Comparison

Language	Value To Specify[a]	Default Setting	
		FROMPOS	TOPOS
Assembly Language	**ASSEMBLER**	1	72
BASIC	**BASIC**	7	72
C	**CC**	1	72
COBOL	**COBOL**	7	72
FORTRAN	**FORTRAN**	1	72
PL/I	**PLI**	1	72
Procedure Language	**PROCEDURE**	1	71
RPG II	**RPG**	6	74

[a] The minimal necessary specification for each language is shown in bold type.

FROMPOS/TOPOS -- You can limit the comparison to a portion of the record by specifying the appropriate values in these fields and setting LANGUAGE to X. Set FROMPOS to the first byte position in the part of the record you want compared, and TOPOS to the last position. When LANGUAGE = X and no values are specified for FROMPOS or TOPOS, the default is to compare the entire record from position 1 through position 80. *If LANGUAGE is set to any other acceptable value, the values in these fields are ignored.*

The value of FROMPOS cannot be greater than 80 or the value of TOPOS. The value of TOPOS cannot be greater than 80 or less than the value of FROMPOS.

LIBRARY/VOLUME -- Use these fields to specify a library and volume to receive the Source Difference Listing file. You cannot put this file in the same library as either of the compared files, since it will have the same name as both (or, if the two files you are comparing have different names, the same name as File 1).

REPLACE -- If the value is YES (the default), COMPARE automatically scratches any identically named file in the library designated to receive the listing file. If you set REPLACE to NO, you may have to respond to an error message reporting that the file exists when the listing file is being written.

LISTOPT -- If the value in this field is the default YES, the source difference listing contains only the lines that differ between the two files. If it is set to NO, the source difference listing contains all the lines in both source files, including those they have in common. (Insertions and deletions are labeled in either case.)

When your specifications are complete, press ENTER. Alternatively, you can press PF1 to return to the Input Specification screen, or PF2 to display the File Information screen (Figure 8-3).

When you press ENTER from the Source Difference Options screen, COMPARE generates the source difference listing and displays the Source Difference EOJ screen. Figure 8-8 shows an example of this screen.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  EOJ
                                                          Message Id:  0009
                                                           Component:  COMPAR

                        Information Required by COMPARE

        VS File Comparison Utility - Version  x.xx.xx (c) Copr. Wang 1988

For:          File      Library    Volume    Record size     Number of records
    Input1:   LXTEST    CFJSRC     WORK          80                  77
    Input2:   LXTEST    CFJJUNK    WORK          80                  75


Press ENTER or (16) to exit COMPARE or select:
(1)  Compare two more files, libraries or volumes

(14) Display the SRCDIFF listing file LXTEST   in #CFJDIFF on WORK
```

Figure 8-8. A Sample COMPARE Source Difference EOJ Screen

Like other EOJ screens, this one identifies the compared files. It also gives you the following choices:

- Pressing PF14 invokes the DISPLAY utility to display the listing file. (The menu prompt for PF14 gives the name, library, and volume of this file.)

- If you began by specifying a file comparison, you have two other options -- pressing ENTER or PF16 to exit from COMPARE, or pressing PF1 to return to the Input Specification screen, where you can begin a new file, library, or volume comparison.

- If, however, you initiated this file comparison from the Matching Files screen (Figure 8-11) when you were comparing libraries, pressing ENTER, PF16, or PF1 returns you to that screen, which may list other files you want to compare.

8.3.3 File Comparison -- Summary Mode

In the Summary Mode of file comparison, COMPARE compares the files and displays a simple message telling you whether the specified files were equal.

When you select this mode from the Input Specification screen, the File Options screen (Figure 8-2) appears. The POSITION and SKIP fields are unavailable, since the file is not displayed in Summary mode, but you should specify the FROMPOS and TOPOS fields in the same way as for File Display mode. After you supply this information, press ENTER.

When the comparison is completed, the Summary EOJ screen (not shown) appears. It identifies the two compared files and gives the record size and number of records for each.

In addition, a message on the screen either reports that the files are identical, or describes the nature of any mismatch found in the comparison. COMPARE also generates a return code (accessible in any of the usual ways), which indicates the nature of the mismatch.[1] Table 8-2 shows the values for all COMPARE return codes.

You have two options from the Summary EOJ screen:

- Pressing ENTER or PF16 to exit from COMPARE.

- Pressing PF1 to return to the Input Specification screen, where you can begin a new file, library, or volume comparison.

[1] When file, library, or volume comparisons are done in series or combinations, the return code always reflects the last comparison done. In addition, a file comparison in File Display mode *always* ends with a return code of 0.

Table 8-2. COMPARE Return Codes

Return Code	Meaning
	File Comparison
0	Files match (at least within the part of each record examined, as set in the specifications).
4	Mismatch: at least one character is different.
8	Mismatch: one file has more records.
12	Mismatch: files have different record sizes.
16	Mismatch: files are of different types.
20	Could not complete comparison: unable to open a file.
24	Could not complete comparison: I/O error reading a file.
	Library Comparison
100	The files are common to both libraries and all match (at the level of comparison specified).
104	The files are common to both libraries, but do not all match.
108	Each library contains both common and exclusive files.
112	All files in LIBRARY2 are common to both, but LIBRARY1 has additional files.
116	All files in LIBRARY1 are common to both, but LIBRARY2 has additional files.
120	These two libraries have no files in common.
	Volume Comparison
200	The libraries are common to both volumes, and match in number of files.
204	The libraries are common to both volumes, but do not match in number of files.
208	Each volume contains both common and exclusive libraries.
212	All libraries on VOLUME2 are common to both, but VOLUME1 has additional libraries.
216	All libraries on VOLUME1 are common to both, but VOLUME2 has additional libraries.
220	These two volumes have no libraries in common.

8.4 COMPARING LIBRARIES

If you choose a library comparison from the Input Specification screen, the Library Options screen (Figure 8-9) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  LIBOPTS
                                                          Message Id:  0002
                                                           Component:  COMPAR

                        Information Required by COMPARE

        VS File Comparison Utility - Version  x.xx.xx (c) Copr. Wang 1988

 Please review the following library comparison options and press ENTER:

 Compare matched files?                                 COMPARE  = YES  (YES / NO)
 If COMPARE=YES,
    Perform a case sensitive comparison?                CASE     = YES  (YES / NO)
    Generate source difference listing for each pair
     of unlike 80 byte consecutive files with the
     same name (CASE has no effect on this option)?     SRCDIFF  = NO   (YES / NO)


 Or select:
 (1) Respecify the libraries                                  (13) Information
```

Figure 8-9. COMPARE Library Options Screen

Note: *This screen also appears if you choose a library comparison from the Matching Libraries screen (Figure 8-13) during a volume comparison.*

Use this screen to determine whether the contents of the files in the two libraries are compared, and how that comparison is done. Review the default values in the fields and enter changes as necessary.

COMPARE -- If COMPARE = YES (the default), identically named files in both libraries are compared record by record, provided they also match in type, record size, and number of records. If COMPARE is set to NO, only the file names are checked, and neither of the next two fields is significant.

CASE -- If CASE = YES (the default), the comparison is case-sensitive, i.e., records are not deemed to match unless upper- and lowercase letters match exactly. If CASE = NO, the records are read as if all letters were uppercase.

SRCDIFF -- If this field and COMPARE are both YES, COMPARE produces a source difference listing file for *each identically named but unlike pair of consecutive files containing 80-byte records*. You must choose options from the Source Difference Options screen, and COMPARE applies these to all files for which it makes listings. If, therefore, the libraries contain files that require different options, set this field to NO. You can later select individual pairs of files for source difference comparison and set the options for each pair in turn.

This option is specified from the Library Options screen rather than the Input Specification screen because library comparisons can be initiated during a volume comparison. This way you can choose the comparison mode for each matched pair of libraries rather than being limited to one mode for all.

Press PF13 for information about your options from this screen. When you have made any changes you want to make, press ENTER. If you want to specify different libraries (or volumes or files) to be compared, you can press PF1 instead to return to the Input Specification screen.

If you have set SRCDIFF to YES, the Source Difference Options screen (Figure 8-7) appears next. Specify the options in the same manner as for source difference file comparison (see Section 8.3.2). When you finish making specifications, press ENTER.

After you press ENTER from the Library Options screen or the Source Difference Options screen, the two libraries you specified are compared. If either contains more than 1,000 files, a warning screen appears with the message that only the first 1,000 will be processed. You can press ENTER to acknowledge this warning and continue, or PF1 to go back to the Input Specification screen and change your specifications.

At the end of the comparison process, the Library Comparison EOJ screen appears. Figure 8-10 shows an example of this screen.

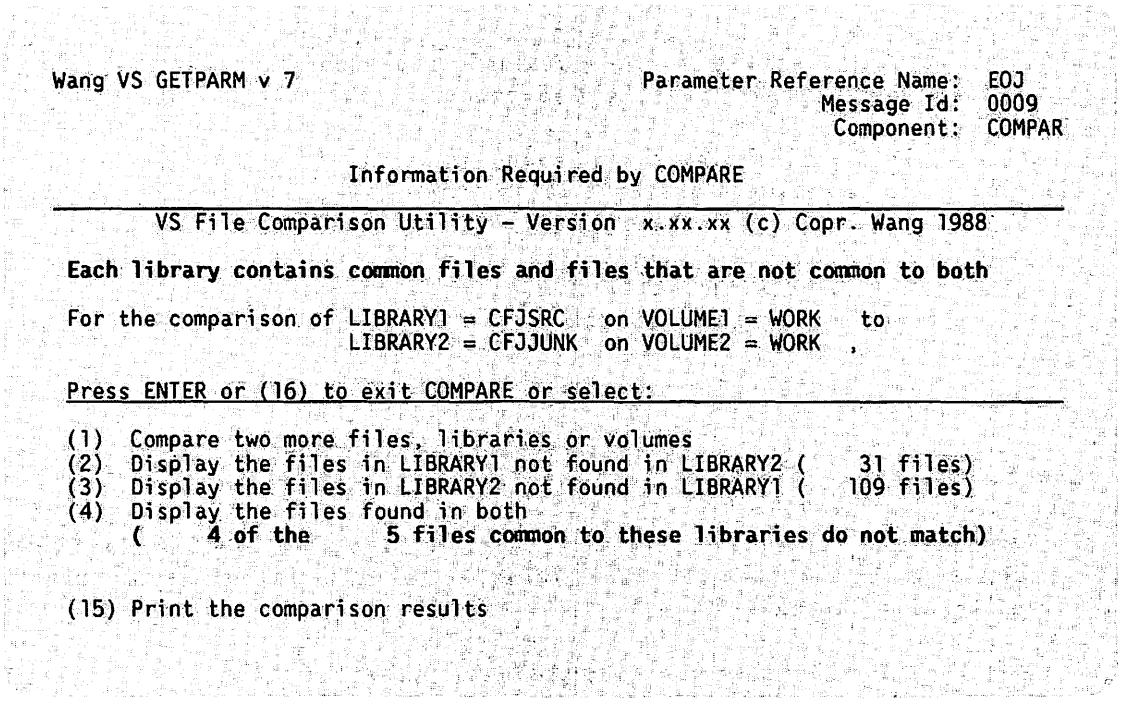


Figure 8-10. A Sample COMPARE Library Comparison EOJ Screen

This screen identifies the libraries that were compared, and offers a quick summary of the results. A line near the top of the screen tells whether the libraries are identical, and, if not, how they differ. If they differ, this line is highlighted.

Below the names of the two libraries that were compared, the screen displays the following options

- If you began by specifying a library comparison
  - Pressing ENTER or PF16 terminates COMPARE processing.
  - Pressing PF1 returns you to the Input Specification screen in order to begin a new file, library, or volume comparison.
- If you initiated this library comparison from the Matching Libraries screen (Figure 8-13) when you were comparing volumes, pressing ENTER, PF16, or PF1 returns you to that screen, which may list other libraries you want to compare.

- You can also press

  - *PF2* to show the files found in LIBRARY1 but not LIBRARY2. Their number is reported.

  - *PF3* to show the files found in LIBRARY2 but not LIBRARY1. Their number is also reported.

  - *PF4* to show the files found in both libraries. A message below this item on the list of options gives the number of files common to both libraries. If you specified file comparison, the message reports either that all common files match, or the number that do not match. (When some do not match, this message is highlighted.)

  - *PF15* to print the results of the comparison.

  (If no files belong to the category represented by PF2, PF3, or PF4, that key is inoperative and its text invisible.)

Pressing PF2 or PF3 produces a display of up to 1,000 file names, shown 60 at a time. The following PF keys can be used to examine or print the display:

PF2	Display the first part of the file name list
PF3	Display the last "page" of file names
PF4	Display the previous 48 file names
PF5	Display the next 48 file names
PF6	Scroll the display down one line
PF7	Scroll the display up one line
PF14	Display the file the cursor is on
PF15	Print a copy of the current screen
PF16	Return to the Library Comparison EOJ screen

A menu at the bottom of the display screen indicates these functions. A key is made inoperative and disappears from the menu when it can serve no valid purpose (e.g., PF2 through PF7 are all inoperative if less than 48 files were found in this particular category).

Pressing PF4 (if it is available) from the Library Comparison EOJ screen causes the Matching Files screen to appear. Figure 8-11 shows an example of this screen.

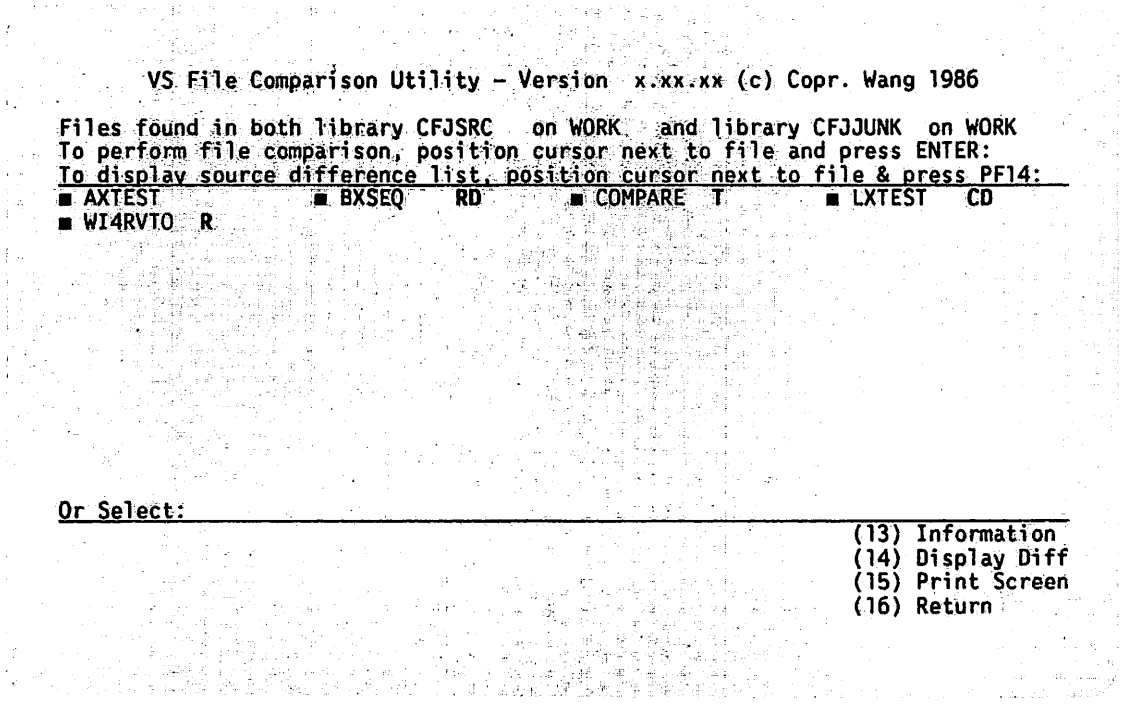


Figure 8-11. A Sample COMPARE Matching Files Screen

Up to 1,000 file names can be displayed, 48 at a time, in a format similar to that of the screens that show files found in one library only. However, this screen also contains further information if the files were compared individually (i.e., if COMPARE on the Library Options screen was YES).

If the files were identical, nothing appears after the file name. If they were not, the following codes, which appear to the right of the file names, indicate the nature of the mismatch:

R	The two files contain unequal numbers of records.
T	The files are of different types.
S	The record sizes are unequal.
C	The files agree in the three previous respects, but at least one character is mismatched.
RD	Records are unequal; a source difference listing was produced.
CD	Characters mismatch; a source difference listing was produced.

The following codes indicate processing errors during attempts to open, read or close the files (when COMPARE = YES):

O	One of the files could not be opened for comparison.
E	One of the files could not be read for comparison, or the file status when closed was unexpected.

Codes RD and CD occur only when SRCDIFF has been set to YES on the Library Options screen. If the attempt to generate a source difference listing fails because of a file processing error (e.g., I/O error or possession conflict), an asterisk (*) appears at the right of the file name and of any of these codes (e.g., O* or E*).

If any source difference listings have been successfully produced, the PF14 key is activated. Placing the cursor next to one of the files with a CD or RD code and pressing this key displays the listing for that file.

The following PF keys may be used with the Matching Files screen:

PF2	Display the first part of the file name list
PF3	Display the last "page" of file names
PF4	Display the previous 48 file names
PF5	Display the next 48 file names
PF6	Scroll the display down one line
PF7	Scroll the display up one line
PF13	Display information about this screen
PF14	Display source difference listing
PF15	Print a copy of the screen
PF16	Return to the Library Comparison EOJ screen

These keys are made inoperative and disappear from the menu when they have no valid purpose (e.g., PF2, PF4, and PF6 are inoperative while the beginning of the list is displayed).

If you position the cursor next to one of the displayed file names and press ENTER, you can initiate a file comparison. If both files are *not* consecutive files with 80-byte records, file display is the only mode available, and the File Option screen (Figure 8-2) appears when you press the ENTER key. Follow the steps described in Section 8.3.1. When you finish displaying the files and press PF16 from the File Display screen, the Matching Files screen reappears so that you can select another pair of files for comparison.

If both files are consecutive files with 80-byte records, you can choose either file display or Source Difference mode for the comparison. When you press ENTER, a screen (not shown) appears to offer you this choice. Enter D in the OPTION field for File Display mode, or S for Source Difference mode. Then press ENTER. Depending on your choice, either the File Option screen (Figure 8-2) or the Source Difference Option screen (Figure 8-8) appears. Follow the steps described in Section 8.3.1 or 8.3.2, as appropriate. When the comparison is complete, pressing PF16 returns you to this screen instead of terminating the program.

8.5 COMPARING VOLUMES

No options apply to a volume comparison. If you specify this type of comparison, it begins as soon as you press ENTER from the Input Specification screen. If either volume contains more than 1,000 libraries, a warning screen appears at this point with the message that, if you continue, only the first thousand will be processed. You can press ENTER to acknowledge the warning and continue, or PF1 to go back to the Input Specification screen and make new specifications.

When the comparison is finished, the Volume Comparison EOJ screen appears. Figure 8-12 shows an example of such a screen.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  EOJ
                                                          Message Id:  0009
                                                           Component:  COMPAR

                        Information Required by COMPARE

        VS File Comparison Utility - Version  x.xx.xx (c) Copr. Wang 1988

 Each volume contains common libraries & libraries that are not common to both

 For the comparison of VOLUME1 = PACE    to
                       VOLUME2 = SYSTEM

 Press ENTER or (16) to exit COMPARE or select:

 (1)  Compare two more files, libraries or volumes
 (2)  Display the libraries in VOLUME1 not found in VOLUME2 (    116 libraries)
 (3)  Display the libraries in VOLUME2 not found in VOLUME1 (    301 libraries)
 (4)  Display the libraries found in both
      (     10 of      16 common libraries do not have the same file count)


 (15) Print the comparison results
```

Figure 8-12. A Sample COMPARE Volume Comparison EOJ Screen

The screen identifies the volumes compared and reports whether their contents are identical, or, if they differ, how they differ. The message is highlighted in the latter case.

The Volume Comparison EOJ screen also offers the following options:

- Pressing ENTER or PF16 to terminate the program.
- Pressing PF1 to return to the Input Specification screen to begin a new file, library, or volume comparison.
- You can also press
  - *PF2* to show the libraries found in VOLUME1 but not VOLUME2. Their number is reported.
  - *PF3* to show the libraries found in VOLUME2 but not VOLUME1. Their number is also reported.
  - *PF4* to show the libraries found in both volumes. A highlighted message reports the number of libraries in this category, and tells how many of them, if any, differ in their file counts.
  - *PF15* to print the results of the comparison.

(If no libraries belong in the category represented by PF2, PF3, or PF4, that key is inoperative and its prompt invisible.)

Pressing PF2 or PF3 produces a display of up to 1,000 library names, shown 48 at a time. Each library name is followed by the number of files the library contains. The following PF keys can be used to examine or print the display:

PF2	Display the first part of the library list
PF3	Display the last "page" of library names
PF4	Display the previous 48 library names
PF5	Display the next 48 library names
PF6	Scroll the display down one line
PF7	Scroll the display up one line
PF15	Print a copy of the screen
PF16	Return to the Volume Comparison EOJ screen

A menu at the bottom of the screen indicates these functions. The keys are made inoperative and disappear from the menu when they do not serve a valid purpose (e.g., PF3, PF5, and PF6 are inoperative while the end of the list is being displayed.

When you press PF4 from the Volume Comparison EOJ screen, the Matching Libraries screen appears. Figure 8-13 shows an example of this screen.

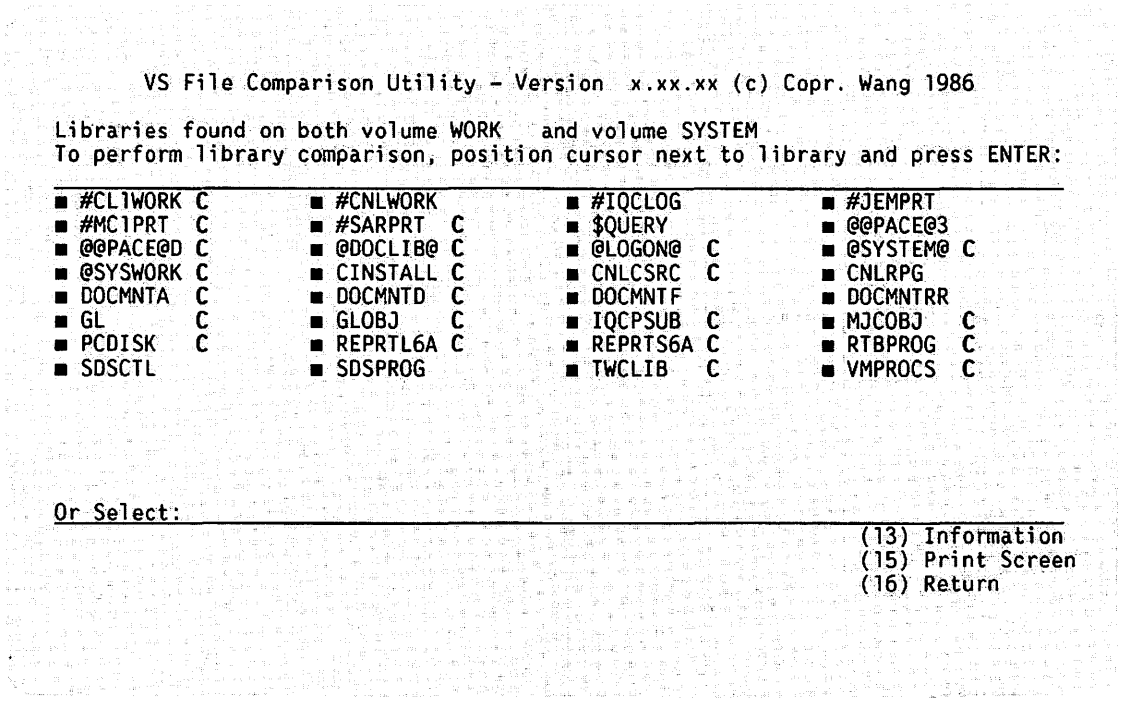


Figure 8-13. A Sample COMPARE Matching Libraries Screen

Up to 1,000 library names can be displayed in a format similar to that of the screens that show libraries in a single volume. The COMPARE Matching Libraries screen does not show the number of files, however, since each name refers to two libraries. Instead, the library name is followed by a highlighted C if both libraries do not contain the same number of files. The absence of this sign indicates that the number of files is the same for both libraries.

The following PF keys may be used with the Matching Libraries screen:

PF2 Display the first part of the library list
PF3 Display the last "page" of library names
PF4 Display the previous 48 library names
PF5 Display the next 48 library names
PF6 Scroll the display down one line
PF7 Scroll the display up one line
PF13 Display information about this screen
PF15 Print a copy of the screen
PF16 Return to the Volume Comparison EOJ screen

The PF keys are made inoperative and disappear from the menu when they have no valid purpose (e.g., PF2 through PF7 are inoperative if less than 48 libraries were found in this category).

If you place the cursor next to one of the displayed library names and press ENTER, the two libraries are compared. As the first step in this process, the Library Options screen (Figure 8-9) appears. During the library comparison, you can initiate any number of file comparisons. Follow the steps described in Section 8.4. When the library comparison is done, pressing PF16 returns you to this screen instead of terminating the program.

8.6 ON-LINE INFORMATION

Several screens in the COMPARE program offer the option of pressing PF13 for information. The latter is also available through VS INFO, if the appropriate help text has been installed. If this text cannot be located, a screen appears with instructions for installing it (see Figure 8-14).

```
Wang VS GETPARM v 7                          Parameter Reference Name:  NOHELP
                                                          Message Id:  0013
                                                           Component:  COMPAR

                        Information Required by COMPARE

        VS File Comparison Utility - Version  x.xx.xx (c) Copr. Wang 1988

 The COMPARE help text has not been installed in your INFO system.

 Please copy the file WI5HELP (supplied with the COMPARE utility) into either
 @DOCLIB@ or your alternate topic library, run INFO, and regenerate your
 your topic directory.




                              Press ENTER to return
```

Figure 8-14. COMPARE Help Text Information Screen

CHAPTER 9
CONDENSE

9.1 INTRODUCTION

The CONDENSE utility facilitates the extraction of information from a data file containing more than one type of record. Provided that each record type in the input file is described by a control file, CONDENSE can combine fields from records of different types into output records of a single type, creating both an output data file that contains these hybrid records, and a control file that describes them. The REPORT utility (which only accepts files with a single record type) can then extract from the output file information it could not access in the original data file. In the same way, the output file also becomes accessible to the other Data File Management utilities documented in Part I of this manual.

The VS Data File Management Utilities do not produce data files with multiple record types; such files may, however, be created by user-written applications programs. In order for CONDENSE to be useful, the records in these files, however they may vary in type, must be arranged in such a way that contiguous records have some relation to one another. In a file consisting of several types of personnel record, for example, the records for each employee must occur in a contiguous set, so that it makes sense to combine values from several consecutive input records into a single output record. Using letters to represent record types and numbers to represent employees, this organization can be illustrated as follows:

```
 A1, B1, C1, B1, C1, D1, B1, A2, D2, B2, C2, A3, B3, D3, A4, C4, ...
{           Set 1          }{     Set 2    }{   Set 3  }{   Set 4 ...
```

Provided the records occur in sets, some randomness within sets is permissible, but CONDENSE has no way to organize an input file whose records are distributed entirely at random (e.g., A1, B4, D1, A2, C3, etc.).

9.1.1 Overview

CONDENSE processing involves the following steps:

1. The user specifies

   - A control file for each record type to be incorporated in the output file.

   - Criteria by which CONDENSE can recognize this type of record.

   - Optionally, further criteria by which specific records can be selected or omitted.

   - The manner in which the records are to be read into the output file. (The way the available options are specified determines the structure of the output record.)

   - The fields to be included from each record.

   From these specifications, CONDENSE constructs a parameter file that governs the creation of the output file.

2. Once a parameter file is built, CONDENSE creates the condensed output data file according to the specifications in the parameter file. It also creates a corresponding control file.

3. CONDENSE also provides the following options:

   - Previously created parameter files can be modified.

   - The REPORT utility can be invoked to make a report on the output file.

   - Through a user exit subroutine, the user can define selection criteria more restrictive than those CONDENSE uses, modify the contents of the condensed data file, or both.

9.1.2 Software Requirements

CONDENSE requires a control file for each record type in the input data file. If these do not exist, they must be created before you can run CONDENSE on the file. Chapter 2 describes the use of the CONTROL utility to create control files.

9.1.3 Running CONDENSE

You can run CONDENSE by pressing PF1 (Run Program or Procedure) from the Command Processor menu. The following sections describe CONDENSE processing:

Section	Process
9.2	Specifying a Parameter File and Selecting Condense Options
9.3	Creating a Parameter File
9.3.1	Specifying Record Types
9.3.2	Selecting Fields From the Records
9.3.3	Examples of CONDENSE Record Specification
9.3.4	Specifying Input Data and Control Files
9.4	Modifying a Parameter File
9.5	Creating a Condensed Output Data File
9.6	Running the REPORT Utility on the Condensed File
9.7	Incorporating a User Exit Subroutine in CONDENSE Processing
9.7.1	Writing a User Exit Subroutine
9.7.2	Two Sample User Exit Subroutines

9.2 SPECIFYING A PARAMETER FILE AND SELECTING CONDENSE OPTIONS

When CONDENSE processing begins, the Parameter File Specification screen (Figure 9-1) is the first to appear.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  PARMFILE
                                                    Message Id:  M001
                                                     Component:  CONDEN

                    Information Required by CONDENSE

                    VS Condense File Utility - Version x.xx.xx

This program is used to combine fields from a file which contains multiple
record types into a single record type so that the file can be used as input
to another program.

Please specify the parameter file to be processed and press ENTER:

        PARMFILE = ________  PARMLIB  = CABCOND_  PARMVOL  = ______

Or Select:
                                                    (16) To exit CONDENSE
```

Figure 9-1. CONDENSE Parameter File Specification Screen

Use this screen to supply the name and location of the parameter file. This file contains all the information required to condense the input data file, with its multiple record types, into an output data file with a single record type that incorporates information from different types of output records. If the parameter file does not yet exist, provide a name for it; the following sections tell how to supply the necessary information.

Enter the requested information as follows:

PARMFILE -- Supply a valid file name for the new parameter file you intend to create, or the name of an existing parameter file you wish to use or modify.

PARMLIB -- CONDENSE provides a default library name by concatenating your user ID with the string COND, e.g., USRCOND.

PARMVOL -- If the INVOL field of your usage constants is set, the value from this field appears here as a default.

Enter or modify the information in the fields as appropriate, and press ENTER to continue. Alternatively, you can press PF16 to exit from the program.

When you press ENTER from the Parameter File Specification screen, the CONDENSE Function menu (Figure 9-2) appears.

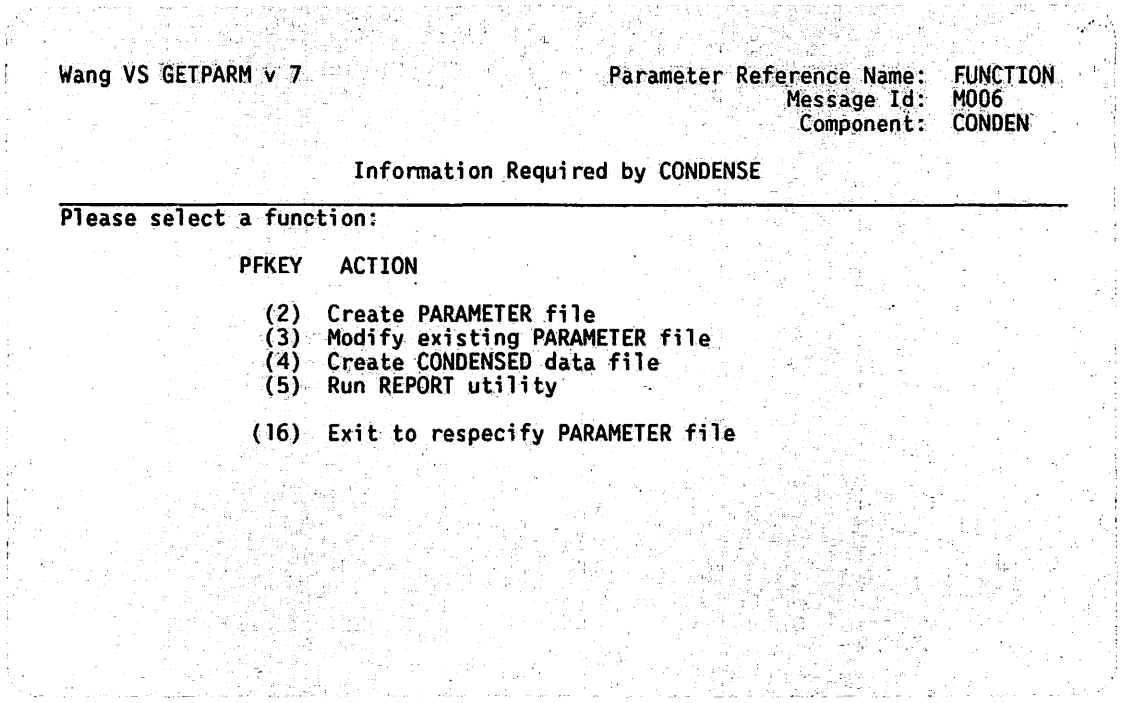


Figure 9-2. CONDENSE Function Menu

All CONDENSE functions are selected from this menu. You have the following options:

PF Key	Option and Description
2	**Create parameter file** -- CONDENSE prompts you for information and uses it to make a new parameter file. See Section 9.3.
3	**Modify existing parameter file** -- CONDENSE displays the specifications for the parameter file you named and gives you the opportunity to replace them. See Section 9.4.

PF Key	Option and Description
4	**Create condensed data file** -- Using the specified parameter and input files, CONDENSE constructs the output data file. See Section 9.5.
5	**Run REPORT utility** -- CONDENSE links to REPORT so that you can extract information from the condensed data file. See Section 9.6.
16	**Exit to respecify parameter file** -- PF16 returns you to the Parameter File Specification screen (Figure 9-1), from which you can specify another parameter file or exit from CONDENSE.

9.3 CREATING A PARAMETER FILE

9.3.1 Specifying Record Types

When you press PF2 from the CONDENSE Function menu, the Record Type Specification screen (Figure 9-3) appears.

```
                    Wang VS file management utility - CONDENSE

  Please enter the name of a control file and describe the record type in
your file that this control file defines

          File = ■■■■■■■■   Library = USRCTL■■   Volume = ■■■■■■

        Byte     Rel.                          Byte     Rel.
      Location   Oper. Char. Conn.           Location   Oper. Char. Conn.
      --------   ----- ----- -----           --------   ----- ----- -----
  (1) ■■■■       ■■    ■■    ■■■       (2)   ■■■■       ■■    ■■    ■■■
  (3) ■■■■       ■■    ■■    ■■■       (4)   ■■■■       ■■    ■■    ■■■
  (5) ■■■■       ■■    ■■    ■■■       (6)   ■■■■       ■■    ■■    ■■■
  (7) ■■■■       ■■    ■■    ■■■       (8)   ■■■■       ■■    ■■    ■■■
  (9) ■■■■       ■■    ■■    ■■■       (10)  ■■■■       ■■    ■■
  (Location: 1-2024; Relation: EQ,NE,GT,GE,LT,LE; Connector: AND,OR,)
             (See PF14 screen for additional information)

    Specify output record options when processing this record type
              WRITE = NO■■■■  (Options: NO,BEFORE,AFTER)
              RESET = NO■■■■  (Options: NO,BEFORE,AFTER)

        Press PF14 for additional relation/option descriptions
                  Press PF16 to exit record definition
```

Figure 9-3. CONDENSE Record Type Specification Screen

You specify only one record type from each specification screen. CONDENSE displays a succession of screens for you to specify more record types, until you indicate by pressing PF16 that the record definition process is completed. The Record Type Specification screen displays three sets of fields for different kinds of information. The next three sections describe them in detail.

You also have these options from the Record Type Specification screen:

PF Key	Option and Description
14	**Additional relation and option descriptions** -- Pressing PF14 causes CONDENSE to display a screen with information on how to specify recognition criteria and write and reset options. See the next three sections.
16	**Exit record definition** -- Press PF16 when you have no more record types to specify, or if you decide not to specify any. The Data Files Specification screen (Figure 9-8) appears.

Specifying the Control File for the Record Type

In the upper part of the screen are three fields in which to identify the control file that describes the specified type of record. As previously noted, an existing control file that describes each record type is a prerequisite for CONDENSE processing; the values you enter in these fields must enable the program to locate this file.

FILE -- Enter the name of the control file.

LIBRARY -- A default control library of the form USRCTL (where USR represents your user ID) is supplied. Change this value as necessary if your control file is in a different library.

VOLUME -- If the INVOL field of your usage constants is set, the value from this field appears as the default volume. Change the value if your control file resides elsewhere.

Specifying Recognition Criteria for the Record

The middle part of the screen contains a set of 39 fields in which to specify criteria that will enable CONDENSE to recognize the record type. The criteria you enter here constitute the record type specification as far as CONDENSE is concerned, since the program has no way to recognize a particular type except by comparing each record to these specifications as it is read.

Each criterion consists of one byte from the record that is compared to a specified value. It is your responsibility to identify a unique combination of bytes such that the comparison criteria you specify will be true only for the record type you are specifying. If each record in your input file contains an identifying "flag byte" with a value unique to its type, the task is simple; otherwise it may be necessary to compare the records closely and to define several criteria for each in order to provide a distinctive "signature" that CONDENSE can recognize. As many as 10 separate criteria can be entered for a single record on the Record Type Specification screen.

Once the purpose of identifying the record type is achieved, you have the option of using any remaining criteria up to the maximum of 10 for the purpose of selecting records by value. For example, knowing that values in numeric fields are right-justified, you could specify that the value of a certain byte in a numeric field be greater than or equal to 5. By doing so, you would limit the output file to records containing 5 or more, 50 or more, 500 or more, etc., in this field.

However, criteria designed to limit the record selection by value should be defined only *after you have defined criteria that make the identification of the record type unmistakable.*

Each criterion is defined by means of three fields and linked to the criterion that follows it (if any) by means of a logical connector entered in a fourth field. The fields are arranged as follows:

Byte Location -- The position in the input record of the byte CONDENSE is to test. The value must be a number between 1 and the specified length of the record.

Relational Operator -- The relationship that must exist between the value in the field and the value specified for comparison in order to satisfy the criterion. Valid relational operators are

EQ	Equal to
NE	Not equal to
GT	Greater than
GE	Greater than or equal to
LT	Less than
LE	Less than or equal to

Character -- The value to which the value in the specified byte location is to be compared according to the relational operator. Two pseudoblanks are provided. If you enter a single character (which must be in the left position), the comparison value is interpreted as an ASCII character. If you enter two characters (which must be valid hexadecimal digits), it is interpreted as a hexadecimal value.

Logical Connector -- A logical relationship to link multiple criteria. Acceptable values are AND and OR. If you specify AND, the test is positive when both linked criteria are true; if you specify OR, it is positive if either criterion is true. Up to 10 criteria can be linked for a given record type. All AND relations are evaluated first. Each AND result is then treated as a single criterion to be used by the OR operators. Consider the following example, in which each bold letter represents a simple criterion such as Byte 23 EQ C, Byte 118 LE 05, etc.:

Original formulation:
A AND B AND C OR C AND D OR E

After evaluation of ANDs:
ABC OR CD OR E

This field does not appear on the screen after the three fields that define the tenth criterion.

Specifying Write and Reset Options for the Record

In the lower part of the Record Type Specification screen are two fields in which to specify the Write and Reset options for the record type. The values you enter in these fields control the way the output record is constructed.

To understand the way these options operate, it is necessary to understand a little about the way CONDENSE processes records. The records are not read directly into the output file, but rather into a buffer internal to the program. This buffer reserves a place for the value from each selected field from each specified record type. As CONDENSE reads one of these records, it reads the values from the selected fields into the appropriate places in the buffer. CONDENSE neither writes the buffer contents to the output file nor clears them from the buffer unless specifically told to do so by means of the options specified in these two fields.

Write -- This field controls the writing of the buffer contents to the output file when CONDENSE processes the specified record type. The available options are as follows:

NO -- The buffer contents are not written to the output file when this type of record is processed. The values in the selected fields from the current record are written to the appropriate locations in the buffer, overwriting any information already in these locations.

BEFORE -- The buffer contents are written to the output file *before* the values in the selected fields from the current record are written to the appropriate locations in the buffer.

AFTER -- The buffer contents are written to the output file *after* the values in the selected fields from the current record are written in the appropriate locations in the buffer.

Reset -- This field controls the clearing and reinitialization of the entire buffer. When values are transferred into the buffer from selected fields of a particular record type, they remain until overwritten by new values when another record of the same type is read, or until the buffer is cleared by means of the option specified in this field. The *entire* buffer is cleared, not just the part allocated to fields from this record type. The available options are as follows:

NO -- The buffer is not cleared when this record type is processed. The values in the selected fields from the current record are transferred to the appropriate locations in the buffer, overwriting any information already in these locations.

BEFORE -- The buffer is cleared *before* the values from the selected fields of the current record are transferred into it. When CONDENSE reads the next record, the buffer is empty except for these values.

AFTER -- The buffer is cleared *after* the values from the selected fields of the current record are transferred into it. When CONDENSE reads the next record, the buffer is empty.

As CONDENSE reads through the input records, it always processes write and reset specifications in the following order:

1. WRITE BEFORE
2. RESET BEFORE
3. Write contents of selected fields to the buffer
4. WRITE AFTER
5. RESET AFTER

Consequently, if a write and a reset are *both* specified either before or after the record is transferred to the buffer (steps 1 and 2 or 4 and 5), the write always takes place before the reset. Combinations of WRITE BEFORE and RESET AFTER and vice versa (steps 1 and 5 or 2 and 4) are also permitted, and are processed in the same order (i.e., the lower-numbered step is done first). For most applications the write and reset specifications will be identical, but for some unusual applications they may be different. The combination of WRITE AFTER and RESET BEFORE, for example, writes *only the values from the selected fields in the current record* to the output file, all other values in the buffer having been cleared before these were transferred into it.

It is by selecting the correct combination of write and reset specifications that you control the structure of the output file. The examples in Section 9.3.3 illustrate some of the ways in which this can be done. (The examples refer not only to write and reset specifications but also to certain kinds of fields described in the next section.)

9.3.2 Selecting Fields From the Records

As soon as you complete your specifications from a Record Type Specification screen and press ENTER, the corresponding Field Selection screen appears. Figure 9-4 shows an example of this screen.

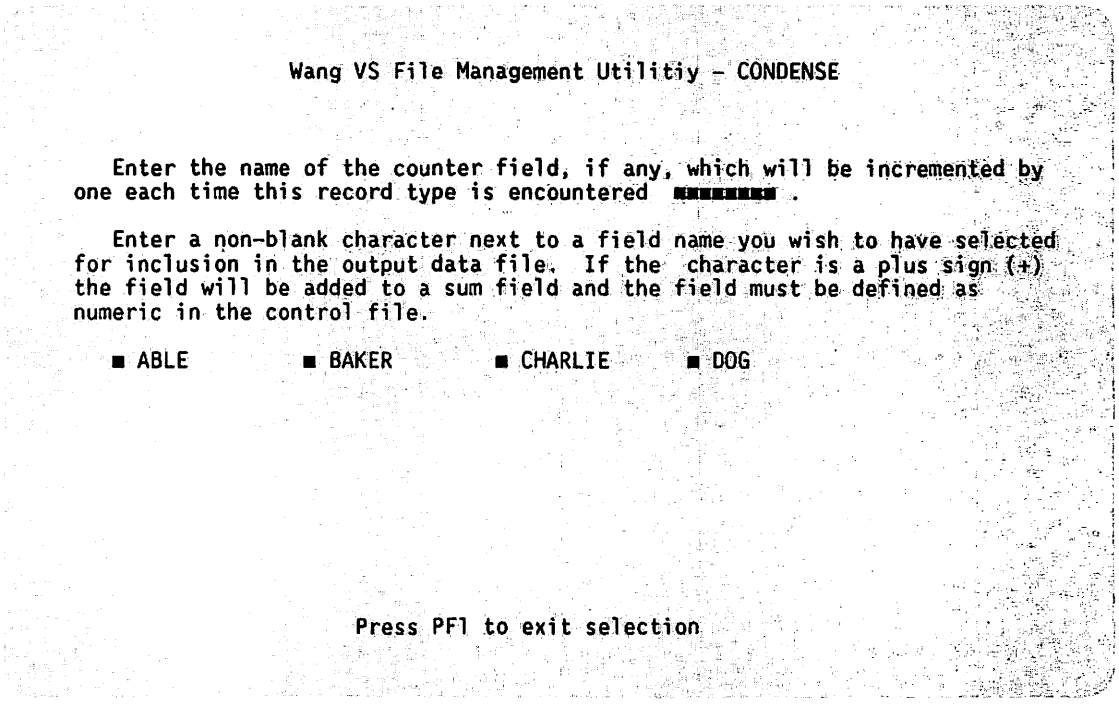


Figure 9-4. A Sample CONDENSE Field Selection Screen

CONDENSE obtains the names of the fields from the control file you just specified and displays them in alphabetical order on this screen. To select a field for inclusion in the output record, you need only enter any character, other than a blank or a plus sign, in place of the pseudoblank before the name of the field. (As explained below, a plus sign has a special meaning and should not be used unless you intend to define a sum field.)

CONDENSE places the fields you select in the buffer in alphabetical order within the space allotted for this record type. (The same order is used when the buffer is written to the output file.) Since the fields you select will eventually be described in a single control file constructed for the output file records, they must have unique names. If you find it necessary to include two fields from different records that have the same name, you will have to run the CONTROL utility to change the name of one of these fields before you can run CONDENSE.

Besides simply selecting fields to be included, you can define two additional types of fields from the Field Selection screen: *counter fields* and *sum fields*.

Counter fields -- A counter field accumulates the total number of input records of this type encountered since the buffer was last reset. When a reset takes place, the value in this field is set to 0; thereafter, it is incremented by 1 each time CONDENSE reads a record of the specified type from the input file. To define a counter field for this record type, enter any unique and valid field name in the blank field provided for this purpose at the top of the screen. CONDENSE creates a field four bytes in length (COBOL specification PIC S9(7) COMP) in the output file under the name you have provided.

Sum fields -- A sum field accumulates the values of a specific numeric field in all input records of this type encountered since the buffer was last reset. When a reset takes place, the value in this field is set to 0; thereafter, it is incremented by whatever value is contained in this field each time a record of this type is read from the input file. To define a sum field, enter a plus sign (+) next to the name of the field you want summed. You can define more than one sum field per record type; there can be as many sum fields as there are numeric fields in the input record. (Note that a field cannot be both summed and included; i.e., CONDENSE cannot place both the individual values from specific records and the accumulated values from all records for the same field in the output record. A sum field retains the same name it has in the control file and cannot be renamed during CONDENSE processing.) For each sum field, CONDENSE creates a field eight bytes in length (COBOL specification PIC S(15) COMP) in the output file under the original field name.

You must select at least one field from those listed, even if you also define a counter field. The field you select can be a sum field.

When you finish selecting and defining fields, press ENTER. The next Record Type Specification screen appears, and you can specify another record type and select fields from it by repeating the procedure described in these sections.

If you have specified all the record types for this file, press PF16 from the blank Record Type Specification screen to continue processing from the Data File Specification screen.

Alternatively, you can press PF1 to exit from the Field Selection screen without selecting any fields. Since you cannot specify a field without selecting at least one record from it, however, this action invalidates your specifications from the Record Type Specification screen, which is blank when it reappears.

9.3.3 Examples of CONDENSE Record Specification

The three examples that follow both make use of a data file that contains multiple record types involving vendor accounts. (As explained previously, CONDENSE is useful because the records in this file are arranged in sets -- that is, all records having to do with a given vendor are placed contiguously in the file.) The records are of three types:

A -- A header record containing a single vendor's name and number. There is only one record for each vendor and it comes at the beginning of the set.

B -- A record of such information as the vendor's address, telephone number, name of representative to contact, etc. Records of this type are entered whenever any of this information happens to change. Their number varies and they may occur anywhere in the sequence of the set except at the head.

C -- A record of a single transaction with the vendor, containing the item name, number, and price. A record of this type is entered for each individual item purchased whenever a purchase is made. They vary in number, are usually numerous, and occur anywhere in the set except at the head.

Example 1

The first example assumes that you want to create a file that summarizes the contents of the input file in the form of a single record for each vendor. It must contain the following information:

- Vendor name and number from record type A.
- Other vendor information (address, telephone, contact person) from record type B -- but only the most recent information; that is, only what is contained in the last record of this type in the sequence.
- The number of items purchased from this vendor and the total amount spent -- from sum and counter fields created for record type C.

Figure 9-5 illustrates the way this is done with CONDENSE. The contents of a simplified input file containing 19 records are listed first, followed by the settings of the Write and Reset fields used to achieve the desired result. Finally, the contents of the buffer (and its sum and counter fields in particular) as well as the output file are represented symbolically at each stage of processing (i.e., as each record is read). The symbol for each record (A_1, B_2) represents the presence of the values from its specified fields in the CONDENSE buffer. A dollar sign and an increasing number of plus signs represents the accumulation of values in the sum field. The contents of a record in the output file are identical with the contents of the CONDENSE buffer immediately before the record was written.

Input File Contents:

A_1, B_1, C_1, C_2, C_3, B_2, C_4
A_2, B_3, C_5
A_3, B_4, C_6, C_7, B_5, C_8, C_9, C_{10}, B_6, C_{11}

Write and Reset Field Settings:

Type	RESET	WRITE
A	BEFORE	BEFORE
B	NO	NO
C	NO	NO

Buffer Contents	Sum Field	Counter Field	Output File
(WRITE) →			Record 1
(RESET)			
A_1			
A_1 B_1			
A_1 B_1 C_1	$	1	
A_1 B_1 C_2	$+	2	
A_1 B_1 C_3	$++	3	
A_1 B_2 C_3	$++	3	
A_1 B_2 C_4	$+++	4	
(WRITE) →			Record 2
(RESET)			
A_2			
A_2 B_3			
A_2 B_3 C_5	$	1	
(WRITE) →			Record 3
(RESET)			
A_3			
A_3 B_4			
A_3 B_4 C_6	$	1	
A_3 B_4 C_7	$+	2	
A_3 B_5 C_7	$+	2	
A_3 B_5 C_8	$++	3	
A_3 B_5 C_9	$+++	4	
A_3 B_5 C_{10}	$++++	5	
A_3 B_6 C_{10}	$++++	5	
A_3 B_6 C_{11}	$+++++	6	

End of processing.

Figure 9-5. CONDENSE Processing -- Example 1

Commentary on Example 1

As it reads the records in sequence, CONDENSE first stores the values of the fields selected from record type A in the appropriate part of the buffer. There they remain unchanged, since by definition no other records of type A occur in the set. As the program reads successive records of type B, it keeps replacing the values in that part of the buffer, until the values taken from the last record of type B remain. As each successive record of type C is read, the counter field is incremented by 1, and the sum field is incremented by the value in the field in the record that stores the cost of the item purchased.

When CONDENSE reaches the end of a set, the buffer contains the vendor's name and number from the first record (type A); other vendor information from the last record of type B in the sequence, and a record of the number of items purchased from that vendor (the counter field) and the total cost (the sum field), both derived in the process of reading all the records of type C.

At this point, CONDENSE encounters the header record of the next set, which is type A. Before it places this record in the buffer, the program

1. Writes the buffer contents to the input file, creating a record that contains the information summarized in the previous paragraph

2. Resets the buffer, clearing it of all values from the set of records just read

Only then is the type A record (for the next vendor) placed in the buffer so that the process described in these paragraphs can begin again.

Example 2

The method used in Example 1 substantially accomplishes the task of summarizing the data in the file. However, it works imperfectly at each end of the process. First, CONDENSE, having been commanded to write and then reset the buffer before reading in a record of type A, writes an empty record at the beginning of the output file, since the first input record it encounters is invariably of type A, and the buffer contains no information at that point. A less trivial problem occurs at the end of the input file -- because there is no type A record at the end, the buffer contents (representing the information from the last set of records) are not written to the output file.

The optimum solution to these problems involves altering the structure of the input file. If possible, the application that creates the input file should be designed to insert a trailer record at the end of each set of records in the file, immediately before the header record that marks the beginning of the next set. The trailer record has no function except to mark the end of the set and to trigger the writing and resetting of the buffer.

Figure 9-6 illustrates this solution.

Input File Contents:

A_1, B_1, C_1, C_2, C_3, B_2, C_4, D_1
A_2, B_3, C_5, D_2
A_3, B_4, C_6, C_7, B_5, C_8, C_9, C_{10}, B_6, C_{11}, D_3

Write and Reset Field Settings:

Type	RESET	WRITE
A	NO	NO
B	NO	NO
C	NO	NO
D	AFTER	BEFORE

Buffer Contents	Sum Field	Counter Field	Output File
A_1			
A_1 B_1			
A_1 B_1 C_1	$	1	
A_1 B_1 C_2	$+	2	
A_1 B_1 C_3	$++	3	
A_1 B_2 C_3	$++	3	
A_1 B_2 C_4	$+++	4	
(WRITE) →			Record 1
A1 B2 C4 D_1	$+++	4	
(RESET)			
A_2			
A_2 B_3			
A_2 B_3 C_5	$	1	
(WRITE) →			Record 2
A_2 B_3 C_5 D_2	$	1	
(RESET)			
A_3			
A_3 B_4			
A_3 B_4 C_6	$	1	
A_3 B_4 C_7	$+	2	
A_3 B_5 C_7	$+	2	
A_3 B_5 C_8	$++	3	
A_3 B_5 C_9	$+++	4	
A_3 B_5 C_{10}	$++++	5	
A_3 B_6 C_{10}	$++++	5	
A_3 B_6 C_{11}	$+++++	6	
(WRITE) →			Record 3
A_3 B_6 C_{11} D_3	$+++++	6	
(RESET)			
End of processing.			

Figure 9-6. CONDENSE Processing -- Example 2

Commentary on Example 2

The buffer is filled with information from the first set of input records before the first output record is written; thus there is no empty record at the beginning of the file. When CONDENSE encounters a trailer record, it first writes the buffer contents to the output file, adds the type D record to the buffer, and then clears the buffer. (Although you must specify a field from record type D to be included in the output file, you can use the Write and Reset field settings shown here to keep this field from being written to the file. Type D records serve only as markers and contain no significant information.)

Sometimes, however, an input file has no trailer records, and it is not feasible to insert them. In this situation you can avoid losing the last output record if you simply append an extra or dummy type A record to the end of the input file. Set the Write and Reset fields as in Example 1 (i.e., both are set to BEFORE for type A and to NO for types B and C). When CONDENSE encounters this record, it writes and resets the buffer, thus insuring that the last record derived from the input file is written to the output file (and leaving the dummy type A record in the buffer when processing is completed. (Note that this method does not prevent a blank record from being written at the beginning of the output file.)

Example 3

Suppose that you want to extract from the same input file used in Example 1 an inventory of all purchases made from each vendor. Clearly, this makes it necessary to get all the records of type C into the output file.

Because records of type B and type C are mixed in sequence, it is impossible to be sure of getting only the latest information from type B records. Whenever a type C record is encountered, it must be written to the output file, whether or not it comes before the last type B record in the set. It is best, therefore, to omit type B from the specification for this run of CONDENSE. (If the input data file were constructed so that all records of type B occurred before any records of type C were encountered, this restriction would not be necessary.)

To create output records for this example, you can select the name and number fields from type A, and the fields that record details of an individual purchase (item name and number, cost, date, etc.) from type C. Figure 9-7 illustrates the method used for writing the records to the output file.

Input File Contents:

A_1, B_1, C_1, C_2, C_3, B_2, C_4
A_2, B_3, C_5
A_3, B_4, C_6, C_7, B_5, C_8, C_9, C_{10}, B_6, C_{11}

In this example, record type B is not specified and is ignored by CONDENSE.

Write and Reset Field Settings:

Type	RESET	WRITE
A	NO	NO
B	NO	NO
C	NO	AFTER

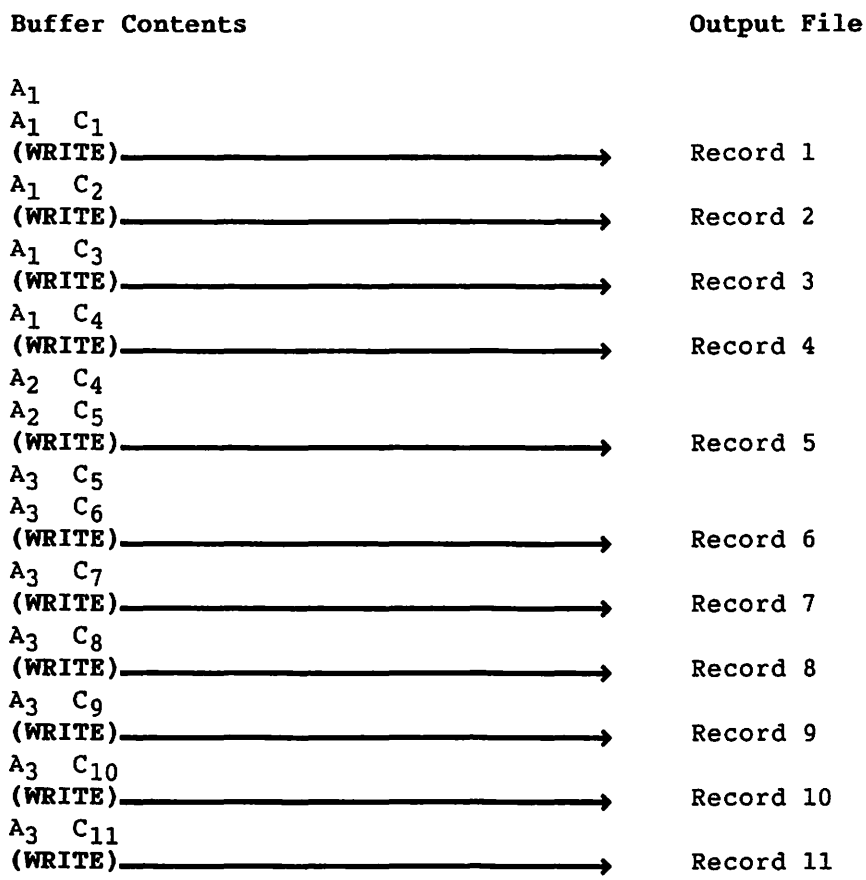


Figure 9-7. CONDENSE Processing -- Example 3

Commentary on Example 3

Whenever it encounters a type C record, CONDENSE reads this record into the buffer and then writes the entire record -- containing the vendor information from the type A header record and the individual purchase information from the type C record -- to the output file. In order to make sure that each purchase record is associated with the vendor's name and number, the buffer is not reset after the record is written. (If it were reset, only the first output record from each set would receive this information.)

As soon as CONDENSE encounters the next set, the values from the type A record at the head of that set replace those held over in the buffer from the previous set. Similarly, the type C values that remain in the buffer are replaced when CONDENSE encounters the first type C record in the new set, and the output record is not written until after this replacement occurs. The buffer, therefore, never needs to be reset.

In this example there is no problem with empty records at the beginning of the output file or lost records at the end. The first record is written after the necessary information is present in the buffer, and the last record passes the values from the last type A and type C records in the input file.

9.3.4 Specifying Input Data and Control Files

When you press PF16 from a Record Type Specification screen (Figure 9-3) to indicate that you have no more record types to specify, CONDENSE displays the Data Files Specification screen (Figure 9-8).

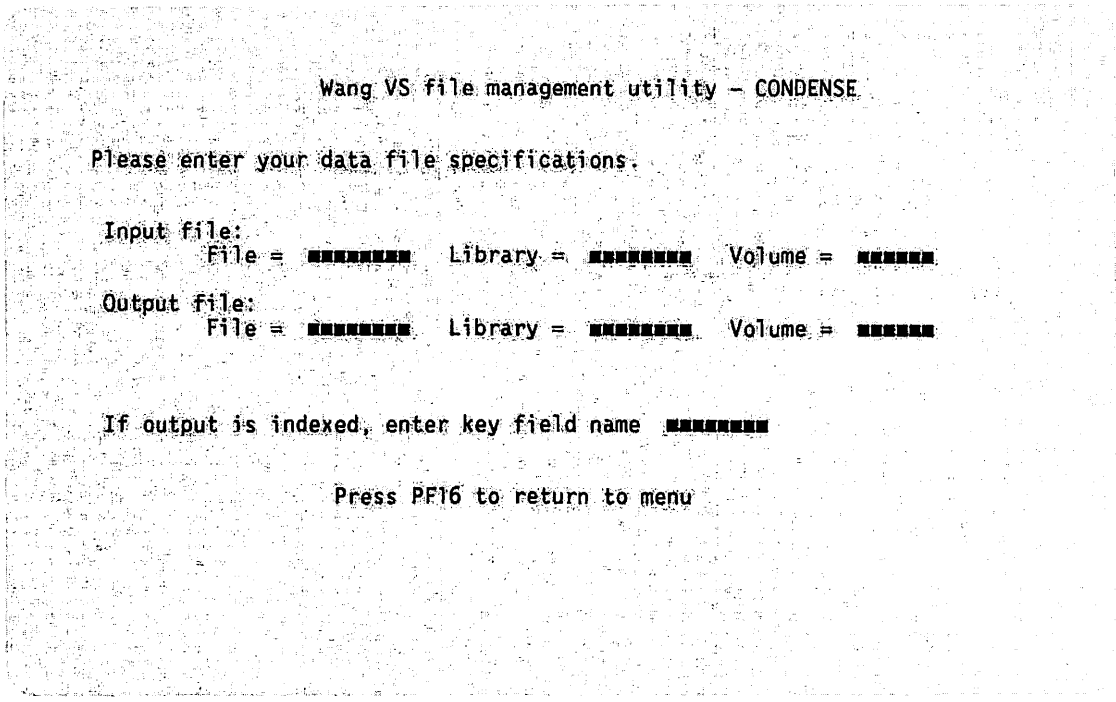


Figure 9-8. CONDENSE Data Files Specification Screen

The output data file that you specify from this screen is not created at this time. However, CONDENSE stores the names and locations of both input and output data files so that you do not have to respecify them when you choose to have a condensed output file created from the CONDENSE Function menu (Figure 9-2). You can also specify an indexed output file from the Data Files Specification screen.

Since CONDENSE neither accesses the input file nor creates the output file during this part of the process, you can leave these fields blank and simply press ENTER to continue. If you do this, and later command CONDENSE to create an output file from this parameter file, you will be prompted for input and output file information. However, this procedure does not allow the creation of an indexed output file. *If you want the output file to be indexed instead of consecutive, you must specify it from this screen.*

Supply the requested information as follows:

- Input file

  File -- The name of the file that contains the input records. Obviously, this must be an existing file.

  Library -- The library that contains the input file. If the INLIB field of your usage constants is specified, the value appears here as the default.

  Volume -- The volume where the input file resides. If the INVOL field of your usage constants is specified, the value appears here as the default.

- Output file

  File -- A valid file name for the output file. This file is not created until you press PF4 (Create condensed data file) from the CONDENSE Function menu.

  Library -- The library where you want the output file to reside. If the OUTLIB field of your usage constants is specified, the value appears here as the default.

  Volume -- The volume where the input file resides. The value (if any) from the OUTVOL field of your usage constants appears here as the default.

  Key field name -- You can, at your option, have CONDENSE create an indexed field output file, simply by using this field to supply the name of the key field. (Obviously it must be one of the fields you have specified for inclusion in the output record.) If you leave this field blank, CONDENSE creates a consecutive output file.

Enter or change the information in the fields as appropriate, and press ENTER to continue.

If you press PF16 instead, CONDENSE displays the Parameter File Maintenance menu (Figure 9-10). See Section 9.4 for information about the available options.

When you press ENTER from the Data Files Specification screen, the Output Control File Specification screen (Figure 9-9) appears.

```
                    Wang VS file management utility - CONDENSE


Please enter the output control file information.

        File = ■■■■■■■■   Library = USRCTL■■   Volume = VOL111

Specify the size of the pad field, if any 000
     The pad field is appended to each output record and contains blanks.

Please identify your user exit program file (if any).
        File = ■■■■■■■■   Library = ■■■■■■■■   Volume = ■■■■■■


                    Press PF16 to return to menu
```

Figure 9-9. CONDENSE Output Control File Specification Screen

From this screen you can specify the control file CONDENSE creates when it creates an output data file on the basis of this parameter file. Like the output data file, the control file is not created at this point, but rather when you press PF4 (Create a condensed output file) from the CONDENSE Function menu (Figure 9-2).

There are two additional options you can specify from the Output Control File Specification screen:

Pad field -- If you anticipate adding more fields to the output data record at some time in the future, a pad field can make this task easier by saving room in the output data file. A pad field, which contains nothing but blanks, is added to the end of each record to save space for data that may be added in the future. You can specify any number of blanks up to 999. Later, you can add the new fields to the control file in place of the pad field. The data file does not have to be rebuilt. When you modify an existing record and enter values in the new fields, this data simply overwrites the blanks placed in the pad field when the output record was created.

User exit subroutine -- By linking to a user-written subroutine during CONDENSE processing, you can increase the degree to which you control the contents of the output file. You can alter the contents of a record before it is written to the output file, or select records by means of criteria more flexible than those you can create from the Record Type Specification screen. The subroutine can examine and/or manipulate a particular type of record before it is placed in the buffer, and it can examine and/or manipulate the entire buffer contents before they are written to the output file. For more information about user exit subroutines, see Section 9.6.

Enter information in the fields of the Control File Specification screen as follows:

- Control file

  File -- Enter a name for the control file CONDENSE will create to describe the condensed data records in the output data file. The control file and the data file are created when you select that option by pressing PF4 from the CONDENSE Function menu (Figure 9-2).

  Library -- The name of the library in which you want the control file to reside. CONDENSE supplies a default library name by concatenating your user ID with the string CTL (e.g., USRCTL).

  Volume -- The volume where you want the control file to reside. If the INVOL field of your usage constants is set, the value from this field is the default.

- Pad field

  Size -- Enter the length in bytes of the pad field you want added to the output record. It should approximate the summed length of the fields you anticipate adding to the record. The maximum allowed is 999. The minimum, which is the default, is 0 -- in other words, no pad field is added to the output record.

- User Exit Subroutine

  File/Library/Volume -- If you want CONDENSE to link to a user exit subroutine during the process of creating the data file, enter the name and location of the executable program file that contains the subroutine in these three fields. A user exit subroutine is optional, but if you specify one, the executable file must be present in the specified location when you press PF4 from the CONDENSE Function menu to create the data file. Otherwise, an error will result.

Enter or change the information in the fields as appropriate, and press either ENTER or PF16. CONDENSE displays the Parameter File Maintenance menu (Figure 9-10). You can use this menu to modify any of the specifications you have entered up to now. Section 9.4 describes the available options.

If you do not want to change any of your specifications, press PF16 to create the parameter file and return to the CONDENSE Function menu (Figure 9-2). This completes the process of specifying and creating the parameter file.

9.4 MODIFYING A PARAMETER FILE

When you press PF2 from the CONDENSE Function menu (Figure 9-2), PF16 from the Data Files Specification screen (Figure 9-8), or ENTER or PF16 from the Output Control File Specification screen (Figure 9-9), the Parameter File Maintenance menu (Figure 9-10) appears.

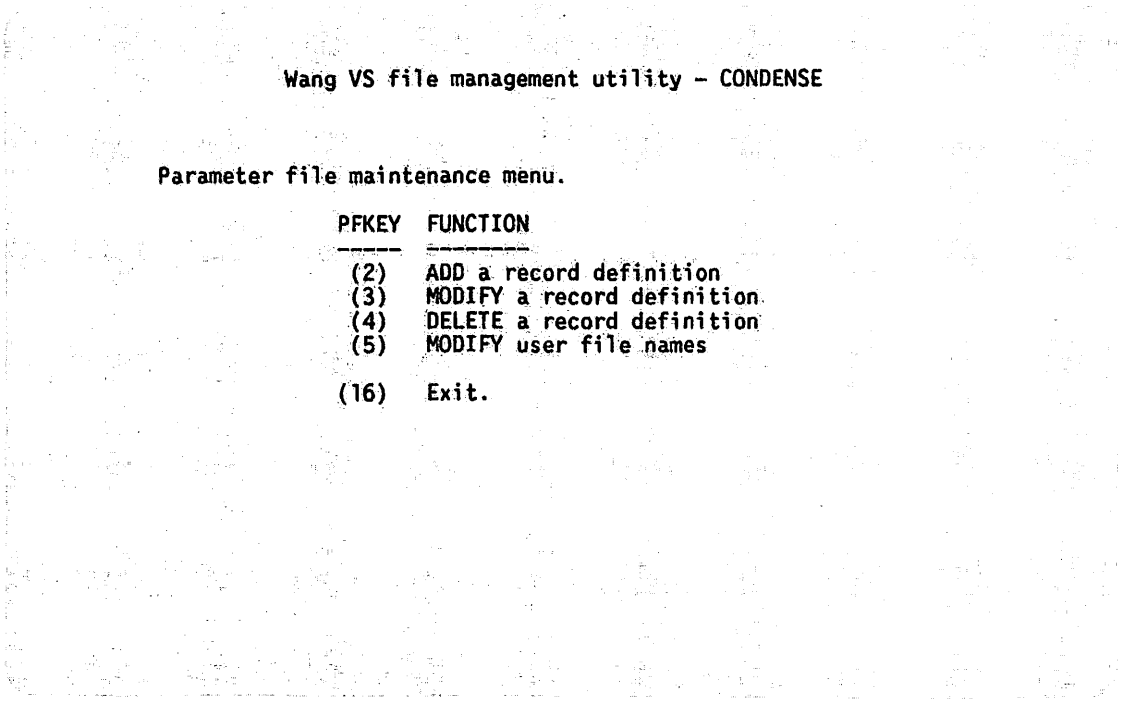


Figure 9-10. CONDENSE Parameter File Maintenance Menu

To select a function from the menu, press the indicated PF key. The following functions are available:

PF Key	Option and Description
2	**Add a record definition** -- To add another record type to the list of record types being extracted for the output file, press PF2. A Record Type Specification screen (Figure 9-3) appears. Enter your specifications for the new record type in the manner described in Section 9.3.1, and press ENTER. A Field Selection screen appears next. Section 9.3.2 describes the process of selecting fields. When you finish selecting fields and press ENTER, a blank Record Type Specification screen appears; you can continue adding record types and selecting fields indefinitely. When you have no more record types to define, press PF16 from a Record Type Specification screen.
3	**Modify a record definition** -- When you press PF3 to modify an existing record type specification, a screen appears that prompts you for the name, library, and volume of the control file that defines that record type. When you supply these and press ENTER, the Record Type Specification screen for that record appears, showing the previously specified value in each field. Change these as appropriate, and press ENTER. The Field Selection screen for the same record type appears next. At your option, change any of the previously specified values shown on this screen, and press ENTER. You are prompted once again for the name of a control file that describes another record type to modify. When you no longer want to modify record type specifications, press PF16 from this screen to return to the Parameter File Maintenance menu. (You can also return to the menu by pressing PF16 from a Record Type Specification screen or a Field Selection screen, but any modifications you have entered from that screen will be lost unless you press ENTER before you press PF16.)
4	**Delete a record definition** -- When you press PF4 to delete an existing record type specification, a screen appears that prompts you for the name, library, and volume of the control file that defines that record type. When you supply these and press ENTER, the record type specification is deleted, and the Parameter File Maintenance menu reappears. (Deleting a record type specification causes CONDENSE to ignore records of that type when creating the output file.) If you change your mind about deleting a record type specification after selecting this option, you can press PF16 from the prompt screen to return to the menu without deleting a specification.

PF Key	Option and Description
5	**Modify user file names** -- This option enables you to change the data and control file specifications for this parameter file. When you press PF5, the Data Files Specification screen (Figure 9-8) appears, with the current specifications in all fields displayed. Change these as you wish, and press ENTER. The Output Control File Specification screen (Figure 9-9) appears next, showing the currently specified values in all its fields. Make changes as appropriate, and press ENTER. You can change the specifications not only for the data and control files, but for all the options available from those two screens (see Section 9.3.4 for information). When you press ENTER from the Output Control File Specification screen, the Parameter File Maintenance menu reappears. You can also return to the menu by pressing PF16 from either screen, but any new specifications you have entered are lost unless you press ENTER before you press PF16.
16	**Exit** -- Press PF16 to return to the CONDENSE Function menu (Figure 9-2). This completes the process of modifying the parameter file.

9.5 CREATING A CONDENSED OUTPUT DATA FILE

When a parameter file has been created or modified so that it specifies the contents of the output file as you want them, you can begin creating that file by pressing PF4 from the CONDENSE Function menu (Figure 9-2). If all the input data files and control files have been correctly specified, and there are no files with duplicate names in the locations in which the output data and control files are created, this process takes place without interaction. The output data and control files are created in the locations you have designated, and the Function menu simply reappears. You can choose another function, or press PF16 to return to the Parameter File Specification menu, where you can either specify a new parameter file and begin CONDENSE processing again, or press PF16 to exit from the program.

If you have left any of these files unspecified, or have specified them incorrectly, CONDENSE displays a series of screens that prompt you for the correct information one file at a time. You can supply the necessary file specifications from these screens and still have the condensed data file created. (Certain options, however, cannot be specified at this time, such as a user file subroutine or an indexed output file.)

If, while one of the prompt screens that request file information is displayed, you decide to return to the function menu without creating an output file, you can do so by pressing PF1.

9.6 RUNNING THE REPORT UTILITY ON THE CONDENSED FILE

If the output file described by the currently specified parameter file exists, you can link to the REPORT utility from within CONDENSE in order to extract a report on the file. To select this function, press PF5. The REPORT Processing menu (Figure 6-1) appears, and REPORT processing continues as described in Chapter 6. The condensed output file created by means of the current CONDENSE parameter file is specified as the primary data file for the report. (However, you can override this specification, if, for example, you want the condensed file to be the secondary data file.)

Unless you modify the field sequence in REPORT, the fields in the condensed file are reported in alphabetical order, regardless of what record types the fields originally belonged to. However, the REPORT utility offers many ways to rearrange the contents of a report, including the order in which fields are reported. See Chapter 6 for a full description of the options available in REPORT processing.

9.7 INCORPORATING A USER EXIT SUBROUTINE IN CONDENSE PROCESSING

Incorporating a user-written subroutine into CONDENSE processing enables you to increase your control over the contents of the output file. You can alter the contents of a record before it is written to the output file, or select records by means of criteria more flexible than those you can create from the Record Type Specification screen. The subroutine can examine a particular type of record before it is placed in the buffer, omitting or modifying it at need. Later, when the buffer contents are about to be written to the output file, the subroutine can modify them according to your specifications before they become a new data record.

- A user exit subroutine could, for example, enable you to use a record type in a counter field calculation but omit it from the output data file. (Standard CONDENSE processing requires that at least one field from such a record type be selected for inclusion.)

- User exit subroutines are particularly well qualified for making global modifications in the output data file while it is being created. This can eliminate the tedious chore of repetitively modifying individual records with DATENTRY.

Incorporating a user exit subroutine in CONDENSE processing involves three steps:

1. The file specifications for the subroutine are included in the parameter file. You enter them from the Output Control File Specification screen (Figure 9-9) as described in Section 9.3.4. (You can also enter them in an existing parameter file after pressing PF5 from the Parameter File Maintenance menu -- see Section 9.4.)

2. The subroutine is written and assembled or compiled. See Section 9.7.1 for information.

3. The condensed file is created. If the subroutine has been created and compiled and is correctly specified in the parameter file, this part of the process requires no additional action on the user's part. Linking is automatic and must not be done manually. The subroutine is called every time CONDENSE encounters a record type specified in the parameter file (but before the record is placed in the buffer), *and* every time the buffer contents are to be written to the output file (but before they are actually read).

The order of the first two steps does not matter as long as both are completed before the third takes place.

9.7.1 Writing a User Exit Subroutine

You can write a user exit subroutine in any of the programming languages supported by the VS. Besides conforming to the specific language's requirements for an external subroutine, it must also receive and return two arguments -- record area and record code -- through which CONDENSE and the subroutine communicate. The subroutine's internal argument list must specify them in the following order:

Argument	Length	Comments
Record Area	Length of record or buffer	CONDENSE sets the length of this argument according to whether it is reading a record from the input file or is about to write the buffer contents to the output file. (The record code argument indicates which action is taking place.) The programmer has the responsibility to see that the variable receiving this argument is large enough to contain it. The value of this argument as passed by CONDENSE to the subroutine is the data in the record or buffer being processed; the subroutine can change the contents of the output file by modifying this value before the argument is returned to CONDENSE.

Argument	Length	Comments
Record Code	1 byte	This argument consists of a single character that indicates the type of record value being passed. When CONDENSE calls the subroutine before placing a record in the buffer, it sets the code to I to indicate an input record. When it calls the subroutine before writing the buffer contents to the output file, it sets the code to O to indicate an output record. The subroutine can cause an input record to be omitted from the buffer or an output record to be omitted from the condensed file by changing the value of this argument to B (Bypass).

The subroutine code must make sure that record length and field definitions are consistent with those in the record type or buffer. If an input record is being processed, the subroutine must determine which record type it is and set these parameters accordingly. Different settings will be needed for output records.

In COBOL programs, the linkage section must contain an 01-level redefinition of the record area argument for each record type in the parameter file and also for the output buffer. Field definitions are permissible, but are not required unless the fields are referenced in the Procedure Division.

Note: *You can obtain COBOL record and field descriptions for each record type by running the COBOL utility on their control files and selecting the source code function. If you use CONDENSE to generate an output control file before writing the user exit subroutine, you can also obtain record and field descriptions for the output buffer in the same way. Section 2.6 describes this CONTROL function.*

9.7.2 Two Sample User Exit Subroutines

Example 1

This BASIC program examines output records only and changes a company name from *HURON* to *SUPERIOR*.

```
10  SUB "EXIT" ADDR (RECORD$, CODE$)
20  DIM RECORD$ 80, CODE$ 1
30  IF CODE$ = "I" THEN 50
40  IF STR(RECORD$,4,5) = "HURON" THEN STR(RECORD$,4,8 =
    "SUPERIOR"
50  END
```

Example 2

This COBOL subroutine examines output records only and specifies that all records containing the last name *JONES* or the full name HENRIETTA HARDY should be bypassed; i.e., these records are not written to the output file. (Since the linkage section contains a redefinition for each record type, the subroutine could easily be modified to process input records as well as output records.)

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID.  USER1.

       ENVIRONMENT DIVISION.
       CONFIGURATION SECTION.
       SOURCE-COMPUTER.  WANG-VS.
       OBJECT-COMPUTER.  WANG-VS.

       DATA DIVISION.
       LINKAGE SECTION.

       01  RECORD AREA                  PIC X(80).

       01  INQUIRY1 REDEFINES RECORD-AREA.
           03  ID1                      PIC X(3).
           03  FNAME                    PIC X(20).
           03  LNAME                    PIC X(20).
           03  ADDRESS                  PIC X(36).
           03  CODE 1                   PIC X.

       01  INQUIRY2 REDEFINES RECORD-AREA.
           03  ID                       PIC X(3).
           03  NAME                     PIC X(35).
           03  AGE                      PIC X(3).
           03  JOB                      PIC X(38).
           03  CODEX                    PIC X.

       01  INQUIRY REDEFINES RECORD-AREA.
           03  ID-O                     PIC X(3).
           03  LNAME-O                  PIC X(20).
           03  CODE1-O                  PIC X.
           03  ID-O                     PIC X(3).
           03  NAME-O                   PIC X(35).
           03  CODEX-O                  PIC X.
           03  FILLER                   PIC X(17).
```

```
01  RECORD-CODE                  PIC X.
    88  INPUT-REC                VALUE "I".
    88  OUTPUT-REC               VALUE "O".
    88  RECORD-TO-BE-BYPASSED    VALUE "B".

PROCEDURE DIVISION USING RECORD-AREA
                         RECORD-CODE.
START-PROGRAM.
    IF INPUT-REC
    THEN EXIT PROGRAM.
    IF OUTPUT-REC
        PERFORM OUTPUT-RTN THRU OUTPUT-EXIT.
    EXIT PROGRAM.
OUTPUT-RTN.
    IF LNAME-O IS EQUAL TO "JONES" OR NAME-O IS EQUAL TO
        "HENRIETTA HARDY" MOVE "B" TO RECORD-CODE.
OUTPUT-EXIT.
    EXIT.
```

CHAPTER 10
CREATE

10.1 INTRODUCTION

The CREATE utility enables you to create a data file to your own design, using the following as components:

- Existing files (in whole or in part)
- Literal strings
- Pad characters
- ASCII numeric values (for record sequencing)

The file is made up of blocks whose structure you specify. A block may contain any number of records, whose structure you also specify. The fields each record contains may be fields from an input file, literal strings, or pad characters (repeated in each record), or ASCII sequencing fields. Different blocks may contain records of different sizes.

Note: *The term* block, *as used in this program and its documentation, has nothing to do with the 2048-byte physical blocks into which the Data Management System (DMS) organizes records. The size of CREATE's blocks is variable.*

Some uses for CREATE are

- Extracting a range of records from a consecutive or indexed file. For a consecutive file, "range" is general enough to allow any positive or negative increment. You can thus extract, for example, every other record, every third record, and so on.

- Merging data from a series of files. The records in the output file, for instance, could be defined so as to include names from one input file, addresses from another, and telephone numbers from a third.

- Deleting specified fields from a file. This may be desirable when converting, for example, 96-column IBM RPG files to VS 80-column files.

- Concatenating several files. This generalization of the EDITOR's XCOPY function can be used with files of any type. One of the commonest uses of CREATE, it is described in Section 10.7.

- Deblocking records. When an input file is made up of short records physically blocked into long records, the COUNT field in the Input File Field Specification screen (see Section 10.4.1) can be used to break these long records up so that the output file will contain records of the original size.

10.1.1 Overview

The first thing you do in running CREATE is specify the general characteristics of the output file -- the new data file the program will construct. When you finish, CREATE prompts you to begin defining the structure of the first data block to go into the file. You do this by specifying, one at a time, the nature of the fields that make up the records this block will contain. After selecting a field type from a menu, you are prompted to provide specifications appropriate to that type.

As soon as the specifications for the first field are complete, the field selection menu reappears. You select a type for the second field and are once again prompted for specifications.

This process is repeated until every field you want the records to contain has been specified. When you decline to specify any more fields, the record structure of the first block is fully defined.

At this point the field selection menu reappears, prompting you to define the first field for the records in the second block. You repeat the same process you carried out for the first block.

There is no need to repeat any particular sequence of fields. The records of each block are defined by the nature of the fields and the combination in which they occur. These need not match or even resemble their counterparts in any other block (unless you are using CREATE for a specific purpose, such as concatenation, that requires them to match). One of CREATE's options lets you specify a different record size for each block.

The procedure of defining fields and blocks is repeated as many times as necessary until you have defined every field for every block you intend the output file to contain.

Finally, when you are prompted once again to define the first field for a new block, you respond instead that you have no further definitions to enter. The structure of the output file is now fully defined, and CREATE, without any further command, proceeds to construct the output file.

10.1.2 Software Requirements

The DISPLAY utility should reside in the system library. CREATE has no other specific software requirements.

10.1.3 Running CREATE

Run the CREATE program from the Command Processor as you run any other program or utility. Enter CREATE into the program name field along with the appropriate library and volume names. (Since this utility does not reside in the system library, you must type the library and volume designations.)

The following sections describe CREATE processing:

Section	Process
10.2	Specifying the Output File
10.2.1	Additional Specifications for Indexed Files
10.3	Block Definition -- Preliminary Specifications
10.3.1	Specifying the Record Length for the Block
10.3.2	Specifying Alternate Keys for the Block
10.4	Specifying Fields
10.4.1	Specifying an Input File field
10.4.2	Specifying a Literal Field
10.4.3	Specifying a Pad Character Field
10.4.4	Specifying an ASCII Number Field

Section	Process
10.5	Completing a Block Definition
10.6	Additional Notes on Using CREATE
10.7	Using CREATE to Concatenate Files

10.2 SPECIFYING THE OUTPUT FILE

As soon as CREATE begins running, the Output File Specification screen appears (Figure 10-1).

```
Wang VS GETPARM v 7                          Parameter Reference Name:  OUTPUT
                                                          Message Id:  C001
                                                           Component:  CREATE

                        Information Required by CREATE

          VS File Creation Utility - Version  x.xx.xx (c) Copr. Wang 1988

Please specify output file parameters and press ENTER to continue:

Location:       FILE     = ■■■■■■■■  LIBRARY  = ■■■■■■■■  VOLUME   = ■■■■■■
Type:           TYPE     = ■■        (file org: A, A+, C, I, I+, O, P, R, W)
Record size:    RECSIZE  = ■■■■■     (output file [maximum] record size)
                MULTIPLE = NO■       (YES/NO for multiple record sizes)
Compression:    COMPRESS = YES       (YES/NO for compressed output)
Record count:   RECORDS  = ■■■■■■■   (approximate number of output records)
File class:     CLASS    = ■         (file class blank, A-Z, @, $, or #)
Access list:    ACL      = ■■■       (ACL/YES/NO for access list)
Retain:         RETAIN   = ■■■ days  (file retention period)
Release:        RELEASE  = YES       (release unused space at CLOSE-time)

Or Select:
                                     (13) Information  (16) To exit CREATE
```

Figure 10-1. CREATE Output File Specification Screen

This screen determines the character of the new data file the program will create. Enter your specifications as described below:

FILE/LIBRARY/VOLUME -- A name and location for the output file.

TYPE -- The organization of the new file. You have the following options:

A Alternate-indexed.
A+ XDMS Plus alternate-indexed.
C Consecutive.
I Indexed.
I+ XDMS Plus indexed.
O Object file, record size 1024 bytes. (COMPRESS must be set to NO.)
P Print file (COMPRESS must be set to YES). All records are created assuming two leading control bytes (see the *VS Principles of Operation*), and have trailing blanks removed.
W WP document file, record size 256 bytes. (COMPRESS must be set to NO.)

Note: *A+ and I+ values are only permitted for systems using XDMS.*

RECSIZE -- The length, in bytes, of the records in the new file. If the multiple-size option (next) is chosen, the value you enter is the maximum record size; otherwise, it is the size of all records in the file. The maximum size, in bytes, that you can specify depends on the file type:

A	2,043	
A+	30,000	
C	2,048	
I	2,043	
I+	30,000	
O	1,024	(only size allowed for this type)
P	2,024	
R	2,040	
W	256	(only size allowed for this type)

Note: *The A+ and I+ options for the TYPE and RECSIZE fields do not appear unless XDMS has been installed on the system where you are running CREATE.*

MULTIPLE -- Your choice (YES or NO) determines whether the file can contain records of different sizes. (The records in any one block must be all of the same size, but different blocks may contain different-sized records.) If you answer YES, you are prompted to specify the record size each time you begin defining a new block.

COMPRESS -- Answer YES and the records in the output file will be compressed.

RECORDS -- Enter the approximate number of records the new file will contain.

CLASS -- The VS file protection class (blank, A - Z, $, @, or #) for the new file.

ACL -- The Access Control List for the output file. You have the following options:

YES The file has the user's default ACL.
NO The file has no ACL.
ACL You can modify a copy of the user's default ACL.

RETAIN -- The number of days in the file retention period.

RELEASE -- Your choice (YES or NO) determines whether unused disk space will be released when the file is closed.

Press ENTER when you finish entering information. Alternatively, you can press PF13 to display two screens containing information about the fields on the Output File Specification screen, or PF16 to terminate CREATE.

10.2.1 Additional Specifications for Indexed Files

If you specified an indexed or alternate indexed output file (type A, A+, I, or I+), the Index Options screen appears as soon as you press ENTER from the Output File Specification screen. If your output file is of any other type, pressing ENTER begins the process of block definition (as described in Section 10.3.1 if multiple record lengths are allowed, or Section 10.4 if they are not.)

```
Wang VS GETPARM v 7                    Parameter Reference Name:  INDEXOPT
                                                    Message Id:  C002
                                                     Component:  CREATE

                    Information Required by CREATE

        VS File Creation Utility - Version  x.xx.xx (c) Copr. Wang 1988

Please specify the indexed file parameters and press ENTER to continue:

Primary key location:      KEYPOS   = ■■■■■
Primary key length:        KEYLEN   = ■■■      (1-255)
File creation mode:        MODE     = OUTPUT   (OUTPUT / IO)
Print key errors           ERRLIST  = NO■      (YES / NO)

XDMS entry ordered file?   ENTRYORD = NO■      (YES / NO)
Use hash table for index?  HASHTABL = ■■■      (YES / NO)
   If YES, size is         HASHSIZE = ■■■■■■■  (In blocks)

Specify secondary extent?  SECEXT   = NO■      (YES / NO)
   If YES, size is         SECEXTSZ = ■■■■■■■  (In blocks)

Or Select:
(1) Respecify OUTPUT                    (13) Information
```

Figure 10-2. CREATE Index Options Screen

Use this screen to supply specifications for your indexed file. The first four fields must be specified for *every* indexed or alternate indexed file:

KEYPOS/KEYLEN -- Define the primary key by position and length.

MODE -- The options are OUTPUT and IO. Output mode is faster but requires that the records be written in ascending key order in the file. If they are not so arranged, you must use I/O mode (enter "IO", not "I/O").

ERRLIST -- If you enter YES, all records that could not be written to the file (either because of duplicate or out-of-sequence key errors or because disk space was lacking) are listed in a print file.

If the output file is an *XDMS file* (i.e., type A+ or I+), specify these fields also:

Note: *The following fields and references to them are displayed only if XDMS has been installed on the system where CREATE is being run.*

ENTRYORD -- Enter YES to create an XDMS indexed or alternate indexed file with an entry-order data set. If you enter NO, the file has the default key-order data set.

HASHTABL -- Enter YES to have the primary index stored in a hash table, or NO to decline this option.

HASHSIZE -- If you entered YES in the HASHTABL field, enter the number of blocks to allocate for the initial hash table size. (If HASHTABL is NO, this field can be left blank.)

SECEXT -- Enter YES to specify the secondary extent size for an XDMS file.

SECEXTSZ -- If you entered YES in the SECEXT field, enter the number of blocks that XDMS will add to the file each time it becomes full. (IF SECEXT = NO, this field can be left blank.)

When the specifications are complete, press ENTER to continue. If you choose, you can instead press PF13 for information about the fields on the Index Options screen or PF16 to terminate the program.

If the output file you have specified is alternate indexed (TYPE = A or A+), the Alternate Key Specification screen (Figure 10-3) appears immediately after the Index Options screen.

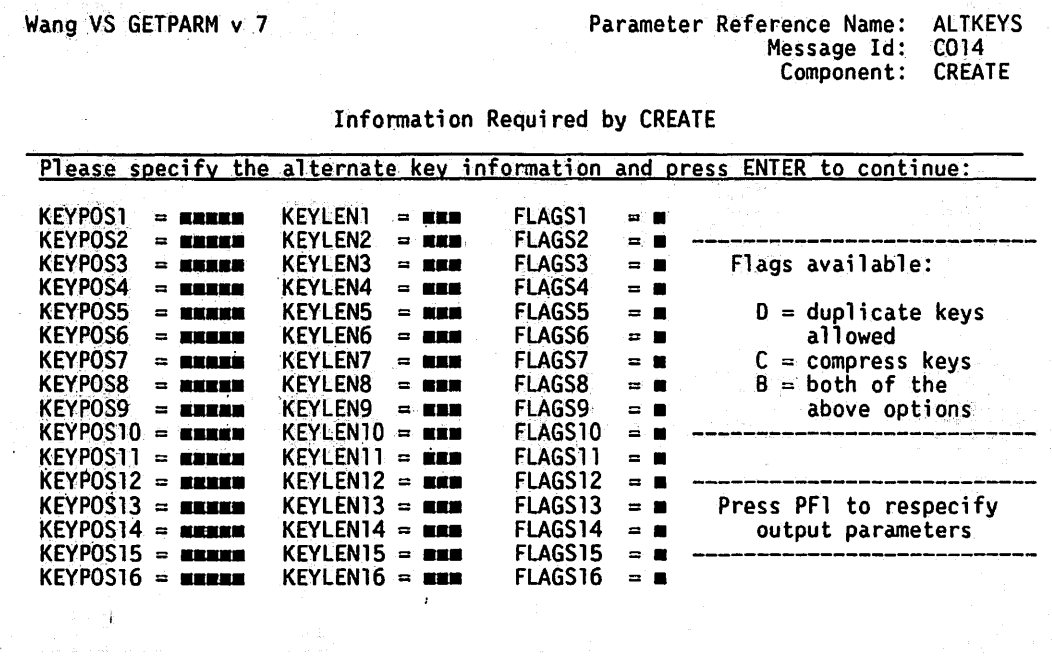


Figure 10-3. CREATE Alternate Key Specification Screen

The screen provides blanks for up to 16 alternate keys to be specified by position, length, and optional flag. Fill in as many of the blanks as you need and press ENTER. Whenever you begin defining a block, you are prompted to specify the alternate key paths for that block.

10.3 BLOCK DEFINITION -- PRELIMINARY SPECIFICATIONS

Note: *If you specified an output file with multiple record lengths, alternate keys, or both, you must begin each block definition with some preliminary specifications before you specify the fields. These preliminary specifications are described in Sections 10.3.1 and 10.3.2. If the output file has neither multiple record lengths nor alternate keys, block definition begins immediately with the specification of fields (see Section 10.4).*

10.3.1 Specifying the Record Length for the Block

When an output file has multiple record lengths, the record length must be defined separately for each block. If you have chosen this option, the definition of a block begins, as soon as output file specifications are completed, with the appearance of the Record Size Specification screen (Figure 10-4).

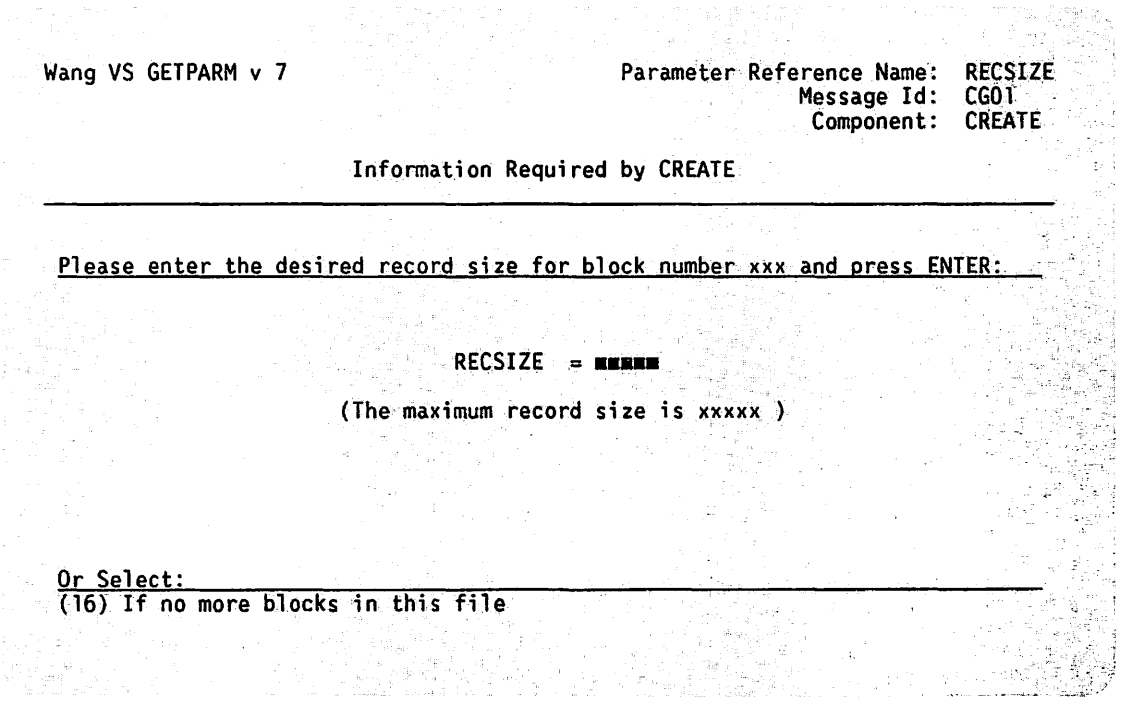


Figure 10-4. CREATE Record Size Specification Screen

The default maximum record size shown is the size you specified on the Output File Specification screen. If the record size for the block you are about to define is different, enter it. Press ENTER.

Whenever you indicate that you have entered all the field definitions for a block, this screen reappears so that you can begin defining the next block.

If the block you have just finished defining was the last, press PF16 to close definitions and begin the process of creating the output file.

Note: *If you press PF16 the first time you see the Record Size Specification screen, before you define any blocks, CREATE produces an output file consisting of a header without records, and displays the End-of-Job screen (Figure 10-12).*

10.3.2 Specifying Alternate Keys for the Block

If you have defined your output file as an alternate indexed file (TYPE = A or A+), you must, at the beginning of each block definition, specify the alternate keys that apply to that block. The Alternate Key Selection screen prompts you for this information. Figure 10-5 shows a sample of this screen.

If you specified multiple record lengths, this screen appears immediately after you specify the record length for this block on the Record Size Specification screen (Figure 10-4).

If you did not specify multiple record lengths, the screen first appears immediately after you specify the alternate keys on the Alternate Key Specification screen (Figure 10-3), and thereafter each time you indicate that you have finished defining fields for a block.

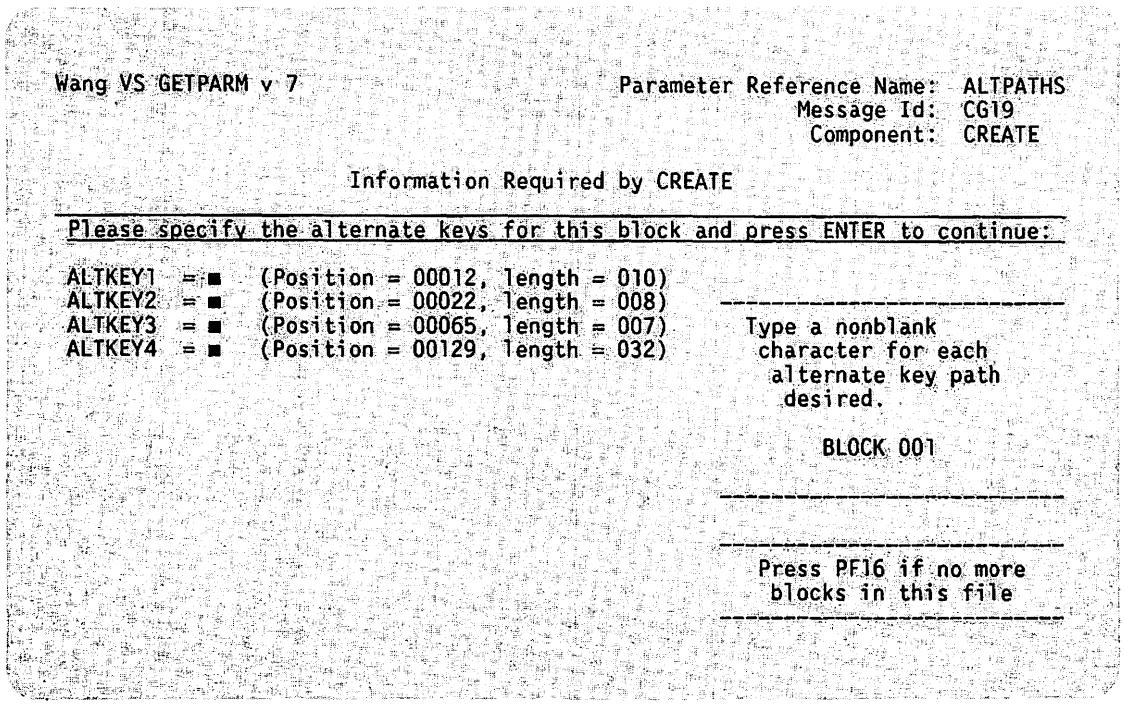


Figure 10-5. A Sample CREATE Alternate Key Selection Screen

The Alternate Key Selection screen shows all the alternate keys you have specified from the Alternate Key Specification screen. To select the ones that apply to the block you are about to define, enter any character other than a blank in the appropriate ALTKEY fields. After making your selections, press ENTER.

When this screen reappears after you have defined the last block, press PF16 to close definitions and begin the creation of the output file.

Note: *If you press PF16 the first time you see this screen, before defining any blocks, the program creates a file consisting of a header without records, and displays the End-of-Job screen (Figure 10-12).*

10.4 SPECIFYING FIELDS

If you have specified neither multiple record sizes nor an alternate indexed output file, the Field Selection menu (Figure 10-6) appears as soon as you press ENTER from the Output File Definition screen. Otherwise, you must first complete some preliminary specifications, as described in Section 10.3. When field specification begins, the Field Selection menu (Figure 10-6) shows the block and field number, and prompts you to select the type of field you are about to specify.

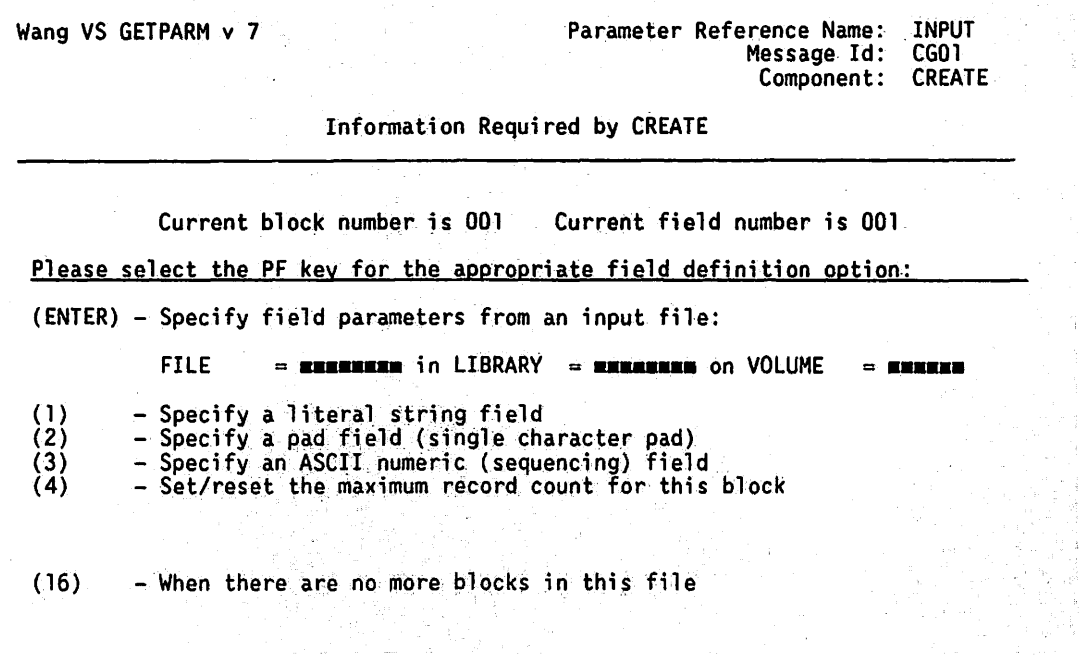


Figure 10-6. CREATE Field Selection Menu

Select an input file field by specifying the file name, library, and volume of the input file and then pressing ENTER. Use PF1, PF2, and PF3 to select other types of fields. (Each of these options leads to its own specification screen. All are described separately below.) Press PF4 to set or change the maximum record count for this block.

Whenever you finish specifying a field, the Field Selection menu reappears with the current field number changed to show that the field you are being prompted to specify is new.

When there are no more fields to specify for the current block, press PF16 so you can begin defining the next block. If you have not yet specified a field for the current block, pressing PF16 indicates that you have no more blocks to define and are ready for the creation of the output file. (The prompt at the bottom of the screen indicates which meaning PF16 has at any given time.)

10.4.1 Specifying an Input File Field

An input file field is filled with data extracted from an existing file. You must identify the file and specify the part of each record to be read from it. You also have the option of supplying a value for comparison so that CREATE will select only certain records to be included in the output file. To specify an input file field, use the Name, Library, and Volume fields provided at the top of the Field Selection menu. Enter the name and location of the input file, and press ENTER. The Input File Field Specification screen (Figure 10-7) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  INPUT
                                                          Message Id:  CG01
                                                           Component:  CREATE

                          Information Required by CREATE
_____________________________________________________________________________

   Input file xxxxxxxx in library xxxxxxxx on volume xxxxxx is a consecutive,
            variable-length file with xxxxxx records of size xxxx

  Please supply the field specifications and press ENTER to continue:
  Starting positions and field length:
   INPOS    = ■■■■     OUTPOS   = ■■■■■    LENGTH   = ■■■■     COUNT    = ■■■■
  Range of input records to be used:
   START    = FIRST■■
   END      = LAST■■■  INCRMENT = 1■■■■■■
  Compare field specification (optional):
   CINPOS   = ■■■■     CLENGTH  = ■■       COMPARE  = ■■   (EQ,NE,GT,GE,LT,LE)
   CSTRING  = ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

  Or Select:
  (1) Return to the field selection menu
  (2) Switch from ASCII to Hex mode          (13) Description of the field options
```

Figure 10-7. CREATE Input File Field Specification Screen (for a Consecutive File)

Enter the specifications as follows:

INPOS -- The starting position in the input record of the field to be extracted.

OUTPOS -- The starting position of the same field in the output record.

LENGTH -- The length of the field.

COUNT -- To deblock or divide large records (e.g., to rewrite one 800-byte record from the input file as ten 80-byte records in the output file), you would enter the divider (10, in the example) in this field.

START/END/INCREMENT -- The range of records to be read from the input file is END specified differently depending on the type of file. *If the input file is a consecutive file,* START and END must both be valid record numbers (alternatively, you may use FIRST and LAST). INCRMENT may be any nonzero value, positive or negative; the default is 1. *If the input file is indexed or alternate indexed,* START and END are primary key values and may be specified in ASCII or hexadecimal form, up to a maximum of 64 characters. (The INCRMENT field is irrelevant to these types and does not appear).

If you choose, you can use the last four fields on this screen for comparing a particular field in the input records to a specified value in order to select the records to be included in the output file. For instance, you could direct CREATE in this manner to select, from a file that contains all employee salary records, only the records of those employees whose salary is above or below a specific figure. Records that did not meet this specification would be left out of the output file. If you do not want to select records in this way, leave these fields blank.

CINPOS -- The starting position of the field in the input record that is to be compared to the specified value.

CLENGTH -- The length of the same field. It may be specified for ASCII but not for hexadecimal values. When CLENGTH is unspecified, the length of CSTRING (if nonzero) is the default.

COMPARE -- The type of comparison:

EQ	Equal
NE	Not equal
GT	Greater than
LT	Less than
GE	Greater than or equal
LE	Less than or equal

CSTRING -- The ASCII or hexadecimal value to be compared to the input field. (Use PF2 as needed to switch back and forth between ASCII and hexadecimal input modes.)

When you have entered all your specifications from the Input File Field Specification screen, press ENTER to continue. The Field Selection menu reappears, with the numbers changed to indicate that you are specifying the next field.

You also have the following options from the Input File Specification screen:

PF Key	Option and Description
1	**Return to the field selection menu** -- Pressing PF1 redisplays the field selection menu so that you can specify another input file or field type.
2	**Switch from ASCII to hex mode** -- To switch to a mode in which CREATE expects hexadecimal instead of ASCII input for the CSTRING field, press PF2. Use this key to switch back and forth at will between these two modes.
13	**Description of the field options** -- You can display a help screen describing the fields of the Input File Field Specification screen by pressing PF13.

10.4.2 Specifying a Literal Field

A literal field is filled with a string of up to 64 characters which you enter in the course of specifying the field. To specify a literal field, press PF1 from the Field Selection menu. The Literal Field Specification screen (Figure 10-8) appears.

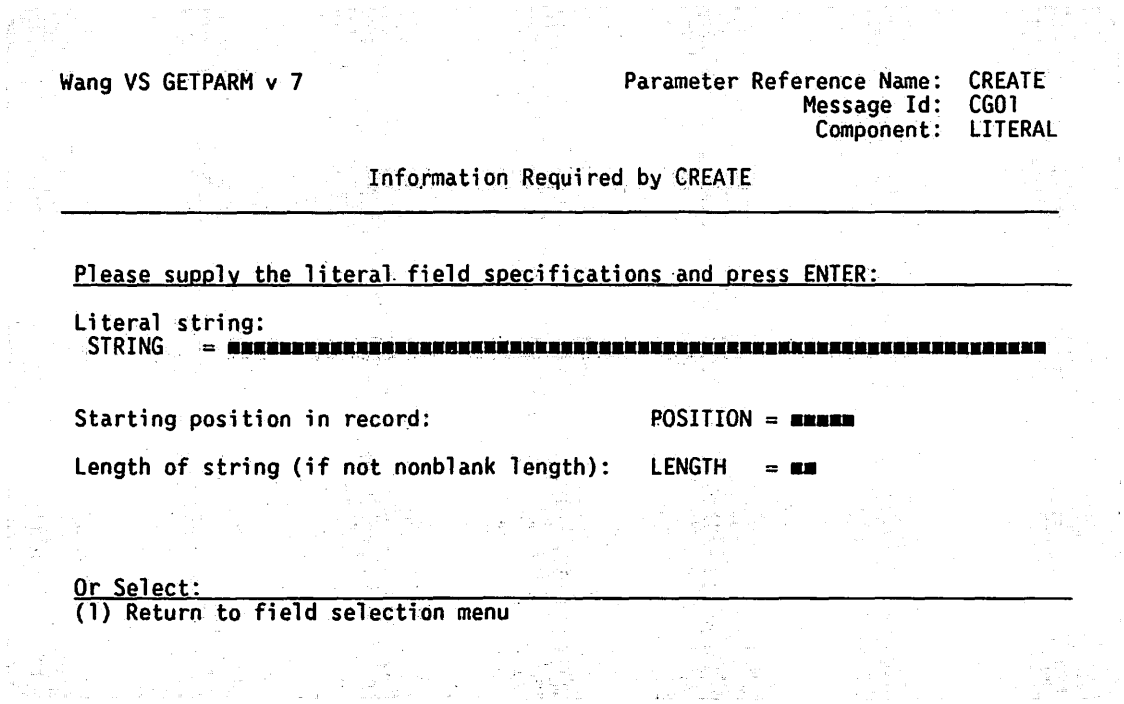


Figure 10-8. CREATE Literal Field Specification Screen

Enter the specifications as follows:

STRING -- The literal string of which this field is to consist. It may be up to 64 characters in length.

POSITION -- The starting position of this field in the output record.

LENGTH -- If the length of the field is different from the number of characters you have entered in STRING, enter the length here.

When the specifications are complete, press ENTER. The Field Selection menu reappears, with the numbers changed to indicate that you are specifying the next field.

If you want to return to the Field Selection menu instead of defining a literal field, Press PF1 instead of ENTER.

10.4.3 Specifying a Pad Character Field

A pad field is used to pad unused parts of the record with repetitions of a particular character, such as a blank or a hexadecimal 0. It is also used to "blank out" variable-length fields in order to prevent the repetition of characters from previous records (a process described in Section 10.6). To specify a pad field, press PF2 from the Field Selection menu. The Pad Character Field Specification screen (Figure 10-9) appears.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  PAD
                                                 Message Id:  CG01
                                                  Component:  CREATE

                    Information Required by CREATE

Please supply the pad character specifications and press ENTER:

Pad character:                         PAD      = ■
Starting position in record:           POSITION = ■■■■■
Length of pad field:                   LENGTH   = ■■■■■



Or Select:
(1) Return to the field selection menu
(2) Switch from ASCII to Hex mode
```

Figure 10-9. CREATE Pad Character Field Specification Screen

Enter the specifications as follows:

PAD -- The character you want repeated throughout this field. (Use PF2 as needed to switch back and forth between ASCII and hexadecimal input modes.)

POSITION -- The starting position of this field in the output record.

LENGTH -- The length of the field.

When specifications are complete, press ENTER to continue. The Field Selection menu reappears, with the numbers changed to indicate that you are defining the next field.

If you want to return to the Field Selection menu without defining a pad field, press PF1 instead of ENTER.

10.4.4 Specifying an ASCII Number Field

An ASCII number field is used to insert a sequencing number into each record. To specify this field type, press PF3 from the Field Selection menu. The ASCII Number Field Specification screen (Figure 10-10) appears.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  ASCII
                                                 Message Id:  CG01
                                                  Component:  CREATE

                    Information Required by CREATE

Please supply the ASCII number field specifications and press ENTER:

Starting value:                        START    = ■■■■■■■■■■
Ending value:                          END      = ■■■■■■■■■■
Increment:                             INCRMENT = ■■■■■■■■■■
Starting position in record:           POSITION = ■■■■■
Length of numeric field:               LENGTH   = ■■
Indefinite repeat option:              REPEAT   = NO■          (YES / NO)

Or Select:
(1) Return to field selection menu
```

Figure 10-10. CREATE ASCII Number Field Specification Screen

Enter the specifications as follows:

START -- The number at which the count should begin.

END -- The number at which the count should end.

INCRMENT -- The increment (it may be any nonzero value).

POSITION -- The starting position of this field in the output record.

LENGTH -- The length of the numeric field (it may be up to 10 characters).

REPEAT -- If you enter YES, the count starts again if the end number is reached before all the output records are created. It is repeated as many times as necessary. If your choice is NO (the default), no further records are created in the output file after the end number is reached.

Once your specifications are complete, press ENTER to continue. The Field Selection menu reappears, with the numbers changed to indicate that you are now defining the next field.

To return to the Field Selection menu without defining an ASCII number field, press PF1 instead of ENTER.

10.5 COMPLETING A BLOCK DEFINITION

Since there is no fixed number of fields allowed in a record, the Field Selection menu reappears every time a field is defined. The number on the screen changes to indicate that you are being prompted to define the next field. When there are no more fields to define, respond by pressing PF16. This completes the definition of a block.

When you finish defining a block, CREATE checks the lengths and positions of the fields you have defined, and compares these with the record length you have specified. If any bytes within the record are found to be undefined, the Undefined Region Warning screen (Figure 10-11) appears.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  GAP
                                              Message Id:  CG01
                                              Component:  CREATE

                    Information Required by CREATE

One or more bytes of the output record have not been defined for block xxx
If not defined, their values will be unpredictable. Press ENTER to return to
the field selection menu:

          The first five undefined regions are listed below:

                    Start     Length
                    xxxx      xxxx
                    xxxx      xxxx
                    xxxx      xxxx
                    xxxx      xxxx
                    xxxx      xxxx

Or Select:
(1)  Return to the field selection menu
(16) End field specifications for this block
```

Figure 10-11. CREATE Undefined Region Warning Screen

The Undefined Region Warning screen shows which bytes of the record will contain unpredictable "data" if the field definitions are not revised or completed. If, after reviewing this information, you do not think changes are needed, press PF16 to verify that field specifications are complete. If you decide to make changes, press ENTER or PF1 to return to the field selection menu.

Note: *CREATE provides no way to recall a specific field specification or block definition for modification. However, you can replace any previously defined field by defining a new field of the same length and setting the starting position in the output record (OUTPOS or POSITION) to the same value.*

If you have specified multiple record lengths, the Record Size Specification screen (Figure 10-4) reappears when a block definition is completed. You can either specify the record size for the next block, or press PF16 to indicate that there are no more blocks to define.

If you have specified uniform record lengths, but defined your output file as an alternate indexed file, the Alternate Key Selection screen (Figure 10-5) reappears when a block definition is completed. You may either select the alternate keys for the next block, or press PF16 to indicate that there are no more blocks to define.

If you have specified neither multiple record lengths not an alternate indexed output file, the Field Selection menu remains on your screen when a block definition is completed. Both numbers change, however, to indicate that you are now being prompted to define the first field of the next block. If you have no more blocks to define, press PF16.

Your indication that the last block has been defined signals CREATE to begin constructing the new data file. At the end of this process, the CREATE End of Job (EOJ) screen appears. Figure 10-12 shows an example of this screen.

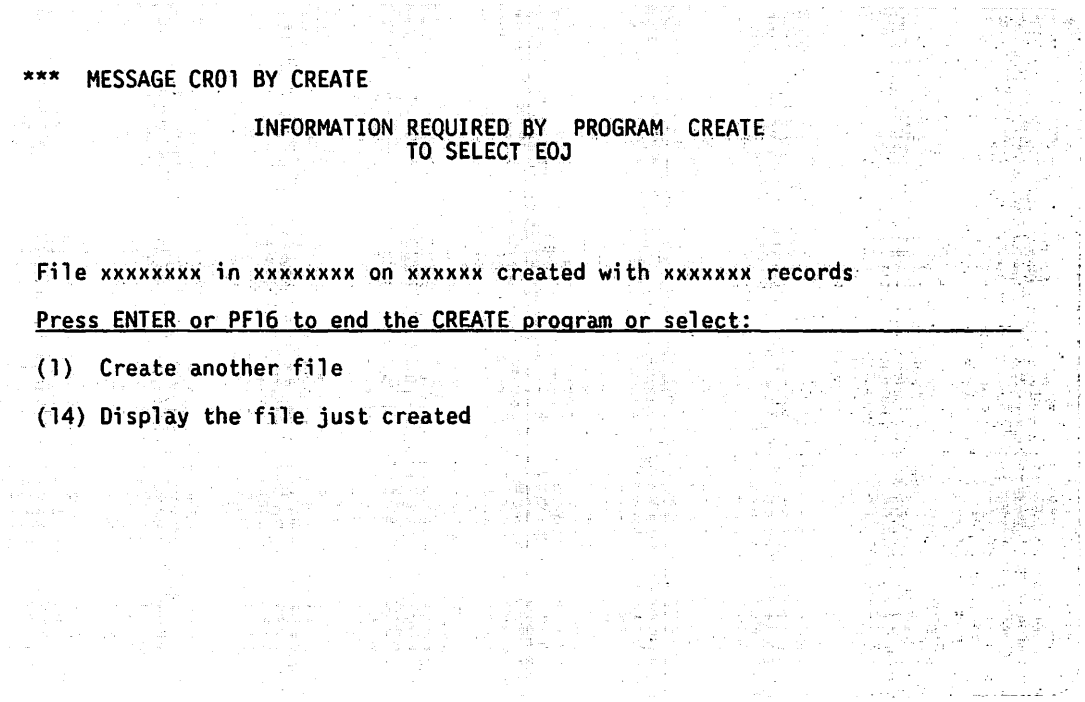


Figure 10-12. A Sample CREATE EOJ Screen

This screen announces the creation of the new file and reports the number of records it contains. It also reports the number of records that could not be created because of various errors. (If you have set the ERRLIST field in the Output File Definition screen to YES, these records are preserved in a print file.)

To display the new data file, press PF14. CREATE links to the VS DISPLAY utility to display the file, and afterwards returns to this EOJ screen. Pressing PF1 starts CREATE over again at the Output File Specification screen, so that you can create another file. PF16 terminates CREATE processing.

10.6 ADDITIONAL NOTES ON USING CREATE

Record Count -- The number of records in a block may be limited by the nature of the fields selected or by an explicit record count which you specify. (You enter the latter from the Field Selection menu by pressing PF4.) For the purpose of determining record count, the field types fall into three categories:

- An input file field implies a record count equal to the number of records selected, based on the starting record or key, the ending record or key, and (in consecutive files) the increment specified, if any.

- An ASCII number field implies a record count equal to the number of values specified by the start, end, and increment fields (unless you select the indefinite repetition option).

- Literal and pad character fields are indeterminate; i.e., they are repeated in any number of records, and imply no particular record count.

If the block definition contains at least one of the following -- a field of type A, a field of type B, or a user-supplied record count -- the number of records written will be equal to whichever of these implicit or explicit counts is *lowest*. There may, of course, be several fields of type A or type B. In all cases, however, the *minimum* count specified determines the number of records written.

If you have specified only fields of type C (or type B with indefinite repetition), CREATE prompts you to supply a record count when you declare the block definition complete.

Limitation -- The only limit on block and field specifications applies to the number of separate input files that can be specified in a *single block*. This limit (approximately 20) is set by the system. If you attempt to specify a new input file after reaching the limit, CREATE forbids you to do so.

Order of Specification -- Fields are placed in the record in the order in which they are specified. This is significant when several fields are specified which include the same character position(s) -- for an example, see the next note.

Variable-length fields and records -- When dealing with variable-length fields, CREATE does not automatically clear the input record area. The result is that a field with fewer characters than the defined record length may appear "padded" with characters left over in the same field from preceding records. For instance, if Record #1 contains HERBERT in the First Name field, and Record #2 is supposed to contain ANN, it will show ANNBERT instead.

Print files, which consist of variable-length records, present the same problem on a larger scale.

You can solve this problem by using pad fields. Before defining a variable-length field, first define a pad field identical to it in length and position. Use an ASCII blank (hexadecimal 20) as the pad character. Then define the variable-length field. When the fields are defined in this order, each time CREATE makes a new record, the pad character field overwrites the corresponding field from the previous record with blanks before new data is written into the same space. This insures that no "leftover" characters trail the new data.

For a print file, define as the first field in each record a blank pad character field as long as the maximum length of the record itself. This blanks out the entire record so that no characters from the previous record appear in it. Extra blanks at the end of short records have no effect on the printed output, since CREATE removes trailing blanks from each record of an output file that has been defined as a print file. (See the description of the TYPE field of the Output File Specification screen, Section 10.2.)

10.7 USING CREATE TO CONCATENATE FILES

CREATE is often used to concatenate files. This is a simple matter of defining one block for each input file, in the order in which you want the files joined together. Unless you intend to modify the files in the process of concatenating them, it is sufficient to define a single field, of the input file type, for each block. Identify the input file and set the specifications to include all the records it contains, from first to last. Then use PF16 to declare the block definition complete.

Continue to define blocks in the same way until you have defined one block for each input file. As soon as you indicate (once again using PF16) that all the blocks have been defined, CREATE concatenates all the files you have named into a single output file.

CHAPTER 11
DISPLAY

11.1 INTRODUCTION

The DISPLAY utility enables you to examine the contents of any disk file on your workstation screen. The utility provides information about the file organization and the number and size of the records in the file. For indexed files, DISPLAY also indicates the size and location of the index key and the number of alternate indexes. DISPLAY functions include the following:

- Displaying a file in ASCII (American Standard Code for Information Interchange) or hexadecimal mode
- Finding and displaying an indexed record by key value
- Listing the keys for an alternate indexed file and optionally changing the path so that you can access records in order of the alternate keys
- Finding and displaying a record in a consecutive file by record, line, or block number
- Finding and displaying a record in a relative file by record, line, or block number
- Finding and displaying a record by text string
- Producing a printed copy of all or part of a file in record or block access mode

11.1.1 Record Access Versus Block Access

A file can be accessed by record or block.

Record access -- Displays the file in records as opposed to blocks. It uses either of two formats:

Record-oriented format -- Displays consecutive, relative, or indexed files on a record by record basis.

Report-oriented format -- Displays consecutive or relative files on a line by line basis; that is, each record is displayed on one line. The DISPLAY utility cannot present indexed files in report-oriented format.

Block access -- Displays an exact image of the file layout as it exists. The file is displayed in 2-KB (2048-byte) blocks in physical block sequence. Data blocks and index blocks are displayed for an indexed file.

For detailed descriptions and examples of all DISPLAY modes and formats, see Section 11.4.

11.1.2 Overview

DISPLAY processing begins when you specify the file you want to examine. At the same time you also specify the access mode (record or block) and whether DISPLAY should open the file in Input or Shared mode. A menu then appears with a list of available functions. (The list varies according to the access mode and the organization of the file you have specified.) From this menu you can display the file, moving or manipulating the display by means of the PF keys listed in the menu. Other PF keys enable you to change formats (when access mode and file organization permit), switch between ASCII and hexadecimal display modes, or print a hard copy of the displayed file.

11.1.3 Running DISPLAY

To run the DISPLAY utility, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify DISPLAY in the Program field and press ENTER.

The following sections describe DISPLAY processing:

Section	Process
11.2	Specifying the Input File and Options
11.2.1	Opening Files in Shared Mode
11.3	Menus and Functions
11.3.1	Using DISPLAY Functions
11.3.2	Changing Display Formats
11.3.3	Printing a Displayed File
11.4	Display Formats
11.4.1	Record Access, Consecutive File, and Record-Oriented Format
11.4.2	Record Access, Consecutive File, and Report-Oriented Format
11.4.3	Record Access, Relative File, and Record-Oriented Format
11.4.4	Record Access, Relative File, and Report-Oriented Format
11.4.5	Record Access, Indexed File, and Record-Oriented Format
11.4.6	Block Access
11.4.7	ASCII and Hexadecimal Display Modes

11.2 SPECIFYING THE INPUT FILE AND OPTIONS

When DISPLAY processing begins, the Input File and Options screen (Figure 11-1) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  INPUT
                                                          Message Id:  0000
                                                           Component:  DISPLY

                        Information Required by DISPLAY

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

          *** Wang VS File Display Utility - Version  x.xx.xx ***
             *** Copyright, Wang Laboratories, Inc., 1986 ***

To display a file, enter the name and location of the file to be displayed.

        FILE     = ■■■■■■■■ in LIBRARY   = ■■■■■■■■ on VOLUME   = ■■■■■■

Select the access mode to be used when displaying the file.

        ACCESS   = RECORD      (RECORD/BLOCK/PRINT)

Specify whether the file should be opened input or shared mode.

        MODE     = INPUT■      (INPUT/SHARED)

                    (Press PF16 to exit the DISPLAY program.)
```

Figure 11-1. DISPLAY Input File and Options Screen

Use this screen to specify the input file, the type of access, and the mode in which the file should be opened. Enter information in the fields as follows:

FILE -- Specify the name of the file you want to display or print.

LIBRARY -- Specify the name of the library in which the input file resides. If the INLIB field of your usage constants is set, the value from that field appears here as a default.

VOLUME -- Specify the name of the volume on which the input file resides. If the INVOL field of your usage constants is set, the value from that field appears here as a default.

ACCESS -- Specify the access mode in which the input file data should be displayed. Acceptable values are RECORD (the default), BLOCK, and PRINT (a special case of RECORD). For more information on display formats, see Section 11.3.

RECORD -- The input file is displayed in logical records. If the input file is consecutive or relative, you can format the display record by record (record-oriented format) or line by line (report-oriented format). You must use record-oriented format for indexed files.

BLOCK -- The input file is displayed in 2-KB physical records. Specifying block access enables you to view an exact image of the file layout as it exists on disk. You cannot display or print shared files in block access mode.

PRINT -- If the input file is consecutive or relative, you can specify the print access mode, which causes the file to be displayed immediately in report-oriented format without the Display Options menu appearing first. The file cannot be shown in record-oriented format, but print access is in other respects identical to record access.

MODE -- The mode in which the input file is to be opened. Specify INPUT (the default) or SHARED for an indexed or consecutive file. *You must specify INPUT for a relative file.*

11.2.1 Opening Files in Shared Mode

If you specify Shared mode, the DISPLAY Lock screen (Figure 11-2) prompts you to indicate whether you want the file to be locked.

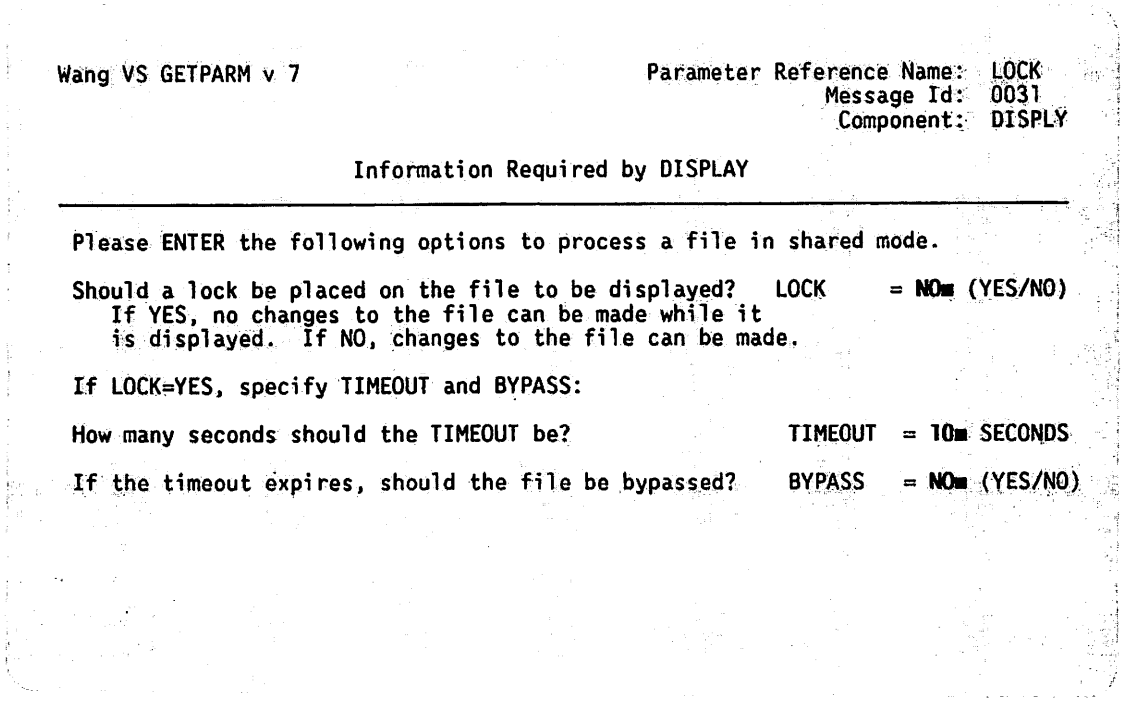


Figure 11-2. DISPLAY Lock Screen

Enter information in the fields as follows:

LOCK -- Enter YES or NO to indicate whether you want to suspend updates to a file while you are displaying it. YES locks the file, and it cannot be changed while it is being displayed. NO (the default) places no lock on the file and permits another program to change it during DISPLAY processing. If LOCK = NO, the values in the Timeout and Bypass fields are ignored.

TIMEOUT -- Specify a timeout value (0 - 255 seconds) for a file if LOCK = YES. If DISPLAY finds another user currently holding the file for update, the Timeout field specifies the length of time for DISPLAY to wait. If the file is released before this time expires, DISPLAY opens the file in Shared mode with a lock. The default value is 10 seconds.

BYPASS -- Enter YES or NO (the default) to indicate whether you want to skip a file if the timeout value expires before the file is released. If BYPASS = YES and the timeout expires, DISPLAY skips the file. If BYPASS = NO and the timeout expires, the Lock screen reappears with the message "FILE filename IN library ON volume IS HELD BY USER xxx."

When a timeout expires, you can continue the DISPLAY operation by respecifying the Lock, Timeout, and Bypass options, and pressing ENTER. DISPLAY also informs you of the following options:

- You can skip the file on which the timeout occurred by pressing PF1.

- You can terminate the DISPLAY operation if you are copying a library or volume by pressing PF16.

If a timeout expires with a Bypass value of NO during a background task, the DISPLAY operation is automatically cancelled.

Note: *The Record Access Method (RAM) is always used to display shared files. For more information about RAM, see the* VS DMS Reference.

11.3 MENUS AND FUNCTIONS

After you specify the input file, the access type, and the open mode, one of the six DISPLAY Functions menus appears. Six different Functions menus are available, corresponding to the six formats in which a file can be displayed:

- Record access, consecutive file, and record-oriented format
- Record access, consecutive file, and report-oriented format
- Record access, relative file, and record-oriented format
- Record access, relative file, and report-oriented format
- Record access, indexed file, and record-oriented format
- Block access

(For a description of each DISPLAY format, see Section 11.4.)

Each menu has a set of functions specific to the format, but all are similar in appearance. The input file name, its file organization, the maximum number of bytes in each record or block, and the number of records or blocks in the file are listed at the top of the menu. Key position, key size, and alternate index information is included for indexed files.

11.3.1 Using DISPLAY Functions

On any screen that displays file contents (whatever the access mode or format orientation), you can shift or manipulate the display by pressing PF keys. Most PF keys used in DISPLAY processing have the same function from screen to screen. However, some PF key functions change according to access mode, display format, and file organization. Table 11-1 summarizes the different uses of PF keys for each access mode and format. The list that follows the table describes all DISPLAY functions in greater detail.

Table 11-1. PF Key Functions According to DISPLAY Format

PF Key	Record Access Mode		Block Access Mode
	Record-Oriented Format	Report-Oriented Format	
1	Display/Menu	Display/Menu	Display/Menu
2	First	First	First
3	--	Position/Indices	--
4	Previous	Previous	Previous
5	Next	Next	Next
6	Down	Down	Down
7	Up	Up	Up
8	Find Record	Find	Find Record
9	Find Text	L Margin	Find Text
10	ASCII/Hex Mode	R Margin	ASCII/Hex Mode
11	Report Mode	Left 15	--
12	--	Right 15	--
13	--	Left 1	--
14	--	Right 1	
15	Print	Print	Print
16	End Processing	Exit	End Processing

Note: *The functions listed in Table 11-1 are described in the following list.*

REC Record-oriented format (record access mode)
REP Report-oriented format (record access mode)
BA Block access mode

PF Key	Function and Description
1	**Display** -- From the menu, switch to a screen that displays the input file in the specified access mode and format.
	Menu -- From the display screen, return to the menu.
2	**First** -- Display the first record (*REC*), the first 20 lines (*REP*), or the first block of data (*BA*) in the file.
3	**Position** (*REC*) -- Indicate the line or record number and column position of the cursor. PF3 has this function for consecutive or relative files only.
	Indices (*REC*) -- List the index (key) descriptions for an alternate indexed file. This option enables you to change the access path by selecting another key. PF3 has this function for indexed files only.
4	**Previous** -- Display the previous record (*REC*), the previous 20 lines (*REP*), or the previous block of data (*BA*) in the file.
5	**Next** -- Display the next record (*REC*), the next 20 lines (*REP*), or the next block of data (*BA*) in the file.
6	**Down** -- Move the display back (down) one record (*REC*), one line (*REP*), or one line within the block (*BA*).
7	**Up** -- Move the display forward (up) one record (*REC*), one line (*REP*), or one line within the block (*BA*).

PF Key	Function and Description
8	**Find Record** (*REC*) -- For an indexed file, move the display to a record located by key value; for a consecutive or relative file, move it to a record located by number. The user is prompted to enter the key value or record number. **Find** (*REP*) -- Move the display to a line located by number or by searching for a text string. The user is prompted to enter the line number or string. (For additional options, see the description of PF9, Find Text.) **Find Record** (*BA*) -- Move the display to a block located by number. The user is prompted to enter the hexadecimal block number.
9	**Find Text** (*REC*, indexed files only, and *BA*) -- Move the display to a record located by searching for a text string. The user is prompted to enter the text string. **L Margin** (*REP*) -- Shift the display to the first 80 columns of each line on the screen.
10	**Mode** (*REC* and *BA*) -- Change the display to hexadecimal mode if in ASCII or to ASCII mode if in hexadecimal. (See Section 11.4.7) **R Margin** (*REP*) -- Shift the display to the last 80 columns of each line on the screen.
11	**Report** (*REC*, consecutive or relative files only) -- Change the display from record-oriented format to report-oriented format. (See Section 11.3.2.) **Left 15** (*REP*) -- Move the display window 15 columns toward the beginning of the lines (shows text further to the left).
12	**Right 15** (*REP*) -- Move the display window 15 columns toward the end of the lines (shows text further to the right).
13	**Left 1** (*REP*) -- Move the display window one column toward the beginning of the lines (shows text further to the left).

PF Key	Function and Description
14	**Right 1** (*REP*) -- Move the display window one column toward the end of the lines (shows text further to the right).
15	**Print** -- Print all or part of the file. (See Section 11.3.3.)
16	**Exit** (*REP*) -- Change the display from report-oriented format to record-oriented format. (See Section 11.3.2.)
	End Processing (*REC* and *BA*) -- Stop displaying the current file. The screen that appears gives you the choice of displaying another file or exiting from DISPLAY.

Note: *When PF9 is pressed (or PF8 for a consecutive or relative file in record-oriented format), the following additional functions become available:*

PF2	***Set Search Defaults*** *-- This option allows you to set the Case field to UPPER to search for uppercase text or UPLOW to search for lowercase text. You can also set the Match field to EXACT to search for an exact text match, or ANY for a case-insensitive search.*
PF8	***Search backward*** *-- If you press PF8 after entering the text string, the display moves to the last occurrence of the string before the cursor position, instead of the next occurrence after it.* This function is not available for indexed files.

11.3.2 Changing Display Formats

When DISPLAY accesses a file using the record access mode, it displays the file in either record-oriented or report-oriented format. If the file is indexed, only record-oriented format is available, but with other kinds of files you can switch formats at your option.

The format in which a file is first displayed depends on the organization of the file:

- All *print files* (defined as all files with PRINT organization, regardless of contents or record size) and all *source files* (defined as all files with consecutive organization and 80-byte records, regardless of contents) are first displayed in report-oriented format.

- All other files, including consecutive files whose record length is greater or less than 80, are first displayed in record-oriented format.

To switch from record-oriented to report-oriented format, press PF11. The menu or display on your screen is replaced by the corresponding menu or display in report-oriented format.

To switch from report-oriented to record-oriented format, press PF16. The menu or display on your screen is replaced by the corresponding menu or display in record-oriented format.

Pressing the key while a menu is displayed changes the menu. Pressing the key while part of the file is displayed changes the file display. If you switch from one menu to the other and then press PF1, the part of the file last displayed (or the beginning, if you have not previously displayed it) appears in the new format.

Note: *PF11 is inactive and does not appear on the menu when you are displaying an indexed or alternate indexed file.*

11.3.3 Printing a Displayed File

To print all or part of a file as it appears in the display, press PF15 from the Functions menu or display screen. The Print Functions screen (Figure 11-3) appears.

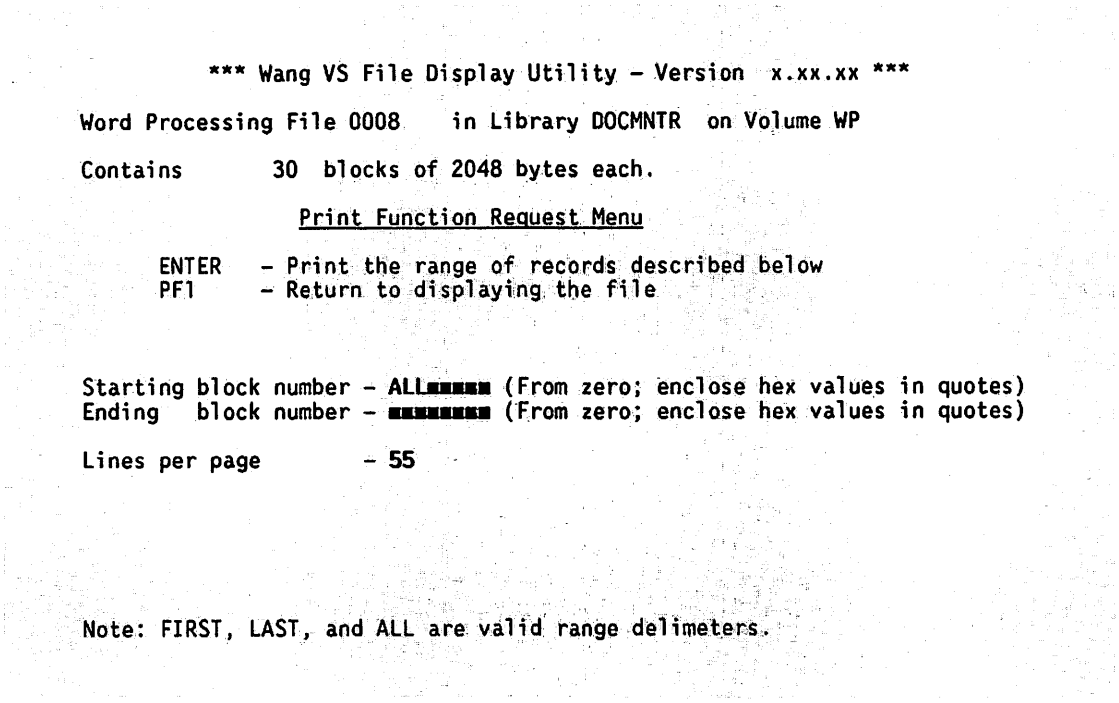


Figure 11-3. DISPLAY (Block Access) Print Functions Screen

The Print Functions screen varies slightly for each input display option available. Use this screen to specify the starting and ending record, line, or block numbers (or key values), and the number of lines per page to be printed.

Whether the file is printed immediately depends on the PRNTMODE default of your usage constants. If PRNTMODE is set to S, the file is automatically spooled for printing; otherwise, it is held until released for printing.

Enter information in the fields of the Print Functions screen as follows:

Field	Comments
Starting Number or Key Value	Specify the location at which the print file should begin, i.e., the beginning of the part of the file you want printed. The default value is ALL; if you accept this value it is unnecessary to specify an ending number. Otherwise, you can enter FIRST, or a number or key value appropriate to the access method, format, and file organization: • If the format is record-oriented for a consecutive or relative file, the starting number is a record number. • If the format is report-oriented for a consecutive or relative file, the starting number is a line number. • If block access is used, the starting number is a block number. *(If you specify a hexadecimal block number, you must enclose it in single quotation marks.)* • If record access is used for an indexed file, this field is Starting Key Value. You can enter any valid key value. *It must be enclosed in double quotation marks.* Press PF2 to enable DISPLAY to accept lowercase values. The file is printed in order of key values from the starting to the ending value specified in the next field.
Ending Number or Key Value	Specify the location at which the print file should end, i.e., the end of the part of the file you want printed. There is no default value for this field; if Starting Number or Key Value is ALL, it is unnecessary to specify an ending number. You can specify LAST, or a number appropriate to the access method, format, and file organization: • If the format is record-oriented for a consecutive or relative file, the ending number is a record number. • If the format is report-oriented for a consecutive or relative file, the ending number is a line number. • If block access is used, the ending number is a block number. *(If you specify a hexadecimal block number, you must enclose it in single quotation marks.)* • If record access is used for an indexed file, this field is Ending Key Value. You can enter any valid key value. *It must be enclosed in double quotation marks.* If you press PF2, DISPLAY accepts lowercase values.

Field	Comments
Lines per page	Specify the number of lines you want printed on each page. The default value is taken from the LINES field of your usage constants.

When you have supplied information as appropriate, press ENTER to begin the file printing operation. Alternatively, you can press PF1 to return to the Functions menu or the file display screen without printing.

11.4 DISPLAY FORMATS

The format in which a file is displayed varies according to access mode and file organization; in addition, you can choose record-oriented or report-oriented format for consecutive or relative files when they are displayed in record access mode. The following sections describe the six kinds of display that are possible. Section 11.4.7 describes ASCII and hexadecimal display modes.

11.4.1 Record Access, Consecutive File, and Record-Oriented Format

The file is displayed record by record. Records are displayed in the order in which they appear in the file. By pressing PF10, you can display the record in the ASCII or hexadecimal mode (see Section 11.4.7). The following example is a consecutive file in record-oriented format (ASCII representation):

```
RECORD 1
   0   13217BOS, BLANCHE              100 JERICHO RD          BUTLER
  64             NJ07405
RECORD 2
   0   14281CLEMENS, CORA             304 VASSAR RD           POUGHK
  64   EEPSIE    NY12648
RECORD 3
   0   14359FARNABY, ANNIE            243 23RD ST             CUYAHO
  64   GA FLS    OH44223
RECORD 4
   0   15692IONA, HELEN               23 E ELM AVE            QUINCY
  64             MA02170
```

11.4.2 Record Access, Consecutive File, and Report-Oriented Format

The file is displayed line by line; that is, each record is displayed on one line. Only 64 bytes of each record, therefore, are visible on the screen at one time, but the available PF key functions (see Section 11.3) allow the display window to be moved horizontally. The following example illustrates a consecutive file in report-oriented format:

```
13217BOS, BLANCHE    100 JERICHO RD      BUTLER          NJ07405
14281CLEMENS, CORA   304 VASSAR RD       POUGHKEEPSIE    NY12642
14359FARNABY, ANNIE  243 23RD ST         CUYAHOGA FLS    OH44223
15692IONA, HELEN     23  E ELM AVE       QUINCY          MA02170
```

11.4.3 Record Access, Relative File, and Record-Oriented Format

The file is displayed record by record. Records are displayed in the order in which they appear in the file. Empty record slots are not displayed. Zero-length records, however, are displayed; the term "<empty>" appears beneath the record number to indicate that the record contains no data. By pressing PF10, you can display the record in ASCII or hexadecimal mode (see Section 11.4.7). The following example illustrates a relative file in record-oriented format. (Because Record 2 represents an empty record slot, it is not displayed. Record 5 is a zero-length record.)

```
RECORD 1
   0   13217BOS, BLANCHE              100 JERICHO RD          BUTLER
  64             NJ07405
RECORD 3
   0   14281CLEMENS, CORA             304 VASSAR RD           POUGHK
  64   EEPSIE    NY12648
RECORD 4
   0   14359FARNABY, ANNIE            243 23RD ST             CUYAHO
  64   GA FLS    OH44223
RECORD 5
 <empty>
RECORD 6
   0   15692IONA, HELEN               23 E ELM AVE            QUINCY
  64             MA02170
```

11.4.4 Record Access, Relative File, and Report-Oriented Format

The file is displayed line by line as described in Section 11.4.1. <MISSING RECORD> indicates an empty record slot. <EMPTY RECORD> indicates a record that is present but contains no data (a zero-length record). The following example illustrates a relative file in report-oriented format:

```
13217BOS, BLANCHE     100 JERICHO RD      BUTLER          NJ07405
 <MISSING RECORD>
14281CLEMENS, CORA    304 VASSAR RD       POUGHKEEPSIE    NY12648
14359FARNABY, ANNIE   243 23RD ST         CUYAHOGA FLS    OH44223
 <EMPTY RECORD>
15692IONA, HELEN      23  E ELM AVE       QUINCY          MA02170
```

11.4.5 Record Access, Indexed File, and Record-Oriented Format

The file is displayed record by record in primary key sequence. By pressing PF3, you can change the access path of an indexed file that has alternate indexes.

Note: *In order to insure that the entire file is read, you should always specify the first record you want displayed when you change the access path or select a record by key or text string. If a particular record is not specified, DISPLAY always shows the next available record in an indexed file; it does not automatically return to the beginning.*

By pressing PF10, you can alternate between ASCII and hexadecimal modes (see Section 11.4.7). The following example illustrates an indexed file in record-oriented format:

```
KEY = 132
   0   13217BOS, BLANCHE              100 JERICHO RD          BUTLER
  64             NJ07405
KEY = 142
   0   14281CLEMENS, CORA             304 VASSAR RD           POUGHK
  64   EEPSIE    NY12648
KEY = 143
   0   14359FARNABY, ANNIE            243 23RD ST             CUYAHO
  64   GA FLS    OH44223
KEY = 156
   0   15692IONA, HELEN               23 E ELM AVE            QUINCY
  64             MA02170
```

11.4.6 Block Access

Records are displayed in 2-KB (2048-byte) blocks in logical block sequence, starting with the first block in the first extent of the file. Each line shows 64 bytes in ASCII mode or 32 in hexadecimal mode.

Because block access mode displays an exact image of each block of the file as it is recorded on disk, characters with no displayable ASCII equivalents (e.g., control characters, tabs, etc.) are represented on the screen, in ASCII mode, by various graphic characters. You can switch the display between ASCII and hexadecimal modes by pressing PF10. Both data blocks and index blocks are displayed for an indexed file. The following example illustrates a file in block format:

```
Block 0

  0   ¶13217BOS, BLANCHE              100 JERICHO RD          BUTLER
 64            NJ07405° ¶14281CLEMENS, CORA             304 VASSAR RD
128             POUGHKEEPSIE    NY12648° ¶14359FARNABY, ANNIE
192     243 23RD ST              CUYAHOGA FLS   OH44223° ¶15692IONA,
256   HELEN              23 E ELM AVE             QUINCY          MA
320   02170
...
...
...
...
...
1984
```

11.4.7 ASCII and Hexadecimal Display Modes

While you are displaying a file in block access mode or in record-oriented format, you can press PF10 to switch the display from ASCII to hexadecimal mode and back again.

In ASCII display mode, every byte in the file that is equivalent to a displayable character in ASCII (the American Standard Code for Information Interchange) is represented by that character on the screen. Bytes not equivalent to displayable ASCII characters are represented on the screen by dots or by various special characters whose identity depends on the configuration of your workstation. Each line shows 64 bytes, with (decimal) numbers across the top and at the left of the screen.

In hexadecimal display mode, every byte in the file is represented on the screen by two hexadecimal digits (from 00 through FF) that express its numeric value. This mode provides an accurate representation of data that is not ASCII-encoded. Since the hexadecimal mode uses two characters per byte where the ASCII mode uses one, the screen has room for less data when you switch from ASCII to hexadecimal. Each line shows 32 bytes, with (hexadecimal) numbers across the top and at the left of the screen.

The following examples show the same part of an indexed file in ASCII and then in hexadecimal mode:

```
KEY = 132
   0   13217BOS, BLANCHE              100 JERICHO RD          BUTLER
  64             NJ07405
KEY = 142
   0   14281CLEMENS, CORA             304 VASSAR RD           POUGHK
  64   EEPSIE    NY12648

KEY = 313332
0000  31333231 37424F2C 20424C41 4E434845 20202020 20202020 20202020 20313030
0020  204A4552 4943484F 20524420 20202020 20202020 20204255 544C4552 20202020
0040  20202020 20204E4A 30373430 35
KEY = 313432
0000  31343238 31434C45 4D454E53 2C20434F 52412020 20202020 20202020 20333034
0020  20564153 53415220 52442020 20202020 20202020 2020504F 5547484B 45455053
0040  49452020 20204E59 31323634 38
```

CHAPTER 12
FILEDISP

12.1 INTRODUCTION

The FILEDISP (File Name Display) utility enables you to display the names and locations of any files, libraries, and currently mounted volumes on your system (subject to the quantitative limit described in the note below). You can use wildcard characters to construct a mask and display all names that match its pattern. PF keys let you page through the resulting display and sort it by file, library, or volume name or any combination of the three.

When systems are linked through Resource Sharing Facility (RSF), FILEDISP searches and lists the volumes mounted on all attached systems, not just the local system.

Note: *FILEDISP is intended primarily for locating specific files and sets of files, not for listing the entire contents of extensive storage systems. A search specification loose enough to include all or most files could cause a long delay on a large system. To prevent such delays, FILEDISP is set to return no more than 2,000 entries for any one search specification.*

12.1.1 Running FILEDISP

Run FILEDISP from the Command Processor as you run any other program or utility. Enter FILEDISP into the program name field, the appropriate library in the library field, and the appropriate volume in the volume field. (Since this utility does not reside in the system library, you must enter the library and volume designations.)

The following sections describe FILEDISP processing:

Section	Process
12.2	Specifying a Search Mask
12.3	Examining and Sorting the Display

12.2 SPECIFYING A SEARCH MASK

When FILEDISP processing begins, the Mask Specification screen (Figure 12-1) appears.

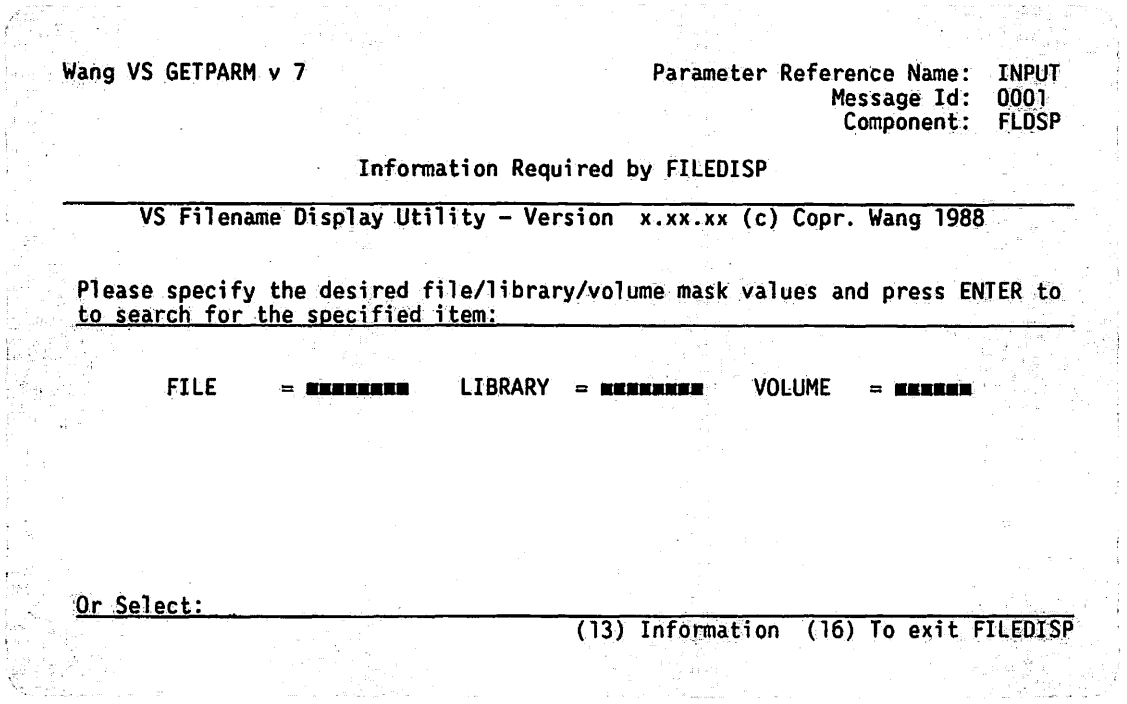


Figure 12-1. FILEDISP Mask Specification Screen

Use this screen to specify the mask or pattern of names to be searched for. The mask is determined by the information you enter in the fields.

To search for file names, enter values in all three. To search for library names, leave the FILE field blank, and to search for volume names leave both the FILE and LIBRARY fields blank.

You can focus the search on a specific range of names in any or all categories by combining alphanumeric characters with the wildcard characters ? and * in the mask specification. These characters have the following meanings:

? -- corresponds to any string of any length in the name. For example, if LIBRARY = *?XYZ?*, FILEDISP lists all libraries whose names contain the string *XYZ*, regardless of how many characters, if any, precede or follow it.

* -- corresponds to a single nonblank character in the name. For example, if LIBRARY = *, the program lists all (and only) the libraries on the specified volume with one-character names.

Blanks are ignored in the mask specifications: " *?PRT*", "*?PRT* " and "? *PRT*" all return exactly the same set of items.

Leaving a field completely blank specifies the next search level. For instance, if you leave FILE blank, FILEDISP displays a list of libraries. If you leave both FILE and LIBRARY blank, it displays a list of volumes. To list all the files in one or more libraries, you must set FILE to ?; to list all the libraries on one or more volumes, you must set LIBRARY to ?. Table 12-1 shows the results obtained from several kinds of mask specifications.

Table 12-1. Examples of File, Library, and Volume Mask Specifications Used with FILEDISP

File	Library	Volume	Items Located[a]
X	Y	Z	X, Y, and Z if file X exists in Library Y on Volume Z; otherwise, locates nothing
*	?	?	All one-character file names
?	?ABC?	?	All file names in every library whose name contains ABC
?	?	VOL123	All files on volume VOL123
Blank	#?PRT	SYSTEM	All print library names on volume SYSTEM
Blank	Blank	?	Names of all volumes currently mounted on the system or RSF grouping

[a] Up to a maximum of 2,000 eligible entries. See Section 12.1.

When you have entered specifications that will locate the file, library, and volume names you want, press ENTER.

You also have the following options from the Mask Specification screen:

- Pressing PF13 to display an information screen
- Pressing PF16 to terminate FILEDISP processing

12.3 EXAMINING AND SORTING THE DISPLAY

When FILEDISP has located the file, library, and volume names you asked for, it displays them on a FILEDISP Display screen similar to the example shown in Figure 12-2. The sample display was produced by entering the following specifications:

```
FILE = TE?
LIBRARY = ?
VOLUME = ?
```

The system had three volumes (ONE, TWO, and SYS) mounted.

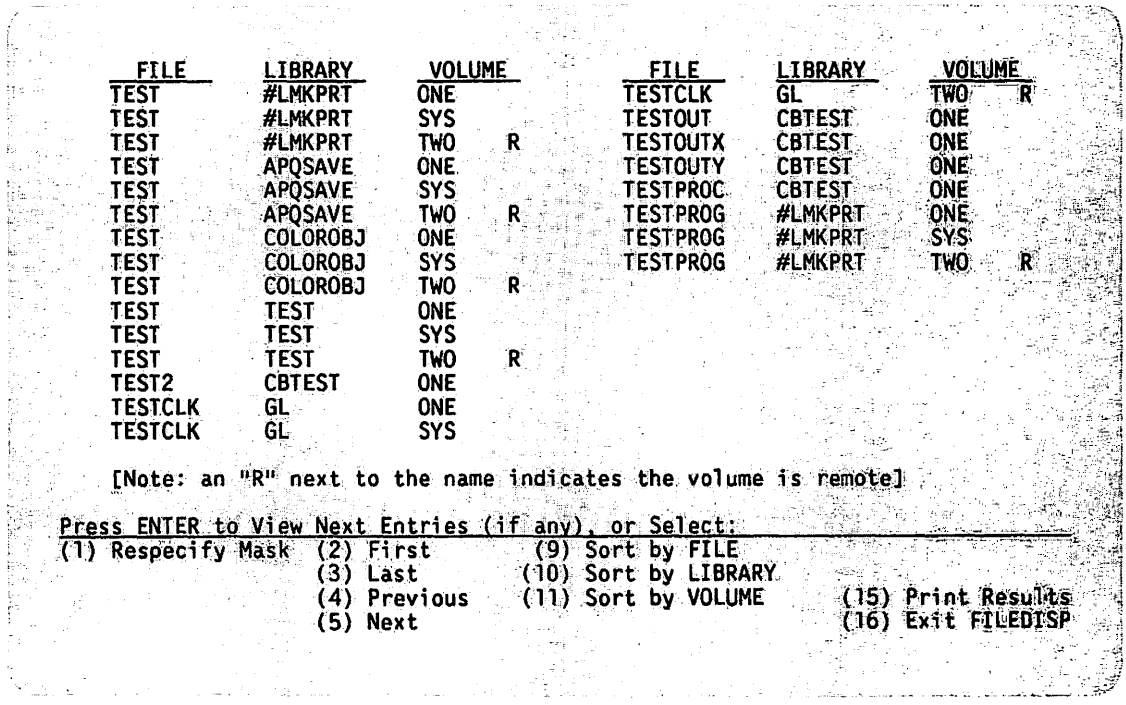


Figure 12-2. A Sample FILEDISP Display Screen

When FILEDISP is listing only volume names, or library and volume names, the screen appears the same, except that the unused columns are filled with dashes. Up to 30 file, library, or volume names are displayed at once. If there are more, they are displayed on successive screens, and you can use PF keys 2 to 5 (and ENTER) to move among screens:

PF2 Displays the first screen in the series
PF3 Displays the last screen
PF4 Moves the display back one screen
PF5 Moves the display forward one screen
ENTER

For remote volumes, an R appears to the right of the volume name.

PF keys 9 to 11 control the sorting of the display. When it first appears, the display is unsorted: file, library, and volume names are listed in the order in which FILEDISP encounters them. For a file name display, this means that files are grouped in libraries and libraries are grouped in volumes, but none of these groups is sorted internally.

PF9 Sorts the display in order of file names
PF10 Sorts it in order of libraries
PF11 Sorts it in order of volumes

By using these keys in combination, you can combine sorts. For example, if you press PF9, PF10, and PF11 in that order, the resulting display shows files sorted by volume, within volumes by library, and within libraries by name. The opposite sequence, PF11, PF10, and PF9, produces a display in which files are sorted by name, identically named files are sorted by library, and identically named libraries are sorted by volume. (This method was used to produce the display shown in Figure 12-2.)

You also have these options from the FILEDISP Display screen:

PF Key	Option and Description
1	**Respecify mask** -- Press PF1 to return to the Mask Specification screen to initiate a new search.
15	**Print results** -- When you press PF15, FILEDISP prints not merely the names that show on the screen, but all the names found in the current search (i.e., all the names you could display by using PF keys 2 to 5).
16	**Exit FILEDISP** -- Press PF16 to terminate the program.

CHAPTER 13
SELPRINT

13.1 INTRODUCTION

The SELPRINT (Select Print) utility enables you to specify a library and

- Send all the print files to the print queue at once
- Display the file names and select individual files to be treated in any of the following ways:
  - Print and dequeue
  - Print and requeue
  - Print and scratch
  - Scratch without printing

13.1.1 Running SELPRINT

Run SELPRINT from the Command Processor as you run any other program or utility. Enter SELPRINT in the program name field, the appropriate library in the library field, and the appropriate volume in the volume field. (Since this utility does not reside in the system library, you must enter the library and volume designations.)

The following sections describe SELPRINT processing:

Section	Process
13.2	Specifying the Print Library and Options
13.2.1	Managing Individual Files

13.2 SPECIFYING THE PRINT LIBRARY AND OPTIONS

When SELPRINT processing begins, the Print Library and Options screen appears. Figure 13-1 shows an example of this screen.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  OPTIONS
                                                          Message Id:  0001
                                                           Component:  SELPRT
                        Information Required by SELPRINT
_____________________________________________________________________________

          VS Selective Print Utility - Version  3.00.02 (c) Copr. Wang 1988

Please supply the library, volume and options and press ENTER to select files:

                    LIBRARY  = #USRPRT■    VOLUME   = WORK■■

                    PRNTMODE = H           ('H'old / 'S'pool)
                    PRTCLASS = A           ('A' thru 'Z')
                    COPIES   = 01

Or Select
(2) Print the entire library                          (16) To exit SELPRINT
```

Figure 13-1. A Sample SELPRINT Print Library and Options Screen

The user's spool library and volume appear as default values in the LIBRARY and VOLUME fields, and defaults for print mode, print class, and number of copies appear in the lower part of the screen. (All, including spool library and volume, are the default values regularly associated with the user or set by means of PF2 -- Set Usage Constants -- from the Command Processor menu.)

Any of the default values shown may be changed from this screen. When you have specified the library that contains the files you want to print and changed the values in other fields as appropriate, you have the following options:

- Press PF2 to print *all* the print files in the library. SELPRINT immediately places these files in the print queue with the disposition set to SAVE so that the files will not be scratched after printing. Files of other kinds are ignored.

- Press ENTER to display the file names and decide on the disposition of each file individually.

- Press PF16 to exit from SELPRINT without printing any files.

If you press PF2 to print the entire library, the Print Library and Options screen reappears after a short pause so that you can select another library, change options, or terminate SELPRINT.

If you press ENTER to dispose of individual files, the File Disposition screen (Figure 13-2) appears.

13.2.1 Managing Individual Files

When you press ENTER from Print Library and Options Specification screen, the first File Disposition screen appears. Figure 13-2 shows an example of this screen.

```
                        Selective Print Utility

                  LIBRARY = USRPRT      VOLUME = WORK
               PRNTMODE = H     PRTCLASS = A     COPIES = 01

     Please type one of the following to the left of each file to be printed
R = Print + Requeue  D = Print + Dequeue  S = Print + Scratch  X = Scratch now

     ■ FILE0001      ■ FILE0002      ■ FILE0003      ■ FILE0004
     ■ FILE0005      ■ FILE0006      ■ FILE0007      ■ FILE0008
     ■ FILE0009      ■ FILE0010      ■ FILE0011      ■ FILE0012
     ■ FILE0013      ■ FILE0014

Press ENTER to continue, or select:
(1) Return to respecify OPTIONS                          (16) Exit SELPRINT
```

Figure 13-2. A Sample SELPRINT File Disposition Screen

This screen displays the names of the files in the selected library in order of their creation, up to a maximum of 48. A pseudoblank appears to the left of each. You can enter a character from the following list at the pseudoblank opposite any file to determine the disposition of that file:

Character	Disposition of File
D	Print and remove from queue
R	Print and replace in queue
S	Print and scratch
X	Scratch immediately without printing

Any of these characters may be entered for any number of files. You can also leave any number of files unmarked so that no action will be taken on them. When you finish marking files, press ENTER.

One File Disposition screen shows as many as 48 file names from the specified library. When you press ENTER, the files you marked are processed in the manner you specified, and a File Disposition screen that displays the next set of files, if there are more, appears. Mark these in the same way and press ENTER. This process is repeated until SELPRINT has displayed the names of all files in the library.

When you press ENTER after marking the last set of files, the first set reappears, unchanged except that pseudoblanks have replaced the characters you entered, and the names of the files you scratched are absent.

You also have the following options from the File Disposition screen:

- Pressing PF1 to return to the Print Library and Options screen (Figure 13-1) so that you can specify another library or change print defaults. (The File Disposition screen shows the print defaults, but they can only be changed from the Print Library and Options screen.)

- Pressing PF16 to exit SELPRINT. You are returned to the VS Command Processor menu, or to the program or procedure from which SELPRINT was called.

CHAPTER 14
SORT

14.1 INTRODUCTION

The SORT utility can sort a file according to one or more key fields within the records. It can also merge two or more sorted files. Using these two capabilities, you can do any of the following jobs:

- Sort a single data file in an order you specify.
- Sort up to 10 data files into a single, ordered output file.
- Merge up to 20 data files into a single, ordered output file.
- Select specific records from one or more input files and sort them into a single, ordered output file.
- Produce an output file that contains only the primary index key field from each record in the input file. You can specify the sort order for this key field.
- Produce an ordered output file of 3-byte records to be used by RPG II programs.
- Reformat the record layout of the output file, modifying the length and sequence of fields, extending and padding the record, or truncating it so as to include only selected fields.
- Select the file organization, packing factors, maximum record size, and record type of the output file. If you create an indexed output file, you can also specify alternate keys.

SORT can retrieve input files from tape volumes (which have only consecutive files) or disk volumes (which have consecutive, indexed, or relative files). Input files are left intact unless you assign the same name to the input and output files -- doing this causes the input file to replace the output file.

The *Sort function* takes an unordered file or files and arranges them in a specified order, producing an ordered output file whose organization and record description you can specify. When you sort two or more files, SORT automatically merges them; the output is always a single file.

The *Merge function* takes two or more files that already have the same format and order, and combines them into a single ordered, consecutive output file. Both files must be sorted before being merged. If SORT detects an unsorted file while attempting to merge it with another, the program is terminated.

Both Sort and Merge functions allow you to extract specific input records or fields within records for processing.

14.1.1 Overview

To do SORT processing, you specify the following items in the order listed:

1. The program function (Sort or Merge)
2. The collating sequence
3. The input file or files to be sorted or merged
4. The keys on which to sort or merge the files
5. Optionally, a new format for the output record
6. The output file that will receive the sorted data

14.1.3 Running SORT

To run the SORT utility, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify SORT in the Program field and press ENTER.

The following sections describe SORT processing:

Section	Process
14.2	Specifying SORT Options
14.2.1	Output Options for a Sorted File
14.2.2	Modifying the Collating Sequence
14.3	Specifying Input Files To Be Sorted or Merged
14.3.1	Sorting Files in Shared Mode
14.3.2	Defining Record Selection Criteria
14.4	Defining the Sort or Merge Keys
14.5	Specifying the Output Record Format
14.6	Specifying the Output File
14.6.1	Primary Keys for Indexed Files
14.6.2	Specifying Alternate Index Keys
14.7	Restarting the SORT Utility
14.8	A Sample SORT Procedure

14.2 SPECIFYING SORT OPTIONS

When SORT processing begins, the Options screen (Figure 14-1) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  OPTIONS
                                                          Message Id:  0000
                                                           Component:  SORT

                        Information Required by SORT

You are now running Wang VS Sort/Merge Program, Version xx.xx.xx
Please specify the program options below:

          FUNCTION = SORT     (SORT or MERGE)
          MEMORY   = 128 K    (Memory used, Maximum size = 174 K)

Do you want an ADDROUT output file?      ADDROUT  = NO     (YES or NO)
Do you want a KEYOUT output file?        KEYOUT   = NO     (YES or NO)
Do you require a STABLE sort?            STABLE   = NO     (YES or NO)
Do you want to REFORMAT the records?     REFORMAT = NO     (YES or NO)
Type of alternate collating sequence:    ALTSEQ   = NONE   (EBCDIC, ASCII,
                                                             TABLE, or NONE)

                        Press PF16 to terminate SORT

                        (C) COPR. WANG LABS, INC., 1985
```

Figure 14-1. SORT Options Screen

Use this screen to specify the Sort or Merge function, the amount of memory to be used for the Sort or Merge operation, and the options that determine the nature of the output file. To terminate SORT processing without specifying program options, press PF16.

Enter information in the fields of the Options screen as follows:

FUNCTION -- Specify the function (SORT or MERGE) that you want SORT to perform. The Sort function sorts one or more files into a single ordered output file. The Merge function merges two or more ordered files into a single sorted output file. The default value is SORT.

MEMORY -- Specify the amount of main memory you want allocated for processing the Sort or Merge operation. The greater the amount of memory used, the more efficient the operation. For most operations, 128 KB (the default value) is sufficient. If that much main memory is unavailable, the system determines the amount of memory actually available and uses it in running the program.

You can enlarge the work area beyond 128 KB to improve Sort or Merge processing time, but this increases competition with other users for main memory and may degrade system performance. The maximum amount of main memory you can allocate (which depends on the size of physical memory in the system) is displayed next to the Memory field on the Options screen.

The optimum amount of main memory for each Sort or Merge operation can only be determined by experimenting with the actual operating environment. If the Sort or Merge operation requires a large amount of processing time or main memory, it is best to run it in batch mode during the off-peak hours of system use so that other users are not affected.

ADDROUT -- If you specify YES in the Addrout field, SORT produces a single-field consecutive file of sorted, 3-byte records that can be used by RPG II programs. Each record is a positive binary number representing the relative number of a record in the input file. When you do a sort with the Addrout option, these numbers are the *only* output. For an example of the Addrout option, see Section 14.3.1. For more information about Addrout files, see the *VS RPG II Language Reference*.

You cannot use the Addrout field with the Merge function, nor can you specify YES in both the Addrout and Keyout fields. Addrout can be combined with a Stable sort. The default value for Addrout is NO.

KEYOUT -- Specify YES in the Keyout field to produce a single-field consecutive file consisting of the primary keys of the records in the order specified by the sort criteria. The input file must be indexed.

You cannot use the Keyout field with the Merge function, nor can you specify YES in both the Addrout and Keyout fields. Keyout can be combined with a Stable sort. The default value for Keyout is NO. For an example of the Keyout option, see Section 14.2.1.

STABLE -- Specify YES in the Stable field to produce an output file in which records with identical sort keys remain in the same order as the input order sequence. The default value for Stable is NO. For an example of the Stable option, see Section 14.2.1.

REFORMAT -- Specify (YES or NO) whether you want SORT to reformat the record layout of the output file -- i.e., to arrange the fields in a different order than that of the input file. If you specify reformatting, SORT later displays a screen from which you can specify a new order for the fields (see Section 14.5). The default value is NO.

You can use the Reformat option in conjunction with the Stable option, but not with the Addrout or Keyout option. The Reformat option is not available with the Merge function, and it applies to consecutive output files only; do not select it if you intend to specify an indexed output file.

ALTSEQ -- The value in this field indicates whether you want an alternate collating sequence for character sort keys. You can specify EBCDIC, ASCII, TABLE, or NONE. NONE, the default value, produces a sort according to the ASCII collating sequence, which you cannot modify. If you specify ASCII, EBCDIC, or TABLE, a collating sequence table appears and you can modify the sort sequence by changing the entries in the table. For more information about the collating sequence table option, see Section 14.2.2.

Table 14-1 shows which combinations of program functions and output options are possible.

Table 14-1. SORT Utility Function/Option Matrix Table

Option	Program Function		Output Option				
	SORT	MERGE	ADDROUT	KEYOUT	STABLE	REFORMAT	ALTSEQ
ADDROUT	YES	NO	-	NO	YES	NO	YES
KEYOUT	YES	NO	NO	-	YES	NO	YES
STABLE	YES	YES	YES	YES	-	YES	YES
REFORMAT	YES	NO	NO	NO	YES	-	YES
ALTSEQ	YES	YES	YES	YES	YES	YES	-

Enter or change information in the fields of the Options screen as appropriate, and press ENTER to continue SORT processing.

14.2.1 Output Options for a Sorted File

The following examples illustrate the output options (Addrout, Keyout, and Stable) for a sorted file.

Sample Addrout Output File

An Addrout output file is a consecutive file consisting of 3-byte records that represent the sequence numbers of the records in the input file. In the following sample, the input is a consecutive file to be sorted on the Code field in ascending order. For example, the name Varkonyi is in Record Number 008; it is listed first in the output file because it has the lowest value (1110) in the Code field, which is the sort field. The name Kaplan is in Record Number 003; it is listed second because it has the second lowest value (1120) in the Code field.

Sorted Input File				**Addrout Output File**
Record Number	**Name**	**Code**	**Address**	**Record Sequence After Sort by Code**
001	Busby	2221	Greenwich St	008
002	Isaac	8677	Pawtucket Rd	003
003	Kaplan	1120	Roman Rd	005
004	Mayer	3001	E. 67th St	001
005	Metcalfe	2220	Sachem St	007
006	Pascucci	6676	W Haven	004
007	Sterling	2231	Kiel Ave	006
008	Varkonyi	1110	Beacon St	002

Note: *The SORT utility cannot produce Addrout files from a Merge function or from multiple input files.*

Sample Keyout Output File

In the following sample, the input is an indexed file with three key fields: Name, Code, and Address. The primary index key is Name. When the input file is sorted on the Code field in ascending order, SORT finds the lowest value in that field and places the corresponding primary index key (Name) in the Keyout output file. Since Varkonyi has the lowest value (1110) in the Code field, for example, the contents of the primary index key field for that record become the first record in the output file.

Sorted Input File			**Keyout Output File**
Name	**Code**	**Address**	**Name**
Busby	2221	Greenwich St	Varkonyi
Isaac	8677	Pawtucket Rd	Kaplan
Kaplan	1120	Roman Rd	Metcalfe
Mayer	3001	E. 67th St	Busby
Metcalfe	2220	Sachem St	Sterling
Pascucci	6676	W Haven	Mayer
Sterling	2231	Kiel Ave	Pascucci
Varkonyi	1110	Beacon St	Isaac

Sample Stable Output File

Input Files 1 and 2 are to be sorted in *descending order*, using the Code field as the primary sort key. Note the order of the entries that have identical sort keys in the input files. Then note the order in which they are placed in the output file depending on the Stable option selected.

If you specify YES in the Stable field, all output records with the same sort key value remain in the sequence of their occurrence in the input files. For example, Johnson and Moyer, in Input File 1, have the same code as Metcalfe, in Input File 2. SORT places the records in the output file in the sequence of their occurrence in the first and then the second input file.

When the value in the Stable field is NO, however, the order of these three records in the output file has no necessary connection with their sequence in the input files.

Input File 1

Name	Code	Address
Crawford	1110	Foster Ave
Johnson	2220	Jericho St
Moyer	2220	Elm St
Purchis	2221	Ritter St
Spencer	3001	Speakman Dr

Input File 2

Name	Code	Address
Busby	2221	Greenwich St
Isaac	8677	Pawtucket Rd
Kaplan	1120	Roman Rd
Mayer	3001	E. 67th St
Metcalfe	2220	Sachem St
Pascucci	6676	W Haven
Sterling	2231	Kiel Ave
Varkonyi	1110	Beacon St

Input File 1 sorted with Input File 2 creates this output file when Stable = YES.

Name	Code	Address
Isaac	8677	Pawtucket Rd
Pascucci	6676	W Haven
Spencer	3001	Speakman Dr
Mayer	3001	E. 67th St
Sterling	2231	Kiel Ave
Purchis	2221	Ritter St
Busby	2221	Greenwich St
Johnson	2220	Jericho St
Moyer	2220	Elm St
Metcalfe	2220	Sachem St
Kaplan	1120	Roman Rd
Crawford	1110	Foster Ave
Varkonyi	1110	Beacon St

Input File 1 sorted with Input File 2 creates this output file when Stable = NO.

Name	Code	Address
Isaac	8677	Pawtucket Rd
Pascucci	6676	W Haven
Spencer	3001	Speakman Dr
Mayer	3001	E. 67th St
Sterling	2231	Kiel Ave
Busby	2221	Greenwich St
Purchis	2221	Ritter St
Moyer	2220	Elm St
Metcalfe	2220	Sachem St
Johnson	2220	Jericho St
Kaplan	1120	Roman Rd
Varkonyi	1110	Beacon St
Crawford	1110	Foster Ave

Note: *When you specify NO in the Stable field, records are sorted only by the fields that you specify. No specific order is imposed for the sort of records with duplicate sort keys. As a result, the order of the sorted output for the same two input files may differ for successive sorts, but only for fields that were not specified as sort key fields.*

14.2.2 Modifying the Collating Sequence

If you specify ASCII, EBCDIC, or TABLE in the ALTSEQ field of the Options screen (Figure 14-1), the Collating Sequence Table screen (Figure 14-2) appears when you press ENTER from the Options screen. The default table shown on the screen depends on your specification. Figure 14-2 shows a sample of a default collating sequence table.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  TABLE
                                                 Message Id:  0000
                                                  Component:  SORT

                        Information Required by SORT

Please modify as desired and/or select from choices listed:

    HEX00#0F = 00010203 04050607 08090A0B 0C0D0E0F     ENTER - Continue
    HEX10#0F = 10111213 14151617 18191A1B 1C1D1E1F
    HEX20#0F = 20212223 24252627 28292A2B 2C2D2E2F     PF  1 - Return to
    HEX30#0F = 30313233 34353637 38393A3B 3C3D3E3F           OPTIONS screen
    HEX40#0F = 40414243 44454647 48494A4B 4C4D4E4F
    HEX50#0F = 50515253 54555657 58595A5B 5C5D5E5F     PF  2 - Change to
    HEX60#0F = 60616263 64656667 68696A6B 6C6D6E6F           ASCII-EBCDIC
    HEX70#0F = 70717273 74757677 78797A7B 7C7D7E7F
    HEX80#0F = 80818283 84858687 88898A8B 8C8D8E8F     PF  3 - Change to
    HEX90#0F = 90919293 94959697 98999A9B 9C9D9E9F           EBCDIC-ASCII
    HEXA0#0F = A0A1A2A3 A4A5A6A7 A8A9AAAB ACADAEAF
    HEXB0#0F = B0B1B2B3 B4B5B6B7 B8B9BABB BCBDBEBF     PF  4 - Set to
    HEXC0#0F = C0C1C2C3 C4C5C6C7 C8C9CACB CCCDCECF           Default TABLE
    HEXD0#0F = D0D1D2D3 D4D5D6D7 D8D9DADB DCDDDEDF
    HEXE0#0F = E0E1E2E3 E4E5E6E7 E8E9EAEB ECEDEEEF     PF  13 - Help
    HEXF0#0F = F0F1F2F3 F4F5F6F7 F8F9FAFB FCFDFEFF
```

Figure 14-2. SORT Collating Sequence Table Screen

The Collating Sequence Table screen enables you to customize a collating sequence table in order to modify the criteria by which the input files are sorted. When new values are entered in the table, the corresponding characters are repositioned in the collating sequence.

Note: *Characters in the input or the output file are not affected by changes to the table. Changes to the table affect only the criteria by which the file is sorted.*

The ASCII, EBCDIC, and TABLE options are described in the sections that follow.

You also have these options from the Collating Sequence Table screen:

PF Key	Function and Description
1	**Return to Options screen** -- Press PF1 to redisplay the Options screen without making any changes in the default collating sequence.
2	**Change to ASCII - EBCDIC** -- Pressing PF2 has the effect of changing the specification in the ALTSEQ field of the Options screen to EBCDIC (sort an ASCII table in EBCDIC collating sequence). If the default table for this option is already displayed, PF2 has no effect except to cancel any changes you have made.
3	**Change to EBCDIC - ASCII** -- Pressing PF3 has the effect of changing the specification in the ALTSEQ field of the Options screen to ASCII (sort an EBCDIC table in ASCII collating sequence). If the default table for this option is already displayed, PF3 has no effect except to cancel any changes you have made.
4	**Set to Default Table** -- Pressing PF4 has the effect of changing the specification in the ALTSEQ field of the Options screen to TABLE (modify an ASCII collating table). If the default table for this option is already displayed, PF4 has no effect except to cancel any changes you have made.
13	**Help** -- Press PF13 to display a screen that summarizes the information in this section.

TABLE Collating Sequence

If you specify TABLE in the ALTSEQ field of the Options screen (Figure 14-1), the Collating Sequence Table screen displays the full set of ASCII codes in hexadecimal notation. To modify the table:

1. Find the hexadecimal number that represents the character that you want to reposition in the collating sequence.

2. Specify the hexadecimal number of the character that indicates the new collating position of the character.

If, for example, you change 61 in row HEX60#6F to 41, you indicate that the character *a* is to be sorted equally with the character A. You can make the entire sort case-insensitive by changing the hexadecimal numbers in rows HEX60#6F and HEX70#7F so that they run from 41 through 5A (like the numbers in rows HEX40#4F and HEX50#5F) instead of from 61 through 7A. This causes SORT to treat upper- and lowercase letters as identical.

ASCII to EBCDIC Collating Sequence

If you specify EBCDIC in the ALTSEQ field, the default collating sequence table is set to sort an ASCII input file using an EDCDIC collating sequence. The output file is an ASCII file. The column on the left side of the screen (HEX00#0F) represents the ASCII hexadecimal codes 00 through 0F in the first row, 10 through 1F in the second row, and so on. The table displays the EBCDIC hexadecimal equivalent of the ASCII hexadecimal codes. (For example, the default table shows that ASCII 41 (the second modifiable hexadecimal number in row 5, which represents the character A) has a value of C1 in EBCDIC.)

The letters of the alphabet naturally have the same order in EBCDIC as in ASCII code, but there are major differences between the two collating sequences nevertheless. The digits *0 - 9*, which come *before* any letters in ASCII sequence, come *after* all letters in EBCDIC sequence. Similarly, the uppercase letters *A - Z*, which *precede* the lowercase letters *a - z* in ASCII sequence, *follow* them in EBCDIC sequence. You can manipulate the results of the sort by taking advantage of such differences, or if you wish you can minimize them by changing the values in the default table.

EBCDIC to ASCII Collating Sequence

If you specify ASCII in the ALTSEQ field, the default collating sequence table is set to sort an EBCDIC input file using an ASCII collating sequence. The output file is an EBCDIC file. The column on the left-hand side of the screen (HEX00#0F) represents the EBCDIC hexadecimal codes 00 through 0F in the first row, 10 through 1F in the second row, and so on. The table displays the ASCII hexadecimal equivalent of the EBCDIC hexadecimal codes. For example, the default table shows that EBCDIC C1 (the second modifiable hexadecimal number in row 13, representing the character A) has a value of 41 in ASCII.

The previous section, "ASCII to EBCDIC Collating Sequence," discusses differences between the ASCII and EBCDIC collating sequences and their consequences for the output of a sort.

Specifying the Collating Sequence Table Through a Procedure

If you run the SORT utility from a procedure, it must specify each field of the Collating Sequence Table screen (Figure 14-2) as a whole, the way it is shown on the screen. For example, if you want to change hexadecimal 61 through 7A to hexadecimal 41 through 5A using a procedure, valid ENTER statements are written as follows:

```
ENTER TABLE HEX60#6F = "60414243 44454647 48494A4B 4C4D4E4F",
            HEX70#7F = "50515253 54555657 58595A7B 7C7D7E7F"
```

Even when you are changing only one value in a row, the ENTER statement must specify the entire row, including the default values of the other 15 bytes.

Quotation marks are required only when a space is used as a filler character. Other filler characters are acceptable. For more information about the VS Procedure language, refer to the *VS Procedure Language Reference*.

14.3 SPECIFYING INPUT FILES TO BE SORTED OR MERGED

After you finish specifying the program options, the first Input File Specification screen (Figure 14-3) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  INPUTNS
                                                          Message Id:  0000
                                                           Component:  SORT

                        Information Required by SORT

Please enter name of the file to be sorted.

INPUT FILE     = ________ in LIBRARY  = ________ on VOLUME  = ______

Is this a SHARED file?                    SHARED   = NO_ (YES or NO)
Do you want to select input records?      SELECT   = NO_ (YES or NO)
Do you have more input files?             MOREFILE = NO_ (YES or NO)


FILE INPUT DEVICE   = DISK    (DISK or TAPE)

If input device is tape, then enter file sequence number   FILESEQ  = 1___
and the maximum number of input records.                   RECORDS  = 1000__
```

Figure 14-3. SORT Input File Specification Screen

Use this screen to specify the file, library, and volume names of an input file you want to process. Other options enable you to specify whether you want to open a consecutive or indexed file in Shared mode, whether you want to select records from the file instead of sorting or merging all the records it contains, and whether you want to process more than one input file in a SORT operation.

Enter information in the fields of the Input File Specification screen as follows:

INPUT/FILE -- The name of the first or next input file you want to sort or merge. You can specify up to 10 input files for the Sort function or 20 for the Merge function. The Input File Specification screen automatically reappears until you press ENTER from an unaltered screen. *(Note that, for the Sort function, you must specify YES in the Morefile field of the first Input File Specification screen if you want to process more than one input file; see the description of that field.)*

LIBRARY -- The name of the library in which the input file resides. The default value is taken from the INLIB field of your usage constants.

VOLUME -- The name of the volume on which the input file resides. The default value is taken from the INVOL field of your usage constants.

SHARED -- Specify YES or NO to indicate whether you want to open consecutive and indexed input files in Shared mode. If you enter YES, the Lock screen appears next and you must specify the fields on that screen. (See Section 14.3.1.) If you enter NO, the input file is opened in Input mode. The default value is NO.

SELECT -- Specify YES or NO to indicate whether you want to select specific input records for use in a Sort or Merge operation. If you enter YES, the Select screen appears to prompt you for the selection criteria. (For more information on defining selection criteria, see Section 14.3.2.) If you enter YES, only the records that match the selection criteria you specify are used to produce the output file. If you enter NO, the Select screen does not appear, and all the records in the input file are processed. The default value is NO.

MOREFILE -- Specify YES or NO to indicate whether you want to sort more than one input file into a single output file. (The sorted files are also merged.) If you enter YES, you must specify all the input files before SORT processing can continue. Additional Input File Specification screens (up to a maximum of 10) automatically appear for this purpose. (The additional screens do not include the Select and Morefile options, since these options are specified only once, from the initial Input File Specification screen.)

It is not necessary to specify the Morefile field for the Merge function, which implies multiple input files. If you specified a Merge operation from the Options screen, additional Input File Specification screens will appear automatically (up to a maximum of 20), regardless of the value in this field.

When you finish entering the names of files to be sorted or merged, do not enter anything when the next Input File Specification screen appears; simply press ENTER from the blank screen to terminate the file specification process.

FILE INPUT DEVICE -- Specify the type of device (DISK or TAPE) on which the input file resides. If you enter DISK, no further information is needed to complete the input specification, but if you enter TAPE, you must also specify the file sequence number in the Fileseq field and the maximum number of input records in the Records field.

FILESEQ -- If you specified TAPE in the File Input Device field, enter the sequence number of the input file on the tape. The default value is 1.

RECORDS -- If you specified TAPE in the File Input Device field, enter the approximate number of records to be processed from this input file. If you underestimate the number of records in a file, the operation may halt and have to be restarted (see Section 14.7). The smaller the overestimation, however, the more efficient the operation. The default value is 1000.

14.3.1 Sorting Files in Shared Mode

If you specify Shared mode for an input file, a Lock screen (Figure 14-4) appears as soon as you press ENTER from the Input File Specification screen (Figure 14-3).

```
Wang VS GETPARM v 7                          Parameter Reference Name:  LOCK
                                                          Message Id:  0000
                                                           Component:  SORT

                        Information Required by SORT

Please enter the following options to process a file in shared mode.

Should a lock be placed on the file before copying?    LOCK     = YES (YES, NO)
  If YES, No changes to the file can be made during
  the sort.  If NO, changes to the file can be made.

If LOCK=YES, specify TIMEOUT and BYPASS:

How many seconds should the timeout be?                TIMEOUT  = 10
  (If TIMEOUT=NO, sort will wait
   until the file is available).

If the timeout expires, should the file be bypassed?   BYPASS   = NO (YES, NO)
```

Figure 14-4. SORT Utility Lock Screen

The Lock screen prompts you to specify YES or NO to indicate whether a lock should be placed on the file before it is sorted so that no changes can be made to the file during the sort operation.

Enter information in the fields as follows:

LOCK -- Specify YES or NO to indicate whether you want to suspend updates to a file that you are sorting. If you lock a file (YES), no changes can be made to the file while you are sorting it. If you enter NO, other users can change the file during the operation, and there is no need to specify the Timeout and Bypass options. The default value is YES.

TIMEOUT -- Specify a Timeout value (0 to 255 seconds) for a file if Lock = YES. The Timeout value is the length of time for the SORT utility to wait for another user to release the file. The default value is 10 seconds.

BYPASS -- Specify YES or NO to indicate whether you want SORT to skip the file if a timeout occurs (i.e., when the time specified in the Timeout field expires before the file is released). The default value is NO.

If BYPASS = YES and a timeout occurs, SORT skips the file.

If BYPASS = NO and a timeout occurs, the Lock screen reappears with the message "FILE filename IN libname ON volname IS HELD BY USER xxx."

When a timeout occurs, you have a chance to redefine the Lock, Timeout, and Bypass options, and then press ENTER to continue with SORT operation. You also have the following options:

- You can skip the particular file on which the timeout occurred by pressing PF1.

- You can terminate the SORT operation by pressing PF16.

14.3.2 Defining Record Selection Criteria

If you specify YES in the Select field of the Input File Specification Screen (Figure 14-3), the SORT utility selects the records to be sorted or merged on the basis of the relationships you defined from the Record Selection screen, which appears when you press ENTER from the Input File Specification screen. Figure 14-5 shows an example of a SORT Record Selection screen. (If you have also specified Shared mode for the same input file, the Lock screen, described in the previous section, appears first.)

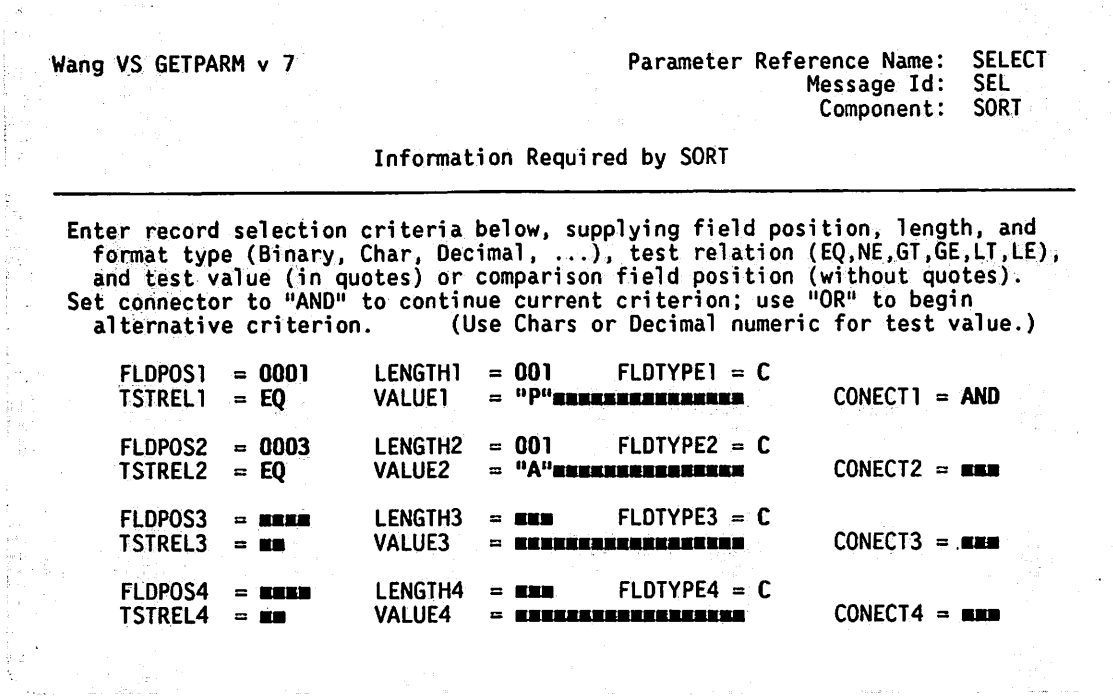


Figure 14-5. Sample SORT Record Selection Screen

Individual input records are selected by testing them against values you specify for comparison from the Record Selection screen. For example, suppose you are processing a payroll file and only want to sort the records of part-time employees. In this file, the records of part-time employees all have the letter *P* in the first byte of each record. Therefore the test criterion is that all input records to be processed must have the value *P* (X'50') in Position 1. If the record matches the test criterion, it is selected for sorting. If not, it is neither sorted nor merged and is not placed in the output file.

Besides being tested against a literal value, one field can also be tested against another field in the same record. For example, you can specify that if the four bytes beginning at Position 12 exceed in value the four bytes beginning at Position 43, the record can be processed. You can also combine multiple criteria by using the optional CONECTx field (where x is a number) to connect successive criteria with a logical AND or OR.

In the sample Record Selection screen illustrated in Figure 14-5, the user wants to sort only those payroll records that indicate *active* part-time employees. The test criteria therefore specify that all input files must have *P* in Position 1 and an A in Position 3 (indicating Active status).

The SORT utility processes test statements in the following ways:

- If Statement 1 is true *OR* Statement 2 is true (either can be true), the record is selected.

- If Statement 1 is true *AND* Statement 2 is true (both must be true), the record is selected.

The fields in the following list represent the components of a single test statement or criterion, followed by an optional connector to the next criterion. Each set of fields on the screen shares a numeric suffix, represented by an *x* in the list (FLDPOSx, etc.) You can specify as many as 32 selection criteria for a single Sort or Merge operation. Four criteria can be defined from each Record Selection screen. If the last CONECTx field on the screen is specified, SORT displays another screen, until the maximum is reached. An unspecified CONECTx field indicates that all record selection criteria have been defined.

Enter information in the fields as follows:

FLDPOSx -- Specify the position in the record of the first byte of the field you intend to test. (The first byte of the record occupies Position 1.) When you are specifying multiple criteria, you do not have to specify field positions in ascending numeric order. For example FLDPOS1 = 5, FLDPOS2 = 3 is just as valid as FLDPOS1 = 3, FLDPOS2 = 5. There is no default value for this field.

LENGTHx -- Specify the number of bytes in the field you want tested, beginning with the byte you specified for the field position (FLDPOSx). For example, if FLDPOS1 = 12 and LENGTH1 = 4, the field tested consists of bytes 12, 13, 14, and 15. Depending on the way the criterion is defined, there are limits on the number of bytes that can be tested; this condition imposes an effective limit on the valid length of the test field. See the description of the VALUEx field for further information. LENGTHx has no default value.

FLDTYPx -- Specify the data type of the field being tested and the criterion specified in the VALUEx field. The default value for FLDTYPx is C. The field type options are

C	Character data (alphanumeric)
B	Binary
D	External decimal signed or unsigned; sign trailing or leading in separate byte
L	Zoned decimal, sign leading included in first byte (overpunched)
P	Packed decimal, sign trailing included in last byte
Z	Zoned decimal, sign trailing included in last byte

TSTRELx -- Specify the type of comparison to be made between the input data (starting at FLDPOSx) and VALUEx. There is no default value for TSTRELx. The test relationship options are

EQ Equal
NE Not equal
GT Greater than
LT Less than
GE Greater than or equal to
LE Less than or equal to

VALUEx -- Specify the value to be compared with the field in each input record, beginning at FLDPOSx and extending for LENGTHx bytes. VALUEx can be a literal data value enclosed in single or double quotation marks, or it can be the starting position of another field in the record.

Note: *You must enclose* literal *data values in quotation marks, even if they are numbers; otherwise, VALUEx is interpreted as a field position. If you intentionally specify the starting position of another field in the record, this field must be of the same type and length as the original field. Specifying a field whose type and length do not match LENGTHx and FLDTYPx or omitting quotation marks from a literal data value may result in an erroneous output file or an aborted operation.*

You can specify a maximum of 16 bytes for a literal data value. For field-to-field comparisons, the maximum depends on the data type:

C 256
B 2 or 4
D 16
L 16
P 16
Z 16

There is no default value for this field.

CONNECTx -- If you intend to define another criterion, specify the logical connector (AND or OR) you want to use to combine it with the one you have just defined. Since you can specify up to 32 comparisons for each Sort or Merge operation, the maximum number of ANDs and ORs that you can use for any one operation is 31. CONECTx is an optional field whose default value is blank. The first blank CONECTx field signifies that all selection criteria have been defined.

Figure 14-6 shows a partial payroll file with part-time and full-time employees who are either active or inactive. Figure 14-7 shows the partial payroll file sorted on the following test criteria, designed to select all active part-time employees:

```
FLDPOS1 = 1  LENGTH1 = 1  FLDTYP1 = C  TSTREL1 = EQ  VALUE1 = "P"  CONECT1 = AND
FLDPOS2 = 3  LENGTH2 = 1  FLDTYP2 = C  TSTREL2 = EQ  VALUE2 = "A"
```

Employee Name	Employment Hours	Employee Status
Frank Franklin	F	A
Eric Ericson	F	I
Quinton Quigly	F	A
Ivan Ives	P	A
Bill Billings	F	A
Pat Paterson	P	A
Neal Neilson	F	I
Kendra Kendall	F	A
David Davidson	P	I
Harry Harrison	P	A
Louise Louis	F	A
Joel Johnson	P	A
Ann Anderson	F	I
Matt Matthews	P	A
Carl Carlson	P	A
Gary Garrison	F	A
Ollie Oliver	P	I

Figure 14-6. Partial Payroll File Before Selective Sort

Employee Name	Employment Hours	Employee Status
Carl Carlson	P	A
Harry Harrison	P	A
Ivan Ives	P	A
Joel Johnson	P	A
Matt Matthews	P	A
Pat Paterson	P	A

Figure 14-7. Partial Payroll File After Selective Sort

14.4 DEFINING THE SORT OR MERGE KEYS

After you have specified the input file and the record selection criteria, if any, the Sort/Merge Keys screen (Figure 14-8) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  KEYS
                                                          Message Id:  0001
                                                           Component:  SORT

                        Information Required by SORT

Please specify Sort/Merge keys:

NUMBER OF KEYS     = 1   (Less than or equal to 8)
Key attributes:
          Position (>0)      Length in Bytes   Sort Key Type    Sort Order (A,D)
Key 1:    POST1     = ■■■■   LENGTH1 = ■■■   TYPE1  = C       ORDER1  = A
Key 2:    POST2     = ■■■■   LENGTH2 = ■■■   TYPE1  = C       ORDER1  = A
Key 3:    POST3     = ■■■■   LENGTH3 = ■■■   TYPE1  = C       ORDER1  = A
Key 4:    POST4     = ■■■■   LENGTH4 = ■■■   TYPE1  = C       ORDER1  = A
Key 5:    POST5     = ■■■■   LENGTH5 = ■■■   TYPE1  = C       ORDER1  = A
Key 6:    POST6     = ■■■■   LENGTH6 = ■■■   TYPE1  = C       ORDER1  = A
Key 7:    POST7     = ■■■■   LENGTH7 = ■■■   TYPE1  = C       ORDER1  = A
Key 8:    POST8     = ■■■■   LENGTH8 = ■■■   TYPE1  = C       ORDER1  = A

* Key Type:    B=Binary, C=Character, D=Decimal, F=Floating, P=Packed
               Z=Zoned decimal, L=Zoned decimal, sign leading
* Sort Order:  A=Ascending, D=Descending
```

Figure 14-8. SORT Sort/Merge Keys Screen

From the Sort/Merge Keys screen, you define the data fields by which the file is to be sorted or merged. The fields you define are the sort keys (which should not be confused with the index keys).

The fields in the following list represent the components of a single sort key definition. Each set of fields shown on the screen shares a numeric suffix, represented in the list by *x*. You can define as many as eight keys. To define a key, specify the fields as follows:

NUMBER OF KEYS -- Specify the number of keys you want to use for this Sort or Merge operation. Up to eight keys are allowed.

POSTx -- Specify the position of first byte of the key field within the record. (The first byte of the record occupies Position 1.) This field has no default value.

LENGTHx -- Specify the length of the key field in bytes. There is no default value. Certain restrictions, which vary according to data type, apply to the length of the key field. See Table 14-2 for a list.

TYPEx -- Specify the data type of the key field, using a single-character code. Table 14-2 shows data types and codes. The default value for this field is C (character).

ORDERx -- Specify the order in which the output file should be arranged: A (Ascending order -- low to high) or D (Descending order -- high to low). The default value for ORDERx is A. Data is sorted in ASCII order unless you have modified the collating sequence (see Section 14.2.2).

If you define more than one key from the Sort/Merge Keys screen, SORT uses the keys in the order in which they are defined; that is, the sort is done first on Key 1, then on Key 2, then on Key 3, and so on. When you finish defining Sort/Merge keys, press ENTER to begin the operation.

Table 14-2. Data Types and Length Restrictions for SORT Key Fields

Code	Type	Key Field Length Restriction
B	Binary	2, 4, or 8 bytes exactly
U	Unsigned binary	2, 4, or 8 bytes exactly
C	Character data (alphanumeric)	256 bytes maximum
D	External decimal signed or unsigned; sign trailing or leading in separate byte	19 bytes maximum
F	Floating point	8 bytes exactly
Z	Zoned decimal, sign trailing included in last byte	18 bytes maximum
L	Zoned decimal, sign leading included in first byte (overpunched)	18 bytes maximum
P	Packed decimal, sign trailing included in last byte	16 bytes maximum

14.5 SPECIFYING THE OUTPUT RECORD FORMAT

If you entered YES in the Reformat field of the Options screen (Figure 14-1) to indicate that you wanted the fields in the output record arranged in a different order from the fields in the input record, the Record Format screen (Figure 14-9) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  FORMAT
                                                          Message Id:  0000
                                                           Component:  SORT

                          Information Required by SORT

Please enter the output record format:

Output record  LENGTH   = ■■80   PAD      = ■■ (2 Hex digits or 1 character)

           Input position          Length          Output Position
Field 1:   INPOS1  = ■■■■   LENGTH1  = ■■■■   OUTPOS1  = ■■■■
Field 2:   INPOS2  = ■■■■   LENGTH2  = ■■■■   OUTPOS2  = ■■■■
Field 3:   INPOS3  = ■■■■   LENGTH3  = ■■■■   OUTPOS3  = ■■■■
Field 4:   INPOS4  = ■■■■   LENGTH4  = ■■■■   OUTPOS4  = ■■■■
Field 5:   INPOS5  = ■■■■   LENGTH5  = ■■■■   OUTPOS5  = ■■■■
Field 6:   INPOS6  = ■■■■   LENGTH6  = ■■■■   OUTPOS6  = ■■■■
Field 7:   INPOS7  = ■■■■   LENGTH7  = ■■■■   OUTPOS7  = ■■■■
Field 8:   INPOS8  = ■■■■   LENGTH8  = ■■■■   OUTPOS8  = ■■■■
Field 9:   INPOS9  = ■■■■   LENGTH9  = ■■■■   OUTPOS9  = ■■■■
Field 10:  INPOS10 = ■■■■   LENGTH10 = ■■■■   OUTPOS10 = ■■■■
```

Figure 14-9. SORT Record Format Screen

Use this screen to specify a new order for the fields in the output record; at your option, you can also change the record length and specify a pad character to fill out a lengthened record. The current position, length, and new position of each field are entered in a set of three fields that share a numeric suffix (represented by **x** in the following list).

Enter information in the fields as follows:

Output Record LENGTH -- The length of the output record. The input record length appears as a default. If you choose, you can enter a different length -- shorter if you want to truncate the record, longer if you want to extend it in order to add more fields at a later time. An extended record is padded with blanks unless you specify a different pad character in the Pad field.

PAD -- If you specify an output record longer than the input record and leave this field unspecified, SORT fills every byte in the record that is not part of a currently defined field with a blank (X'20'). However, you can replace this default pad character with a different character by entering two hexadecimal digits or a nonblank character in this field.

INPOSx -- Enter the byte position in the input record of the field you want to move, i.e., the position of the first byte of the field (counting the first byte in the record as 1).

LENGTHx -- Enter the length in bytes of the field you want to move. You can truncate or extend the field by entering a length different from its length in the input file.

OUTPOSx -- Enter the byte position of the field in the output record, i.e., the position you want the first byte of the field to occupy (counting the first byte in the record as 1).

Use as many sets of fields on the Record Format screen as you need to rearrange the field positions within the output record to your satisfaction. If you fill in all the fields on the screen, SORT displays another Record Format screen when you press ENTER. Pressing ENTER with one or more blank sets of fields on the screen indicates that your specifications for the output record format are complete, and SORT goes on to solicit specifications for the output file.

Note: When you specify new positions for the fields in the output record, be careful that fields are not allowed to overlap. Since SORT does not check for this condition, the data in an overlapped field will be lost when the record is reformatted.

14.6 SPECIFYING THE OUTPUT FILE

After SORT processes the input files, the Output File Specification screen (Figure 14-10) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  OUTPUT
                                                          Message Id:  0000
                                                           Component:  SORT

                          Information Required by SORT

Please specify the output file below:

OUTPUT FILE      = ■■■■■■■■ in LIBRARY  = ■■■■■■■■ on VOLUME  = ■■■■■■

Replace input file with same name?              REPLACE = NO■     (YES or NO)
Fileorg - Consecutive, Relative or Indexed?     FILEORG = C       (C, R or I)
If Indexed, specify packing factors (1-100):    IPACK   = 100
                                                DPACK   = 100
(Maximum) Record Size?                          RECSIZE = 00080
Rectype - Fixed or Variable Length Records?     RECTYPE = V       (F or V)
Do you want the file compressed?                COMPRESS = YES    (YES or NO)

FILE OUTPUT DEVICE  = DISK      (DISK or TAPE)

If output device is tape, then enter file sequence number.   FILESEQ = 1
```

Figure 14-10. SORT Utility Output File Specification Screen

Use this screen to specify the file in which you would like the sorted output to be placed. Enter information in the fields as follows:

OUTPUT FILE -- The name of the output file to be created. If you specify the same name, library, and volume for the output file as for one of the input files, SORT displays a message warning you that this action causes the sorted output file to replace the input file. If you are sure you want the input file replaced, press PF3 to delete the original input file. The sorted output file then takes its place.

LIBRARY -- The name of the library in which you want to place the output file. The default value is taken from the OUTLIB field of your usage constants.

VOLUME -- The name of the volume on which to place the output file. The default value is taken from the OUTVOL field of your usage constants.

REPLACE -- Specify YES or NO to indicate whether you want SORT to replace the input file with the sorted output file without prompting you to confirm the replacement.

If you enter YES in the Replace field, you must specify output file, library, and volume names that are identical with those of the input file. If you enter NO in this field, you must specify an output file, library, or volume name that differs from that of the input file. The default value for Replace is NO.

The Replace option is not available, and the field does not appear on the screen, under any of the following conditions:

- You specified a Merge operation.
- You are sorting more than one input file.
- The input file is on tape.
- The input file is opened in Shared mode.
- You are creating an Addrout or Keyout file.

FILEORG -- Specify C (Consecutive), I (Indexed), or R (Relative) file organization for the output file. The default value is C. SORT cannot create an indexed file if you have specified reformatting of records (see Section 14.5).

Note: *If you specify an indexed file, SORT automatically defines the primary index key. See Section 14.6.1 for important information about this process.*

IPACK -- Specify the packing density of the index blocks (stored on the output disk). The default value is 100.

DPACK -- Specify the packing density of the data blocks stored on the output disk. The default value is 100.

Note: *You must specify packing factors if you specify I (Indexed) in the Fileorg field. See the* VS Data Management System (DMS) Reference *for further information on packing densities.*

RECSIZE -- Enter the record size in bytes of the largest data record in the file. Maximum sizes for data file records range from 2024 to 2048 bytes, depending on the type of file. For fixed length records, RECSIZE is the actual record length; for variable length and compressed files, set RECSIZE to the maximum uncompressed record length for the file. For relative files, RECSIZE represents the logical record size, not the record slot size. See the *VS Data Management System (DMS) Reference* for further information on record size.

RECTYPE -- Specify Fixed (F) or Variable (V) record length. You can use this option to take a variable-length input file and make it a fixed-length output file, provided you do not use the Compress option (that is, COMPRESS should be set to NO).

COMPRESS -- Specify YES or NO to indicate whether you want SORT to compress data on the output file. Compression can save disk space. The default value is YES for variable-length input and NO for fixed-length input. If the input file is variable-length and the output file is to be fixed-length, however, COMPRESS should be NO.

FILE OUTPUT DEVICE -- Specify the type of device on which the output file is to reside. If the output device is DISK, no further information is needed to complete the output specifications. If the output device is TAPE, however, you must specify the file sequence number. The default value for this field is DISK.

FILESEQ -- Specify the sequence number of the output file on the tape, if any. The default value is 1.

When you finish entering specifications, press ENTER. If you entered I (for an Indexed output file) in the Fileorg field, you now have the opportunity to specify alternate index keys, as described in Section 14.6.2. Otherwise, your specifications are complete, and SORT goes on to create the output file.

14.6.1 Primary Keys for Indexed Files

When it creates an indexed output file, SORT constructs a primary key on the basis of the sort keys you have specified. It does this by concatenating the first sort key specified with the sort keys that follow it in the output record. This method has implications of which you should be aware.

Sort keys can be concatenated only if they are contiguous in the record. For example, if the sort keys LNAME, FNAME, and ADDR occur in succession in the output record, SORT combines all three into one primary key. If, however, a field that is not a sort key lies between FNAME and ADDR, only LNAME and FNAME become part of the primary key, increasing the probability of duplicate keys (a large file may well contain more than one SMITH JOHN, for instance). If primary keys are duplicated, it is impossible to create a valid indexed file, and SORT is forced to abort the operation.

The way primary keys are constructed may also lead to a problem of a different kind. Since SORT concatenates *all* contiguous sort keys, it may include numeric fields of various data types as part of the primary key. DMS sorts all keys as if they contained character data, and the inclusion of numeric fields, especially if the numbers they contain are negative, may cause the sort order to be distorted.

Since the two options cannot be combined, you cannot have SORT reformat the records to eliminate primary key problems while creating an indexed output file. If an indexed output file is essential, however, you can use a two-stage process:

1. Run SORT on the input file, specifying a consecutive output file with reformatted records. Rearrange the order of the fields so that the contiguous sort keys will form a unique primary key free of negative numeric data.

2. Run SORT again, using as input (in place of the original input file) the reformatted file that was the output of the first run of the program. Do not attempt to reformat the records. Specify an indexed output file.

If the sort key fields are so arranged that a valid primary key cannot be created by concatenating adjacent sort keys, and if the procedure described above is impracticable, it is advisable to specify a consecutive rather than an indexed output file.

14.6.2 Specifying Alternate Index Keys

If you specified an indexed output file by entering I in the Fileorg field of the Output File Specification screen (Figure 14-10), the Alternate Key Specification screen (Figure 14-11) appears as soon as you press ENTER from that screen.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  FILEKEYS
                                                          Message Id:  000
                                                           Component:  SORT

                          Information Required by SORT

Please specify the alternate keys to be created, if any:

    Key 01:   KEYPOS01 = ____   KEYSIZ01 = ___   DUPOK01 = YES
    Key 02:   KEYPOS02 = ____   KEYSIZ02 = ___   DUPOK02 = YES
    Key 03:   KEYPOS03 = ____   KEYSIZ03 = ___   DUPOK03 = YES
    Key 04:   KEYPOS04 = ____   KEYSIZ04 = ___   DUPOK04 = YES
    Key 05:   KEYPOS05 = ____   KEYSIZ05 = ___   DUPOK05 = YES
    Key 06:   KEYPOS06 = ____   KEYSIZ06 = ___   DUPOK06 = YES
    Key 07:   KEYPOS07 = ____   KEYSIZ07 = ___   DUPOK07 = YES
    Key 08:   KEYPOS08 = ____   KEYSIZ08 = ___   DUPOK08 = YES
    Key 09:   KEYPOS09 = ____   KEYSIZ09 = ___   DUPOK09 = YES
    Key 10:   KEYPOS10 = ____   KEYSIZ10 = ___   DUPOK10 = YES
    Key 11:   KEYPOS11 = ____   KEYSIZ11 = ___   DUPOK11 = YES
    Key 12:   KEYPOS12 = ____   KEYSIZ12 = ___   DUPOK12 = YES
    Key 13:   KEYPOS13 = ____   KEYSIZ13 = ___   DUPOK13 = YES
    Key 14:   KEYPOS14 = ____   KEYSIZ14 = ___   DUPOK14 = YES
    Key 15:   KEYPOS15 = ____   KEYSIZ15 = ___   DUPOK15 = YES
    Key 16:   KEYPOS16 = ____   KEYSIZ16 = ___   DUPOK16 = YES
```

Figure 14-11. SORT Alternate Key Specification Screen

You can make your indexed output file an alternate indexed file simply by specifying the alternate index keys from this screen. Enter the specifications for each alternate key in a set of three fields which share the same numeric suffix (represented by *xx* in the list below). You can use as many of these sets of fields as you want, up to a maximum of 16. If you specify no alternate keys, SORT creates a simple indexed file.

To define alternate keys, enter information in the fields as follows:

KEYPOSxx -- Specify the starting byte position of the alternate key field. A valid key position can be any byte from byte 1 to the last byte of the record.

KEYSIZxx -- Specify the length (in bytes) of the alternate key field. The key size can be any value from 1 to the length of the record, but the sum of the primary key plus any alternate key cannot exceed 255 bytes. The default value is 6.

DUPOKxx -- Specify YES or NO to indicate whether you want to allow duplicate values for this alternate key. The default is YES.

When you press ENTER from this screen, whether or not you have specified any alternate keys, SORT creates the output file and ends processing. You are returned to the Command Processor or to the procedure from which SORT was called, with a message indicating the completion of SORT processing.

If SORT was unable to complete the operation correctly, an error message and a return code appear. (See Appendix B for an explanation of return codes.)

14.7 RESTARTING THE SORT UTILITY

SORT sometimes halts a Sort or Merge operation because it discovers that the input file contains more records than the program is prepared for. This problem comes about in the following way.

Before it begins processing an input file, SORT creates a temporary work file on disk to hold the records it is processing. The size of this work file is determined by the number of records SORT expects to find in the input file.

- For input files housed on disk, SORT gets this information from the disk Volume Table of Contents (VTOC).
- For input files housed on tape, SORT depends on the user's estimate, as specified in the RECORDS field of the Options screen (Figure 14-1).

Once processing begins, if SORT encounters more records in the input file than it is prepared for, it does the following:

1. Halts the sorting or merging operation.
2. Reads through the input file to determine the number of records it contains.

3. Displays an error message. The error message screen reports the actual number of records in the file and offers you the following choices:

   - Pressing PF1 to restart the operation with a new work file based on the actual record count. It is unnecessary to reenter your specifications, since SORT has saved them.

   - Pressing PF16 to exit from SORT without creating an output file.

When the input file is on tape, the disparity arises from the user's underestimate and implies no defect in the file. *When the input file is on disk, however, you should be aware that a disparity between the VTOC information and the number of records in the file indicates that the input file may be damaged.*

SORT can still complete its operation successfully without harm to the input file, so there is no reason why you should not press PF1. Once SORT processing is finished, however, you should use the COPY utility to reorganize the input file if you intend to use it further. See the *VS Media, Transfer, and Device Utilities Reference* for information on file reorganization with the COPY utility.

14.8 A SAMPLE SORT PROCEDURE

SORT processing can be controlled through the VS Procedure Language. In a procedure, you can specify all SORT input and output options, selection criteria, and sort keys. Appendix A contains a complete list of SORT GETPARMs listing parameter reference names (prnames) and all fields. See the *VS Procedure Language Reference* for details about VS Procedure Language syntax.

In the sample SORT procedure that follows, all defaults of the Options screen (Figure 14-1) have been accepted. The output file is sorted in descending order. The record selection criteria specify that the value of the first byte of the output records must be greater than or equal to the value at Byte 10 of the input records, or greater than a literal value of A. Only records that meet one of these criteria are included in the sort.

Note: *The value in VALUE2 is enclosed in single quotation marks within double quotation marks. 'A' is a literal value specified from the Select screen. Because VS Procedure Language uses quotation marks as delimiters, both sets of quotation marks are necessary. If you eliminate one set of quotation marks when writing the SORT procedure, or if both sets of quotation marks are of the same type (i.e., either single or double), the procedure returns an error message indicating incorrect punctuation, with a return code of 4. (Appendix B explains return codes.)*

```
PROCEDURE
  RUN SORT
    ENTER OPTIONS
    ENTER INPUT FILE=SAVE3, LIBRARY=DJDCOPY, VOLUME=NEWSYS,
      SELECT=YES
    ENTER SELECT FLDPOS1=1, LENGTH1=1, TSTREL1=GE, VALUE1=10,
      CONECT1=OR, FLDPOS2=1, LENGTH2=1, TSTREL2=GT, VALUE2="'A'"
    ENTER KEYS POST1=1, LENGTH1=1, ORDER1=D
    ENTER OUTPUT FILE=FILE1, LIBRARY=PROC1
  RETURN
```

CHAPTER 15
SORTINT

15.1 INTRODUCTION

The SORTINT utility is an international version of the SORT utility, which is described in Chapter 14. It can sort up to 10 files into a single, ordered output file. SORTINT can accommodate international character sets as well as standard American Standard Code for Information Interchange (ASCII) characters, and it can sort according to the standard ASCII sequence or according to an externally defined collating sequence. (You can use the TABLEDIT utility, described in Chapter 16, to define an external collating sequence.)

Like SORT, SORTINT can be used for reformatting the output record. It supports all the standard features of SORT except tape input and output and the use of shared consecutive files. Although it does not support its own Merge function, SORTINT makes this function available by linking to SORT. With its ability to accommodate character sets and sort orders other than standard ASCII, the SORTINT utility can do any of the following jobs:

- Sort a single data file in an order you specify.
- Sort up to 10 data files into a single, ordered output file.
- Merge up to 20 data files into a single, ordered output file.
- Select specific records from one or more input files and sort them into a single, ordered output file.
- Produce an output file that contains only the primary index key field from each record in the input file. You can specify the sort order for this key field.
- Produce an ordered output file of 3-byte records to be used by RPG II programs.

- Reformat the record layout of the output file, modifying the length and sequence of fields, extending and padding the record, or truncating it so as to include only selected fields.

SORTINT can retrieve input files from disk volumes, but not from tape volumes. Input files are left intact unless you assign the same name to the input and output files -- doing this causes the input file to replace the output file.

Note: *An output file that has been sorted according to an external collating sequence cannot be used for input to other processes, because the VS Data Management System (DMS) does not support external collating sequences.*

15.1.1 Storage Requirements

To accommodate the work files produced in the course of sorting operations, SORTINT requires a 200 percent overhead in disk space.

15.1.2 Overview

To do SORT processing, you specify the following items in the order listed:

1. The program function (Sort or Merge)
2. The location of the external collating sequence, if any
3. The input file or files to be sorted or merged
4. The keys on which to sort or merge the files
5. Optionally, a new format for the output record
6. The output file that will receive the sorted data

15.1.3 Running SORTINT

To run the SORTINT utility, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify SORTINT in the Program field and press ENTER.

The following sections describe SORTINT processing:

Section	Process
15.2	Specifying SORTINT Options
15.2.1	Specifying an External Collating Sequence
15.3	Specifying Input Files To Be Sorted or Merged
15.3.1	Defining Record Selection Criteria
15.4	Defining the Sort or Merge Keys
15.5	Specifying the Output Record Format
15.6	Specifying the Output File
15.7	Restarting the SORTINT Utility
15.8	A Sample SORTINT Procedure

15.2 SPECIFYING SORTINT OPTIONS

When SORTINT processing begins, the Options screen (Figure 15-1) appears.

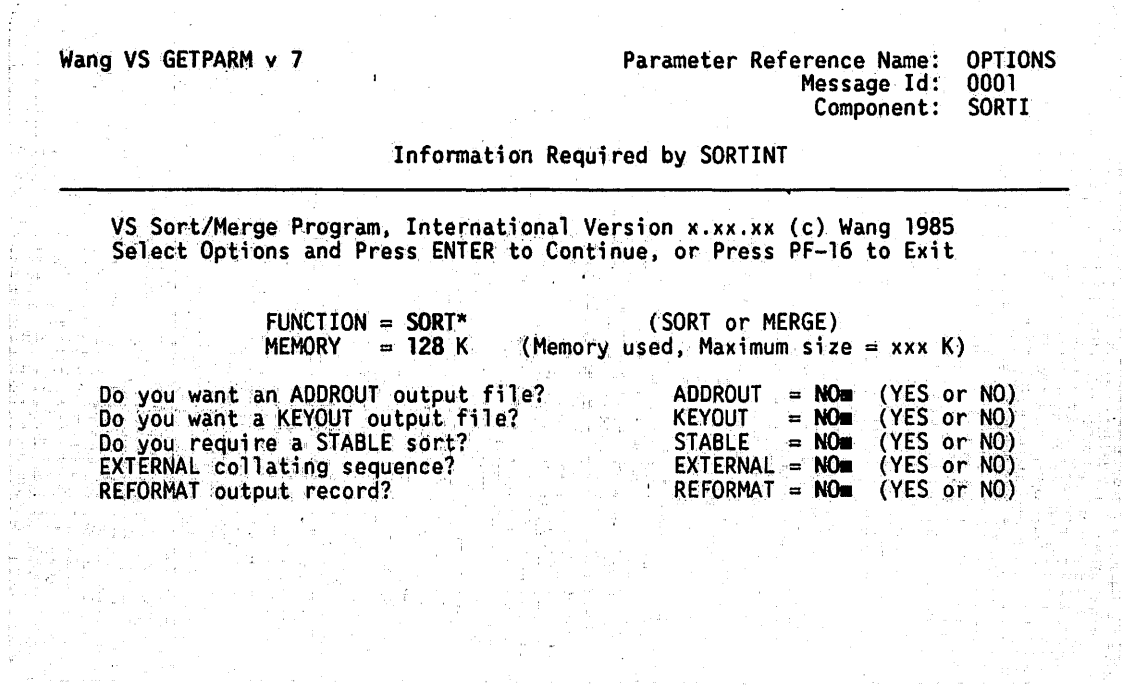


Figure 15-1. SORTINT Options Screen

Use this screen to specify the Sort or Merge function, the amount of memory to be used for the Sort or Merge operation, and the options that determine the nature of the output file. To terminate SORTINT processing without specifying program options, press PF16.

Enter information in the fields of the Options screen as follows:

FUNCTION -- Specify the function (SORT or MERGE) that you want SORTINT to perform. The Sort function sorts one or more files into a single ordered output file. The Merge function, which is called from the SORT utility, merges two or more ordered files into a single sorted output file. The default value is SORT.

MEMORY -- Specify the amount of main memory you want allocated for processing the Sort or Merge operation. The greater the amount of memory used, the more efficient the operation. For most operations, 128 KB (the default value) is sufficient. If that much main memory is unavailable, the system determines the amount of memory actually available and uses it in running the program.

You can enlarge the work area beyond 128 KB to improve Sort or Merge processing time, but this increases competition with other users for main memory and may degrade system performance. The maximum amount of main memory you can allocate (which depends on the size of physical memory in the system) is displayed next to the Memory field on the Options screen.

The optimum amount of main memory for each Sort or Merge operation can only be determined by experimenting with the actual operating environment. If the Sort or Merge operation requires a large amount of processing time or main memory, it is best to run it in batch mode during the off-peak hours of system use so that other users are not affected.

ADDROUT -- If you specify YES in the Addrout field, SORTINT produces a single-field consecutive file of sorted, three-byte records that can be used by RPG II programs. Each record is a positive binary number representing the relative number of a record in the input file. When you do a sort with the Addrout option, these numbers are the *only* output. For an example of the Addrout option, see Section 14.3.1. For more information about Addrout files, see the *VS RPG II Language Reference*.

You cannot use the Addrout field with the Merge function, nor can you specify YES in both the Addrout and Keyout fields. Addrout can be combined with a Stable sort. The default value for Addrout is NO.

KEYOUT -- Specify YES in the Keyout field to produce a single-field consecutive file consisting of the primary keys of the records in the order specified by the sort criteria. The input file must be indexed.

You cannot use the Keyout field with the Merge function, nor can you specify YES in both the Addrout and Keyout fields. Keyout can be combined with a Stable sort. The default value for Keyout is NO. For an example of the Keyout option, see Section 14.2.1.

STABLE -- Specify YES in the Stable field to produce an output file in which records with identical sort keys remain in the same order as the input order sequence. The default value for Stable is NO. For an example of the Stable option, see Section 15.2.1.

EXTERNAL -- Specify YES or NO to indicate whether you want to use a collating sequence other than ASCII. (This option is unavailable with the Merge function.) If you enter YES, SORTINT later displays a screen from which you can specify the file that contains the external collating sequence definition. The default value for this field is NO.

REFORMAT -- Specify YES or NO to indicate whether you want SORTINT to reformat the record layout of the output file -- i.e., to arrange the fields in a different order than that of the input file. If you specify reformatting, SORTINT later displays a screen from which you can change field or record lengths and specify a new order for the fields (see Section 15.5). The default value is NO.

You can use the Reformat option in conjunction with the Stable option, but not with the Addrout or Keyout options. The Reformat option is not available with the Merge function, and it applies to consecutive output files only; do not select it if you intend to specify an indexed output file.

Note: *If you select neither an external collating sequence nor reformatting of records (i.e., if the values of both the External and Reformat fields are NO), SORTINT calls the SORT utility to do the sort operation, since none of SORTINT's special capabilities are required. See Chapter 14 for a description of SORT processing.*

Table 15-1 shows which combinations of program functions and output options are possible:

Table 15-1. SORTINT Utility Function/Option Matrix Table

Option	Program Function		Output Option				
	SORT	MERGE	ADDROUT	KEYOUT	STABLE	REFORMAT	EXTERNAL
ADDROUT	YES	NO	-	NO	YES	NO	YES
KEYOUT	YES	NO	NO	-	YES	NO	YES
STABLE	YES	YES	YES	YES	-	YES	YES
REFORMAT	YES	NO	NO	NO	YES	-	YES
EXTERNAL	YES	NO	YES	YES	YES	YES	-

Enter or change information in the fields of the Options screen as appropriate, and press ENTER to continue SORTINT processing. For examples of the output generated through the Addrout, Keyout, and Stable options, see Section 14.2.1 in the chapter on SORT.

15.2.1 Specifying an External Collating Sequence

If you selected an external collating sequence by entering YES in the External field of the Options screen, SORTINT displays a screen (not shown) that prompts you to specify the name, library, and volume of the file containing the definition of the collating sequence you want SORTINT to use. As a default value in the Library field, SORTINT supplies the name TRNTABLE; the default value in the volume field is taken from the INVOL field of your usage constants.

Defining a special collating sequence is not part of SORT processing. To create a file that defines an external collating sequence, use the TABLEDIT utility, described in Chapter 16.

15.3 SPECIFYING INPUT FILES TO BE SORTED OR MERGED

After you finish specifying the program options, the first Input File Specification screen (Figure 15-2) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  INPUT
                                                          Message Id:  0001
                                                           Component:  SORTI

                        Information Required by SORTINT
-------------------------------------------------------------------------------

Please enter name of the file to be sorted.

  INPUT FILE   = ________  in LIBRARY  = ________  on VOLUME  = ______

  Is this a SHARED file?                    SHARED   = NO_  (YES or NO)
  Do you want to select input records?      SELECT   = NO_  (YES or NO)
  Do you want more input files?             MOREFILE = NO_  (YES or NO)
```

Figure 15-2. SORTINT Input File Specification Screen

Use this screen to specify the file, library, and volume names of an input file you want to process. Other options enable you to specify whether you want to select records from the file instead of sorting or merging all the records it contains, and whether you want to process more than one input file in a Sort operation.

Enter information in the fields of the Input File Specification screen as follows:

INPUT FILE -- The name of the first or next input file you want to sort or merge. You can specify up to 10 input files for the Sort function or 20 for the Merge function. The Input File Specification screen automatically reappears until you press ENTER from an unaltered screen. *(Note that, for the Sort function, you must specify YES in the Morefile field of the first Input File Specification screen if you want to process more than one input file; see the description of that field.)*

LIBRARY -- The name of the library in which the input file resides. The default value is taken from the INLIB field of your usage constants.

VOLUME -- The name of the volume on which the input file resides. The default value is taken from the INVOL field of your usage constants.

SHARED -- Specify YES or NO to indicate whether you want to open consecutive and indexed input files in Shared mode. If you enter YES, the Lock screen appears next, and you must specify the fields on that screen. (This screen is displayed by the SORT utility. For information about the Lock screen and its fields, see Section 14.3.1.) If you enter NO, the input file is opened in Input mode. The default value is NO.

SELECT -- Specify YES or NO to indicate whether you want to select specific input records for use in a Sort or Merge operation. If you enter YES, the Select screen appears to prompt you for the selection criteria. (For more information on defining selection criteria, see Section 15.3.1.) If you enter YES, only the records that match the selection criteria you specify are used to produce the output file. If you enter NO, the Select screen does not appear, and all the records in the input file are processed. The default value is NO.

MOREFILE -- Specify YES or NO to indicate whether you want to sort more than one input file into a single output file. (The sorted files are also merged.) If you enter YES, you must specify all the input files before SORT processing can continue. Additional Input File Specification screens (up to a maximum of 20) automatically appear for this purpose. (The additional screens do not include the Select and Morefile options, since these options are specified only once, from the initial Input File Specification screen.)

Note: *It is not necessary to specify the Morefile field for the Merge function, which implies multiple input files. If you specified a Merge operation from the Options screen, additional Input File Specification screens will appear automatically (up to a maximum of 20), regardless of the value in this field.*
If you specified that the input file should be opened in Shared mode, the Lock screen appears as soon as you press ENTER from the Input File Specification screen. See Section 14.3.1 for information on specifying the fields of this screen.

If you did not specify Shared mode, a new Input File Specification screen appears.

When you finish entering the names of files to be sorted or merged, do not enter anything when the next Input File Specification screen appears; simply press ENTER from the blank screen to terminate the file specification process.

15.3.1 Defining Record Selection Criteria

If you specify YES in the Select field of the Input File Specification screen (Figure 15-2), the SORTINT utility selects the records to be sorted or merged on the basis of the relationships you define from the Record Selection screen, which appears when you press ENTER from the Input File Specification screen. Figure 15-3 shows an example of a SORTINT Record Selection screen. (If you have also specified Shared mode for the same input file, the Lock screen, described in the previous section, appears before the Record Selection screen.)

```
Wang VS GETPARM v 7                          Parameter Reference Name:  SELECT
                                                          Message Id:  SEL
                                                           Component:  SORT

                        Information Required by SORTINT

Enter record selection criteria below, supplying field position, length, and
  format type (Binary, Char, Decimal, ...), test relation (EQ,NE,GT,GE,LT,LE),
  and test value (in quotes) or comparison field position (without quotes).
Set connector to "AND" to continue current criterion; use "OR" to begin
  alternative criterion.      (Use Chars or Decimal numeric for test value.)

   FLDPOS1  = 0001     LENGTH1  = 001     FLDTYPE1 = C
   TSTREL1  = EQ       VALUE1   = "P"                    CONECT1 = AND

   FLDPOS2  = 0003     LENGTH2  = 001     FLDTYPE2 = C
   TSTREL2  = EQ       VALUE2   = "A"                    CONECT2 =

   FLDPOS3  =          LENGTH3  =         FLDTYPE3 = C
   TSTREL3  =          VALUE3   =                        CONECT3 =

   FLDPOS4  =          LENGTH4  =         FLDTYPE4 = C
   TSTREL4  =          VALUE4   =                        CONECT4 =
```

Figure 15-3. Sample SORTINT Record Selection Screen

Individual input records are selected by testing them against values you specify for comparison from the Record Selection screen. For example, suppose you are processing a payroll file and only want to sort the records of part-time employees. In this file, the records of part-time employees all have the letter *P* in the first byte of each record. Therefore the test criterion is that all input records to be processed must have the value *P* (X'50') in Position 1. If the record matches the test criterion, it is selected for sorting. If not, it is neither sorted nor merged, and is not placed in the output file.

Besides being tested against a literal value, one field can also be tested against another field in the same record. For example, you can specify that if the four bytes beginning at Position 12 exceed in value the four bytes beginning at Position 43, the record can be processed. You can also combine multiple criteria by using the optional CONECTx field (where x is a number) to connect successive criteria with a logical AND or OR.

In the sample Record Selection screen illustrated in Figure 15-3, the user wants to sort only those payroll records that indicate *active* part-time employees. The test criteria therefore specify that all input files must have *P* in Position 1 AND *A* in Position 3 (indicating Active status).

The SORTINT utility processes test statements in the following ways:

- If Statement 1 is true *OR* Statement 2 is true (either can be true), the record is selected.

- If Statement 1 is true *AND* Statement 2 is true (both must be true), the record is selected.

For an illustration of record selection, which is done identically for SORT and SORTINT, see Figures 14-6 and 14-7 in Section 14.3.2.

The fields in the following list represent the components of a single test statement or criterion, followed by an optional connector to the next criterion. Each set of fields on the screen shares a numeric suffix, represented by an *x* in the list (FLDPOSx, etc.) You can specify as many as 32 selection criteria for a single Sort or Merge operation. Four criteria can be defined from each Record Selection screen. If the last CONECTx field on the screen is specified, SORT displays another screen, until the maximum is reached. An unspecified CONECTx field indicates that all record selection criteria have been defined.

Enter information in the fields as follows:

FLDPOSx -- Specify the position in the record of the first byte of the field you intend to test. (The first byte of the record occupies Position 1.) When you are specifying multiple criteria, you do not have to specify field positions in ascending numeric order. For example FLDPOS1 = 5, FLDPOS2 = 3 is just as valid as FLDPOS1 = 3, FLDPOS2 = 5. There is no default value for this field.

LENGTHx -- Specify the number of bytes in the field you want tested, beginning with the byte you specified for the field position (FLDPOSx). For example, if FLDPOS1 = 12 and LENGTH1 = 4, the field tested consists of bytes 12, 13, 14, and 15. Depending on the way the criterion is defined, there are limits on the number of bytes that can be tested; this condition imposes an effective limit on the valid length of the test field. See the description of the VALUEx field for further information. LENGTHx has no default value.

FLDTYPx -- Specify the data type of the field being tested and the criterion specified in the VALUEx field. The default value for FLDTYPx is C. The field type options are

C Character data (alphanumeric)
B Binary
D External decimal signed or unsigned; sign trailing or leading in separate byte
L Zoned decimal, sign leading included in first byte (overpunched)
P Packed decimal, sign trailing included in last byte
Z Zoned decimal, sign trailing included in last byte

TSTRELx -- Specify the type of comparison to be made between the input data (starting at FLDPOSx) and VALUEx. There is no default value for TSTRELx. The test relationship options are

EQ Equal
NE Not equal
GT Greater than
LT Less than
GE Greater than or equal to
LE Less than or equal to

VALUEx -- Specify the value to be compared with the field in each input record beginning at FLDPOSx and extending for LENGTHx bytes. VALUEx can be a literal data value enclosed in single or double quotation marks, or it can be the starting position of another field in the record.

Note: *You must enclose literal data values in quotation marks, even if they are numbers; otherwise, VALUEx is interpreted as a field position. If you intentionally specify the starting position of another field in the record, this field must be of the same type and length as the original field. Specifying a field whose type and length do not match LENGTHx and FLDTYPx or omitting quotation marks from a literal data value may result in an erroneous output file or an aborted operation.*

You can specify a maximum of 16 bytes for a literal data value. For field-to-field comparisons, the maximum depends on the data type:

C	256
B	2 or 4
D	16
L	16
P	16
Z	16

There is no default value for this field.

CONNECTx -- If you intend to define another criterion, specify the logical connector (AND or OR) you want to use to combine it with the one you have just defined. Since you can specify up to 32 comparisons for each Sort or Merge operation, the maximum number of ANDs and ORs that you can use for any one operation is 31. CONECTx is an optional field whose default value is blank. The first blank CONECTx field signifies that all selection criteria have been defined.

15.4 DEFINING THE SORT OR MERGE KEYS

After you have specified the input file and the record selection criteria, if any, the Sort/Merge Keys screen (Figure 15-4) appears.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  KEYS
                                                          Message Id:  0001
                                                           Component:  SORT

                        Information Required by SORTINT

Please specify Sort/Merge keys:

NUMBER OF KEYS     = 1   (Less than or equal to 8)
Key attributes:
          Position (>0)      Length in Bytes  Sort Key Type    Sort Order (A,D)
Key 1:    POST1     = ____   LENGTH1  = ___   TYPE1   = C     ORDER1   = A
Key 2:    POST2     = ____   LENGTH2  = ___   TYPE1   = C     ORDER1   = A
Key 3:    POST3     = ____   LENGTH3  = ___   TYPE1   = C     ORDER1   = A
Key 4:    POST4     = ____   LENGTH4  = ___   TYPE1   = C     ORDER1   = A
Key 5:    POST5     = ____   LENGTH5  = ___   TYPE1   = C     ORDER1   = A
Key 6:    POST6     = ____   LENGTH6  = ___   TYPE1   = C     ORDER1   = A
Key 7:    POST7     = ____   LENGTH7  = ___   TYPE1   = C     ORDER1   = A
Key 8:    POST8     = ____   LENGTH8  = ___   TYPE1   = C     ORDER1   = A

* Key Type:    B=Binary, C=Character, D=Decimal, F=Floating, P=Packed
               Z=Zoned decimal, L=Zoned decimal, sign leading
* Sort Order:  A=Ascending, D=Descending
```

Figure 15-4. SORT Sort/Merge Keys Screen

From the Sort/Merge Keys screen, you define the data fields by which the file is to be sorted or merged. The fields you define are the sort keys (which should not be confused with the index keys).

The fields in the following list represent the components of a single sort key definition. Each set of fields shown on the screen shares a numeric suffix, represented in the list by *x*. You can define as many as eight keys. To define a key, specify the fields as follows:

NUMBER OF KEYS -- Specify the number of keys you want to use for this Sort or Merge operation. Up to eight keys are allowed.

POSTx -- Specify the position of the first byte of the key field within the record. (The first byte of the record occupies Position 1.) This field has no default value.

LENGTHx -- Specify the length of the key field in bytes. There is no default value. Certain restrictions, which vary according to data type, apply to the length of the key field. See Table 14-2 for a list.

TYPEx -- Specify the data type of the key field, using a single-character code. Table 15-2 shows data types and codes. The default value for this field is C (Character).

ORDERx -- Specify the order in which the output file should be arranged: A (Ascending order -- low to high) or D (Descending order -- high to low). The default value for ORDERx is A. Data is sorted in ASCII order or according to the collating sequence you specified.

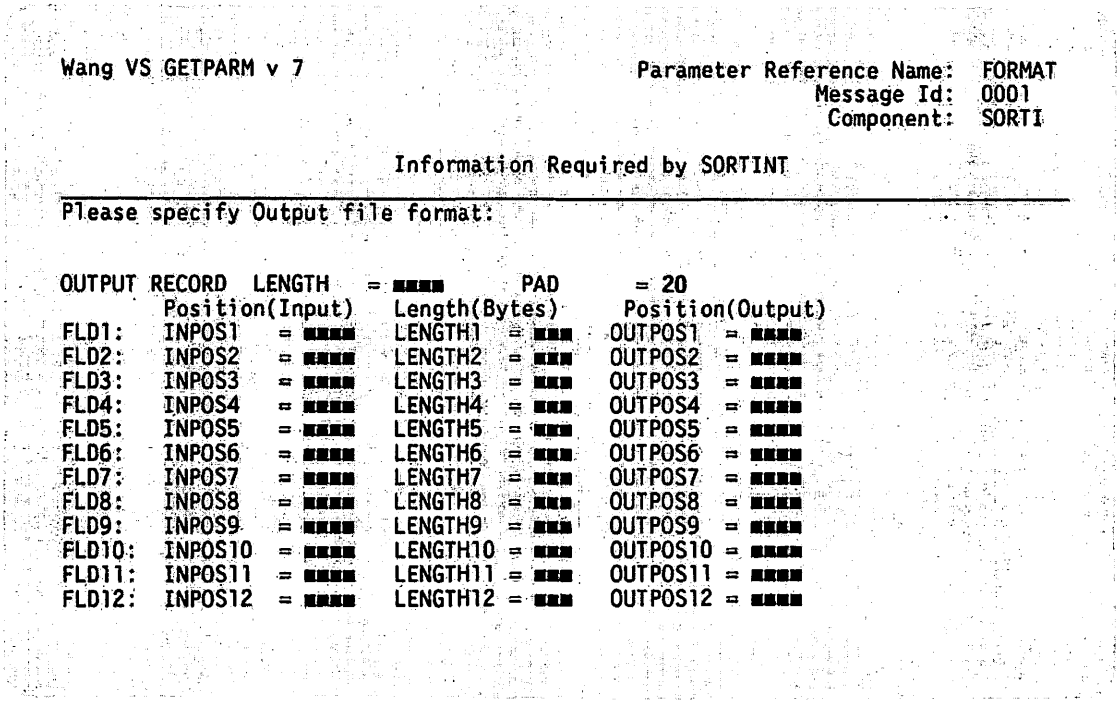
If you define more than one key from the Sort/Merge Keys screen, SORTINT uses the keys in the order in which they are defined; that is, the sort is done first on Key 1, then on Key 2, then on Key 3, and so on. When you finish defining Sort/Merge keys, press ENTER to begin the operation.

Table 15-2. Data Types and Length Restrictions for SORTINT Key Fields

Code	Type	Key Field Length Restriction
B	Binary	2, 4, or 8 bytes exactly
U	Unsigned binary	2, 4, or 8 bytes exactly
C	Character data (alphanumeric)	256 bytes maximum
D	External decimal signed or unsigned; sign trailing or leading in separate byte	19 bytes maximum
F	Floating point	8 bytes exactly
Z	Zoned decimal, sign trailing included in last byte	18 bytes maximum
L	Zoned decimal, sign leading included in first byte (overpunched)	18 bytes maximum
P	Packed decimal, sign trailing included in last byte	16 bytes maximum

15.5 SPECIFYING THE OUTPUT RECORD FORMAT

If you entered YES in the Reformat field of the Options screen (Figure 15-1) to indicate that you wanted the fields in the output record arranged in a different order from the fields in the input record, the Record Format screen (Figure 15-5) appears.



```
Wang VS GETPARM v 7                          Parameter Reference Name:  FORMAT
                                                          Message Id:  0001
                                                           Component:  SORTI

                          Information Required by SORTINT
_____________________________________________________________________________
Please specify Output file format:

OUTPUT RECORD  LENGTH   = ____      PAD      = 20
       Position(Input)   Length(Bytes)     Position(Output)
FLD1:  INPOS1   = ____   LENGTH1  = ___   OUTPOS1  = ____
FLD2:  INPOS2   = ____   LENGTH2  = ___   OUTPOS2  = ____
FLD3:  INPOS3   = ____   LENGTH3  = ___   OUTPOS3  = ____
FLD4:  INPOS4   = ____   LENGTH4  = ___   OUTPOS4  = ____
FLD5:  INPOS5   = ____   LENGTH5  = ___   OUTPOS5  = ____
FLD6:  INPOS6   = ____   LENGTH6  = ___   OUTPOS6  = ____
FLD7:  INPOS7   = ____   LENGTH7  = ___   OUTPOS7  = ____
FLD8:  INPOS8   = ____   LENGTH8  = ___   OUTPOS8  = ____
FLD9:  INPOS9   = ____   LENGTH9  = ___   OUTPOS9  = ____
FLD10: INPOS10  = ____   LENGTH10 = ___   OUTPOS10 = ____
FLD11: INPOS11  = ____   LENGTH11 = ___   OUTPOS11 = ____
FLD12: INPOS12  = ____   LENGTH12 = ___   OUTPOS12 = ____
```

Figure 15-5. SORT Record Format Screen

Use this screen to specify a new order for the fields in the output record; at your option, you can also change the record length and specify a pad character to fill out a lengthened record. The current position, length, and new position of each field are entered in a set of three fields that share a numeric suffix (represented by *x* in the following list).

Enter information in the fields as follows:

OUTPUT RECORD LENGTH -- Enter the length you want the output record to have. You can make it longer or shorter than the input record -- shorter if you want to truncate or eliminate fields, longer if you want to extend the record in order to add fields at a later time. An extended record is padded with blanks unless you specify a different pad character in the Pad field. There is no default length.

PAD -- If you specify an output record longer than the input record and accept the default value in this field (X'20'), SORT fills every byte in the record that is not part of a currently defined field with a blank. However, you can replace this default pad character with a different character by entering two different hexadecimal digits or a nonblank character in the field.

INPOSx -- Enter the byte position in the input record of the field you want to move, i.e., the position of the first byte of the field (counting the first byte in the record as 1).

LENGTHx -- Enter the length in bytes of the field you want to move. You can truncate or extend the field by entering a length different from its length in the input file.

OUTPOSx -- Enter the byte position of the field in the output record, i.e., the position you want the first byte of the field to occupy (counting the first byte in the record as 1).

Use as many sets of fields on the Record Format screen as you need to rearrange the field positions within the output record to your satisfaction. If you fill in all the fields on the screen, SORTINT displays another Record Format screen when you press ENTER. Pressing ENTER with one or more blank sets of fields on the screen indicates that your specifications for the output record format are complete, and SORTINT goes on to solicit specifications for the output file.

Note: *When you specify new positions for the fields in the output record, be careful that fields are not allowed to overlap. Since SORTINT does not check for this condition, the data in an overlapped field will be lost when the record is reformatted.*

15.6 SPECIFYING THE OUTPUT FILE

After SORTINT (or SORT, if that program has been called) processes the input files, the SORTINT Utility Output screen (Figure 15-6) appears.

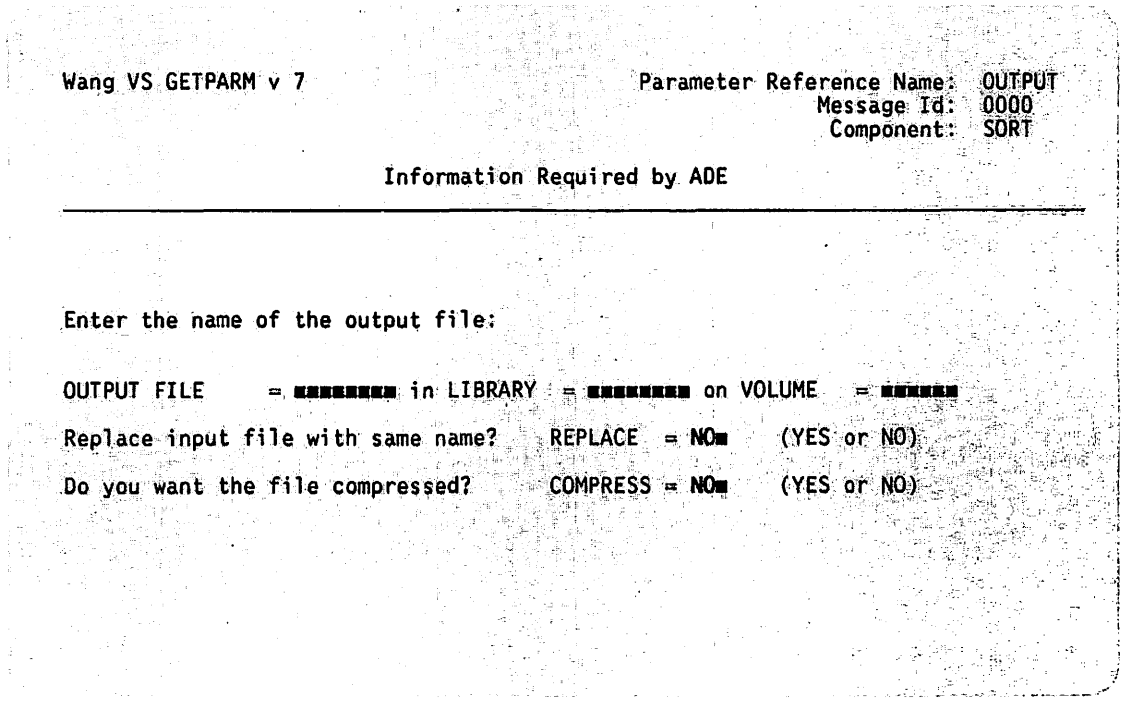


Figure 15-6. SORTINT Utility Output Screen

Use this screen to specify the file in which you would like the sorted output to be placed. Enter information in the fields as follows:

OUTPUT FILE -- The name of the output file to be created. If you specify the same name, library, and volume for the output file as for one of the input files, SORT displays a message warning you that this action causes the sorted output file to replace the input file. If you are sure you want the input file replaced, press PF3 to delete the original input file. The sorted output file then takes its place.

LIBRARY -- The name of the library in which you want to place the output file. The default value is taken from the OUTLIB field of your usage constants.

VOLUME -- The name of the volume on which to place the output file. The default value is taken from the OUTVOL field of your usage constants.

REPLACE -- Specify YES or NO to indicate whether you want SORTINT to replace the input file with the sorted output file without prompting you to confirm the replacement.

If you enter YES in the Replace field, you must specify output file, library, and volume names that are identical with those of the input file. If you enter NO in this field, you must specify an output file, library, or volume name that differs from that of the input file. The default value for Replace is NO.

The Replace option is not available, and the field does not appear on the screen, under the following conditions:

- You specified a Merge operation.
- You are sorting more than one input file.
- The input file is indexed.
- The input file is opened in Shared mode.
- You are creating an Addrout or Keyout file.

COMPRESS -- Specify YES or NO to indicate whether you want SORT to compress data on the output file. Compression can save disk space. The default value is YES for variable-length input and NO for fixed-length input. If the input file is variable-length and the output file is to be fixed-length, however, COMPRESS should be NO.

When you finish entering specifications, press ENTER. SORTINT creates the output file.

15.7 RESTARTING THE SORTINT UTILITY

SORTINT sometimes halts a Sort or Merge operation because it discovers that the input file contains more records than the program is prepared for. This problem comes about in the following way.

Before it begins processing an input file, SORTINT creates a temporary work file on disk to hold the records it is processing. The size of this work file is determined by the number of records the input file contains, according to the disk Volume Table of Contents (VTOC).

Once processing begins, if SORT encounters more records in the input file than it is prepared for, it does the following:

1. Halts the sorting or merging operation.

2. Reads through the input file to determine the number of records it contains.

3. Displays an error message. The error message screen reports the actual number of records in the file and offers you the following choices:

   - Pressing PF1 to restart the operation with a new work file based on the actual record count. It is unnecessary to reenter your specifications, since SORT has saved them.

   - Pressing PF16 to exit from SORT without creating an output file.

Since SORTINT can complete the operation successfully without harm to the input file, there is no reason why you should not press PF1. *You should be aware, however, that a disparity between the number of records reported in the VTOC and the number of records actually in the file indicates that the input file may be damaged.*

Once SORT processing is finished, therefore, you should use the COPY utility to reorganize the input file if you intend to use it further. See the *VS Media, Transfer, and Device Utilities Reference* for information on file reorganization with the COPY utility.

15.8 A SAMPLE SORTINT PROCEDURE

SORTINT processing can be controlled through the VS Procedure Language. In a procedure, you can specify all SORTINT input and output options, selection criteria, and sort keys. Appendix A contains a complete list of SORTINT GETPARMs, listing parameter reference names (prnames) and all fields. See the *VS Procedure Language Reference* for details about VS Procedure Language syntax.

The sample SORTINT procedure that follows illustrates the use of the External and Reformat options. The output file is sorted in descending order. The record selection criterion specifies that the value of the first byte of the input record must be greater than a literal value of A. Only records that meet this criterion are included in the sort.

Note: *The value in VALUE1 is enclosed in single quotation marks within double quotation marks. 'A' is a literal value specified from the Select screen. Because VS Procedure Language uses quotation marks as delimiters, both sets of quotation marks are necessary. If you eliminate one set of quotation marks when writing the SORT procedure, or if both sets of quotation marks are of the same type (i.e., either single or double), the procedure returns an error message indicating incorrect punctuation, with a return code of 4. (Appendix B explains return codes.)*

```
PROCEDURE
  RUN SORTINT
    ENTER OPTIONS EXTERNAL = YES, REFORMAT = YES
    ENTER INTABLE FILE = SAVEX, LIBRARY = USRCOPY,
      VOLUME = SYSTEM, SELECT = YES
    ENTER INPUT FILE = BROKER, LIBRARY = USRDATA, VOLUME = SYSTEM,
      SELECT = YES
    ENTER SELECT FLDPOS1 = 1, LENGTH1 = 1, TSTREL1 = GE,
      VALUE1 = "'A'"
    ENTER KEYS POST1 = 1, LENGTH = 1, ORDER1 = D
    ENTER FORMAT LENGTH = 100, PAD = 20, INPOS1 = 1, LENGTH1 = 10,
      OUTPOS1 = 1, INPOS2 = 7, LENGTH2 = 10, OUTPOST2 = 11,
      INPOS3 = 20, LENGTH3 = 1, OUTPOST3 = 30
    ENTER OUTPUT FILE = SAVEZ, LIBRARY = USRSORT, VOLUME = SYSTEM
RETURN
```

CHAPTER 16
TABLEDIT

16.1 INTRODUCTION

The TABLEDIT (Table Edit) utility enables you to create or modify files that contain collating sequence tables and case flip tables.

- A *collating sequence table* enables you to define a specific sort order or international character set other than the standard American Standard Code for Information Interchange (ASCII) collating sequence and character set. After you create a new collating sequence table, you can direct the SORTINT utility to use that table for sorting files. See Chapter 15 for information on using SORTINT.

- A *case-flip table* is a character translation table used by a software application to translate lowercase letters to uppercase or vice-versa. In a character translation table, you specify, for each character you want to have translated, the character you want it translated to (for example, *ä* to *a*). TABLEDIT's flip table option works the same way. You can use it to construct a table that changes the cases of letters, but you can use it just as easily to edit any character translation table, or to construct a new one.

16.1.1 Overview

Figure 16-1 shows an overview of TABLEDIT processing.

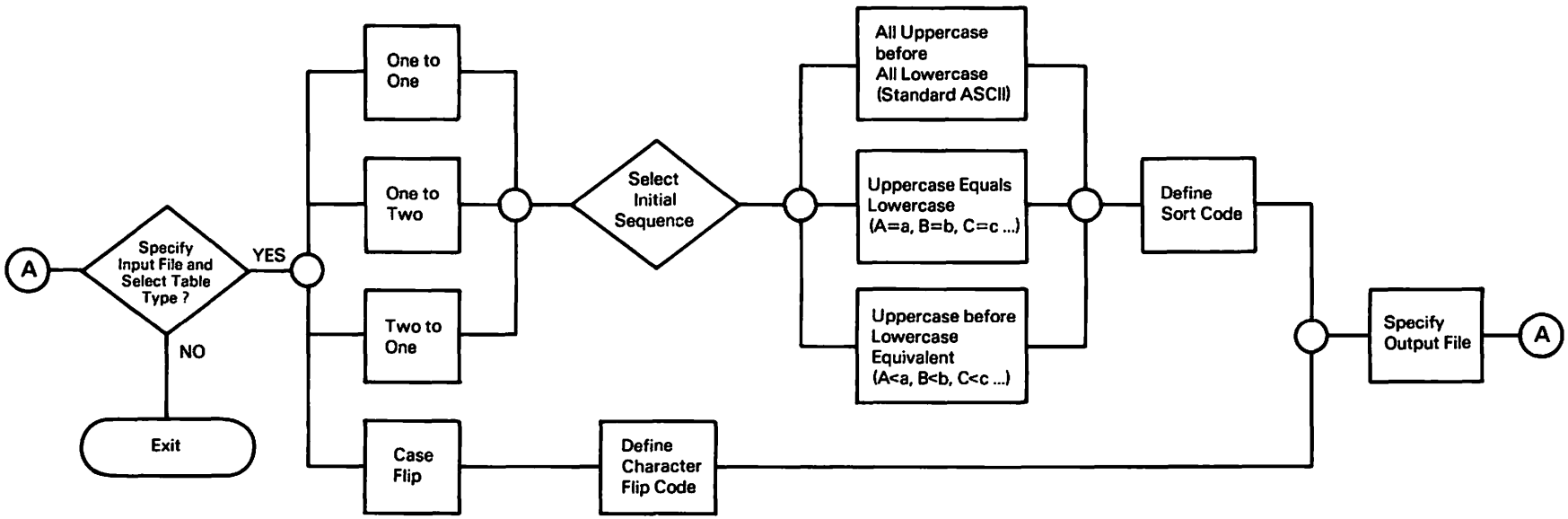


Figure 16-1. TABLEDIT Processing

16.1.3 Running TABLEDIT

To run the TABLEDIT utility, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify TABLEDIT in the Program field and press ENTER.

The following sections describe TABLEDIT processing:

Section	Process
16.2	Specifying the Table
16.3	Selecting a Preliminary Sequence
16.4	Editing the Table
16.4.1	Using the Table Editing Screen
16.4.2	Using the Table Display Screen
16.4.3	Editing Procedure
16.5	Specifying the Output File

16.2 SPECIFYING THE TABLE

When TABLEDIT processing begins, the Table Specification screen (Figure 16-2) appears.

```
TABLEDIT (c) Copyright, 1985 Wang Laboratories, Inc          Select Table Type
Wang VS Table Modification Program, Version x.x                 PrName = TYPE

To edit an existing table, specify the file, library and volume.
To create a new table, leave the fields blank.

The existing table is in      File = ________
                           Library = ________
                            Volume = ______

Select the type of table by pressing the appropriate PF Key:

        (1)  Collating Sequence table - 1 to 1 only
        (2)  Collating Sequence table - 1 to 2 equivalences
        (3)  Collating Sequence table - 2 to 1 equivalences
        (4)  Case-Flip table

                                                          (16) Exit
```

Figure 16-2. TABLEDIT Table Specification Screen

Use this screen to specify the type of table you want to create or edit. You can create a new table or specify an existing table for editing.

To create a new table, leave the File, Library, and Volume fields blank. If you want to edit an existing table, enter information in the fields as follows:

File -- The name of the file that contains the collating sequence or case flip table you want to edit

Library -- The name of the library in which the table file resides

Volume -- The name of the volume on which the table file resides

After you have specified an existing table or created a new one, press the appropriate PF key to specify the type of table you are editing or creating.

PF Key	Table Type and Description
1	**Collating sequence table - 1 to 1** -- A table that defines a sort order in which one character is always treated as one character
2	**Collating sequence table - 1 to 2** -- A table that defines a sort order in which one character can be treated as two characters (for example, ä as *ae*)
3	**Collating sequence table - 2 to 1** -- A table that defines a sort order in which two characters can be treated as one (for example, *ae* as ä)
4	**Case-flip table** -- A character-for-character translation table (such as a table that translates each lowercase letter into its uppercase equivalent)

To exit from TABLEDIT without creating or editing a table, press PF16.

When you press a PF key other than PF16 from the Table Specification screen, one of the following two things happens:

- If you are creating a new collating sequence table, the Initial Sequence screen appears, as described in Section 16.3.
- If you are creating a new case-flip table or editing an existing table of any kind, the Table Editing screen appears, as described in Section 16.4.

16.3 SELECTING A PRELIMINARY SEQUENCE

If you are creating a new collating sequence table, the Initial Sequence screen (Figure 16-3) appears when you press PF1, PF2, or PF3 from the Table Specification screen.

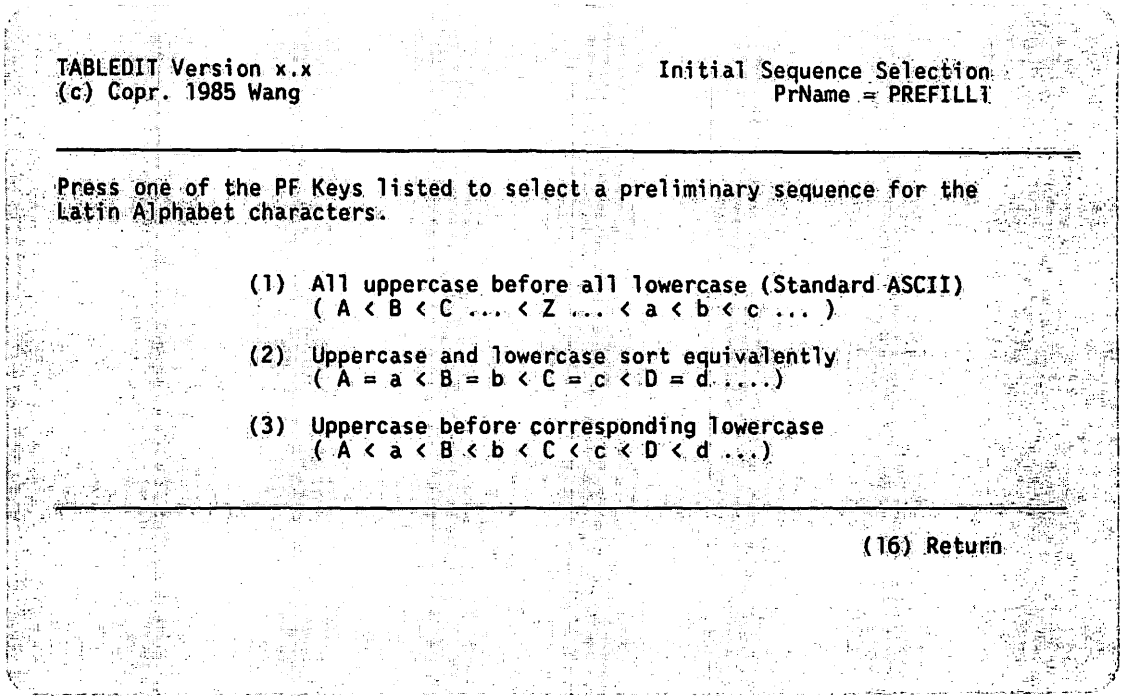


Figure 16-3. TABLEDIT Initial Sequence Screen

Using the Initial Sequence screen, you can specify the general relationship between uppercase and lowercase letters for the entire table and save yourself the labor of changing specifications for each letter individually. The sequence you select from this screen affects only the letters of the Latin alphabet; other characters in the set are not affected. You have the following choices:

PF Key	Option and Description
1	**All uppercase before all lowercase** -- The standard ASCII sort order, in which lowercase characters come after uppercase (*a*, *b*, and *c* come after *X*, *Y*, and *Z*).
2	**Uppercase and lowercase equivalent** -- A "case-insensitve" sort order in which *a* is the same as A, *b* is the same as *B*, and so on.
3	**Uppercase before corresponding lowercase** -- A sort order in which uppercase precedes lowercase *for each letter* (*Az* precedes *aa*, *az* precedes *Ba*, and so on).

The following three lists illustrate the differences among the three sequence options:

Uppercase First (ASCII) -- PF1	Case-Insensitive Order -- PF2	Uppercase First, Letter by Letter -- PF3
Athens	apples	Athens
Peoria	Athens	apples
SORTINT	pears	Peoria
Saratoga	Peoria	pears
apples	Saratoga	SORTINT
pears	SORTINT	Saratoga

To return to the Table Specification screen without selecting a preliminary sequence, press PF16.

16.4 EDITING THE TABLE

16.4.1 Using the Table Editing Screen

After you have selected the preliminary sequence for a new collating sequence table, or immediately after you specify the table type in all other cases, the Table Editing screen appears. Figure 16-4 shows a sample of this screen.

```
TABLEDIT Version 1.0                                    Edit the Table
(c) Copr. 1985 Wang                                      PrName = EDIT

        in                  on          ( One to Two table with 0  equivalences )

   CRT      SORT      !   CRT      SORT      !   CRT      SORT      !   CRT      SORT
   CHAR HEX CODE      !   CHAR HEX CODE      !   CHAR HEX CODE      !   CHAR HEX CODE
                      !                      !                      !
   H    48  4F        !   P    50  5F        !   X    58  6F        !   ◄    60  7A
   I    49  51        !   Q    51  61        !   Y    59  71        !   a    61  42
   J    4A  53        !   R    52  63        !   Z    5A  73        !   b    62  44
   K    4B  55        !   S    53  65        !   [    5B  75        !   c    63  46
   L    4C  57        !   T    54  67        !   \    5C  76        !   d    64  48
   M    4D  59        !   U    55  69        !   ]    5D  77        !   e    65  4A
   N    4E  5B        !   V    56  6B        !   ^    5E  78        !   f    66  4C
   O    4F  5D        !   W    57  6D        !   _    5F  79        !   g    67  4E
                      !                      !                      !

ENTER Summary         (5) Right              (9) Save
  (2) First           (6) Increment ■■                              (14) Show One to Two
  (3) Last            (7) Define
  (4) Left                                                          (16) Cancel
```

Figure 16-4. A Sample TABLEDIT Table Editing Screen

The appearance of the Table Editing screen varies slightly depending on the table type; the differences are described in the sections that follow. You can use this screen to modify the table to your specifications.

The Table Editing screen displays the characters of the table in columns that present three items of information for each character:

CRT Char -- The character as displayed on the workstation screen or CRT (cathode ray tube).

Hex -- The hexadecimal expression of the byte value by means of which the system represents this character.

Sort Code *(collating sequence table only)* -- A hexadecimal number that indicates the position of the character in the sort order (the lower the number, the greater the priority). By modifying this number, you can change the position.

- In a standard, unmodified ASCII sequence (selected by pressing PF1 from the Initial Sequence screen), the value in this column is always identical with the value in the Hex column.

- In a case-insensitive sort sequence (selected by pressing PF2 from the Initial Sequence screen), both *A* and *a* have the value 41, *B* and *b* have 42, and so on.

- In an uppercase first, letter by letter sort sequence (selected by pressing PF3 from the Initial Sequence screen), each lowercase letter has the even number that follows the odd number assigned to the corresponding uppercase letter: *A* is 41, *a* is 42, *B* is 43, *b* is 44, and so on. (Figure 16-4 illustrates this type of sequence.)

A Table Editing screen for a case-flip table has a Case Flip column in place of the Sort Code column:

Case Flip *(case-flip table only)* -- Instead of a sort sequence number (which is irrelevant to a case-flip table), this column is provided for you to enter the hexadecimal value of the character into which you want the character identified in the first two columns to be translated. (The default value for all characters when the screen initially appears is X'20', representing a blank.) To flip all lowercase letters to uppercase, you would enter the values 41 through 5A in the case flip column for *both* *A* through *Z* and *a* through *z*. To flip all uppercase letters to lowercase, you would enter the values 61 through 7A for *both* sequences of letters.

The Table Editing screen shows the entire VS ASCII character set in a series of 16 panels. You can use PF keys (as described in the next section) for shifting the display horizontally, three panels at a time. (To provide continuity, one panel from the previous display remains visible at the right or left of the screen, depending on the direction in which you are moving.) In this way you can gain access to and edit any part of the table.

Besides entering new values in the Sort Code or Case Flip column of the Table Editing screen, you can use the following options for editing the table and manipulating the display:

PF Key	Option and Description
ENTER	**Summary** -- Switch to the Table Display screen (Figure 16-5) to see an illustration of the collating sequence you are defining or editing. (See Section 16.4.2.)
2	**First** -- Move the display to the first set of characters in the table (the leftmost four panels). If these four panels are already displayed, PF2 has no effect.
3	**Last** -- Move the display to the last set of characters in the table (the rightmost four panels). If these four panels are already displayed, PF3 has no effect.
4	**Left** -- Move the display one screen (three panels) to the left. If the leftmost screen is already displayed, PF4 has no effect.
5	**Right** -- Move the display one screen (three panels) to the right. If the rightmost screen is already displayed, PF5 has no effect.
6	**Increment** -- Add 1 to all values in the sort code column from the specified value to the end of the table. See the section entitled "The Increment Key" following this list.
7	**Define** -- Define a specific two-to-one or one-to-two equivalence (depending on the type of table you are editing). For more information, see the section entitled "The Define Key" following this list. *This option is not available for one-to-one collating sequence tables or case-flip tables.*
8	**Delete** -- Delete a specific two-to-one or one-to-two equivalence (depending on the type of table you are editing). *This option is not available for one-to-one collating sequence tables or case-flip tables.*

PF Key	Option and Description
9	**Save** -- Save the edited table. PF9 indicates that you are finished editing and want to keep the table with the changes you have made. From a screen that appears later (see Section 16.5), you can assign the table to a file.
14	**Show One to Two** or **Show Two to One** -- List the one-to-two or two-to-one equivalences currently defined (depending on the type of table you are editing). *This option is not available for one-to-one collating sequence tables or case-flip tables.*
16	**Cancel** -- Return to the Table Specification screen (Figure 16-2) without saving any of the changes you have made. (A message prompts you to confirm this decision by pressing PF16 a second time.)

The Increment Key

PF6 enables you to open a position in the sort sequence (or character translation table) so that you can insert one character between two others that were previously adjacent. Say, for example, that in a standard ASCII collating sequence table you want to place the character ä (position 15) between *a* and *b* (positions 61 and 62) in the sort sequence. The increment key enables you to make room between these two characters in the following manner:

1. Place the cursor on the 2-character field after "Increment" in the menu at the bottom of the screen and enter 62. Press PF6. TABLEDIT adds 1 to 62 and to every number that comes after it in the sequence (b goes from 62 to 63, c from 63 to 64, and so on). This has the effect of creating a gap at 62 to which no character is currently assigned.

2. Move the cursor to the Sort Code column for the character ä and change the value from 15 to 62. The sort sequence (as you can confirm by pressing ENTER to see the Table Display screen) is now *a* > ä > *b*.

Note: *When you insert a character in a new place in the sort order, be sure to create the space by using the Increment key* before *you assign the new value to the character you want to insert. In the example above, if* ä *had been changed from 15 to 62 before the values from 62 upwards were incremented,* ä *and* b *would be equivalent -- that is, they would share position 62 before the increment and position 63 afterwards. In a sort, equivalent characters are not distinguished, but are processed in the order in which they are encountered.*

The Define Key

PF7 is used to assign equivalences in a two-to-one or one-to-two collating sequence table. (You can use this key without regard to the position of the cursor or to whether the characters used in the definition are currently visible on the screen.)

When you press PF7 from a one-to-two collating sequence table, the following fields appear at the bottom of the screen:

```
CRT Char ■  or Hex AA    = Sort Code AAAA
```

To identify the single character to which you want to assign the equivalence, you have the choice of entering it directly in the CRT Char field (provided it is a character that can be entered directly from your keyboard), or entering its hexadecimal equivalent (taken from the Hex column) in the Hex field. The equivalence is assigned in the Sort Code field; here you should enter the Sort Code values of the two characters to which you want this single character to be equivalent in the redefined sort sequence. For example, to define ö as equivalent to *oe*, you would enter 18 in the Hex field (note that ö cannot be typed directly) and the sort codes for *o* and *e* (without spaces or delimiters) in the Sort Code field.

Note: *Unlike the Hex values, the Sort Code value for a given character may vary depending on the initial sequence you specified. See Section 16.4.3 for more information.*

Defining an equivalence from a two-to-one collating sequence table is a very similar process. The following fields appear at the bottom of the screen when you press PF7:

```
CRT Char AA or Hex AAAA = Sort Code AA
```

The only difference is that you have two characters to enter in either of the first two fields, and the Sort Code value for one character to enter in the last field. For example, to define *ae* as equivalent to *e*, you could enter ae in the CRT Char field or 6165 in the Hex field, and the Sort Code value of *e* in the Sort Code field. *As above, note that Sort Code values may vary according to the type of initial sequence specified.*

When you use PF7 more than once, the last equivalence defined supplies default values for all three fields. Be sure to specify or else blank out both the CRT Char and Hex fields before you press ENTER to define another equivalence. Conflicting values in these fields cause an error message.

Whenever you enter the definition of an equivalence, TABLEDIT displays a screen that shows all the equivalences currently defined for the table, including the one you just entered. It is the same screen you see if you press PF14, except that you have the option of either pressing ENTER to confirm the current definition, or pressing PF16 to cancel it.

16.4.2 Using the Table Display Screen

When you press ENTER from the Table Editing screen, the Table Display screen appears. Figure 16-5 shows an example of this screen.

```
TABLEDIT Version 1.0                                    Display Table
(c) Copr. 1985 Wang                                  PrName = SUMMARY

TESTFILE in USRTABLE on VOL111 .

  < ♦ < ► < ◄ < ↔ < _ < | < ¨ < ´ < ~ < ° < ■ < ‼ < ↕ < ß < ¶ < â < ê < î < ô
< û < ë < ï < à < è < ù <   < ! < " < # < $ < % < & < ' < ( < ) < * < + < , <
- < . < / < 0 < 1 < 2 < 3 < 4 < 5 < 6 < 7 < 8 < 9 < : < ; < < < = < > < ? < @
< A ( Ä = Ae ) < B  < C < D < E < F < G < H < I < J < K < L < M < N < O ( Ö =
Oe ) < P  < Q < R < S < T < U ( Ü = Ue ) < V  < W < X < Y < Z < [ < \ < ] < †
< _ < ° < a ( ä = ae ) < b  < c < d < e < f < g < h < i < j < k < l < m < n <
o ( ö = oe ) < p  < q < r < s < t < u ( ü = ue ) < v  < w < x < y < z < § < ‼
< é < ç < ¢

ENTER  Continue
```

Figure 16-5. A Sample TABLEDIT Table Display Screen

This screen illustrates the sort order and character equivalences for the table you are defining. The characters are displayed in a relational string, with characters whose values are identical separated by an equal sign (=) and characters whose values differ separated by a less than sign (<). For example, if *ä* is defined as equivalent to *a*, the display shows ° < *ä* = *a* < *b* instead of simply ° < *a* < *b*. This notation indicates that a and *ä* are treated as if they were the same character, which comes after ° and before *b*.

The Table Display screen is organized so that by reading from left to right you can follow the sort sequence from the first character sorted to the last.

In the display, an *equal sign* indicates that the characters on either side have identical Sort Code values and are therefore sorted as if they were the same character.

A *less than sign* indicates that the character to the left of the sign has a lower value than the character to the right, and is therefore sorted first.

For example, *A = a < B = b* means that neither A nor *a* takes precedence over the other in a sort (since both have the Sort Code value 41), but that *A* and *a* are sorted before *B* or *b* (42). Similarly, *B* and *b*, having the same code value, are sorted as if they were identical. (This example illustrates a case-insensitive search.)

When upper- and lowercase letters are treated as identical, items that differ *only* in case (for instance *RICHARD, Richard,* and *richard*) are not ordered with respect to one another. Instead, they appear in the output file in the order in which they were read from the input file (or files).

The same is true of all one-to-one character equivalences. When *Ä* is defined as equivalent to *A*, items that differ only in having one equivalent in place of the other (for instance *Apfel* and *Äpfel*) are treated as identical. Their relative position in the output file depends only on the order in which they were encountered.

Figure 16-5, which represents a collating sequence table with one-to-two equivalence, shows that the characters *Ä, Ö, Ü, ä, ö,* and *ü* have been redefined as equivalent to *Ae, Oe, Ue, ae, oe,* and *ue* respectively. The table represents this relationship by showing (for example) *= < A (Ä = Ae) < B* in the sort precedence. The parentheses indicate that *Ä* is not treated as if it were identical to *A*, but instead is sorted between *A* and *B* (or, to be more precise, between *Ad* and *Af*, as if it were an *A* followed by an *e*).

Although the characters in the sort table itself are highlighted on the Table Display screen (while the characters that indicate their relationships are displayed in normal intensity), this screen has no modifiable fields to specify, nor are there any options to select. The purpose of the Table Display screen is to show the sort precedence at any given stage in process of table definition, so that you can verify the effect of any modifications you have entered from the Table Editing screen.

The only action you can take from this screen is to press ENTER and return to the Table Edit screen.

16.4.3 Editing Procedure

In the Table Editing screen, to move a character to a different place in the collating sequence (for instance, to have è sorted between e and f instead of between à and ù), you need only enter a new sort code in the Sort Code column for the character you want to move.

- If you want it treated identically with another character (i.e., moved to the same place in the sequence), enter the sort code that is assigned to this other character in the table.

- If you want it to occupy a unique position between two other characters, identical with neither of them, use PF6 (as described in Section 16.4.1 under the heading "The Increment Key") to create a gap by freeing a code between these two numbers which you can assign to the character you want to move.

You can move an item to the end of the sort order simply by assigning it a sort code greater than 7F. For instance, to have digits (*0 - 9*) sorted *after* letters of the alphabet (both upper- and lowercase) instead of before them, you could enter 80 as the sort code for *0*, 81 for *1*, and so on. (If the items to be sorted will include other characters as well as digits and letters, however, you must check the table to make sure that their positions in the sort sequence are not changed in a way that is contrary to your intention.)

Modifications of this kind, which involve single characters only, can be made in any type of table. One-to-two and two-to-one equivalence tables also let you define one character as equivalent to two, or vice-versa. See "The Define Key" in Section 16.4.1 for information about this procedure, which is unique to these two types of table.

Using the Correct Sort Code

Whenever you modify a collating sequence table you enter a sort code to indicate the "destination" of the move (i.e., the new position of the specified character or characters). This is a simple operation, but it must be done with care.

Whenever you enter a new value in the Sort Code field to change the position of a character in the collating sequence or to define a one-to-two or two-to-one equivalence, *be sure that the new Sort Code value you are entering is correct*. This may require some preliminary research. If, for example, you define ü as equivalent to *u*, you have to enter the sort code for *u* in the Sort Code column opposite ü. From this position in the table it is impossible to see and verify the value you are entering.

TABLEDIT sets the values in the Sort Code column according to the initial sequence you specify, and for many letters this means a different sort code for each sequence (see the description of the Sort Code field in Section 16.3).

Suppose, for example, that you are defining the equivalence *ä = ae* in a one-to-two collating sequence table. After you press PF7 (the Define key), entering the sort code for *ae* in the Sort Code field means that you enter 6165 for a standard ASCII sequence, 4145 for a case-insensitive sequence, and 424A for a sequence in which lowercase follows uppercase letter by letter.

You should also take care not to enter the value from the Hex column when the value from the Sort Code column is called for. For only one kind of sequence -- the standard ASCII sort order specified by pressing PF1 from the Initial Sequence screen -- the values in the Hex and Sort Order columns are identical when the Table Editing screen first appears. However, this is true only until the table is modified. *No matter what initial sequence you choose, therefore, getting the Hex value confused with the Sort Order value for a given character is likely to cause an error.*

Although the initial sequence specification affects only letters, the modifications you make to the table are likely to change the sort values of other characters as well. (The operation of the Increment key, PF6, is an obvious example.)

The safest course, therefore, is to check each sort code you intend to specify (that is, the sort code for the character that will appear to the *right* of the equals sign in an expression like "*é = e*") immediately before you enter it.

You should also check the Table Display screen frequently to make sure the changes you are entering from the Table Editing screen are having the intended results.

When you have finished editing the table and are ready to save the modified version in a file, press PF9.

16.5 SPECIFYING THE OUTPUT FILE

When you press PF9 from the Table Editing screen to save the edited version of the collating sequence or case-flip table, the Output File Specification screen (Figure 16-6) appears.

```
TABLEDIT Version x.x                                  Save Table in a File
                                                        PrName = Save
_____________________________________________________________________

Please Specify Information to identify a file where the table can be saved.

                        File = ________
                     Library = ________
                      Volume = ______

_____________________________________________________________________
(ENTER) Save

    (3) Replace Input File                    (16) Return to Edit
```

Figure 16-6. TABLEDIT Output Specification Screen

If you have been editing an existing table housed in a file that you specified from the Table Specification screen (Figure 16-2), you can replace the original version with the edited version by pressing PF3. Otherwise, specify a new output file by entering information in the fields as follows:

File -- A valid name for the file that will contain the table you have edited

Library -- The name of a library in which you want the table file to reside

Volume -- The name of the volume on which you want the table file to reside

To return to the Table Editing screen without saving the table, press PF16.

CHAPTER 17
TRANSL

17.1 INTRODUCTION

The TRANSL utility enables you to translate a file from one character code or character set to another. It contains standard tables for translating in either direction between American Standard Code for Information Interchange (ASCII) and Extended Binary Coded Decimal Interchange Code (EBCDIC). TRANSL also allows you to specify (and if necessary to define) an individual translation table by which individual characters can be substituted for one another, making it possible to transfer files among systems with different character sets. In addition to its translation functions, the utility can also be used to select and rearrange fields in the records of a data file.

TRANSL can do the following jobs:

- Translate files from ASCII to EBCDIC or EBCDIC to ASCII character codes. A standard conversion table is used for either translation.

- Translate files to or from a nonstandard character set. TRANSL requires a user-defined translation table for this kind of translation, and it provides means for creating, retrieving, or modifying such a table. If you create a table through TRANSL, you can save it for subsequent file translation.

- Rearrange, insert, or delete input record fields to create a modified output record. Field manipulation of this kind can be combined with file translation or done independently.

- Select for processing particular types of records from a file that contains multiple types.

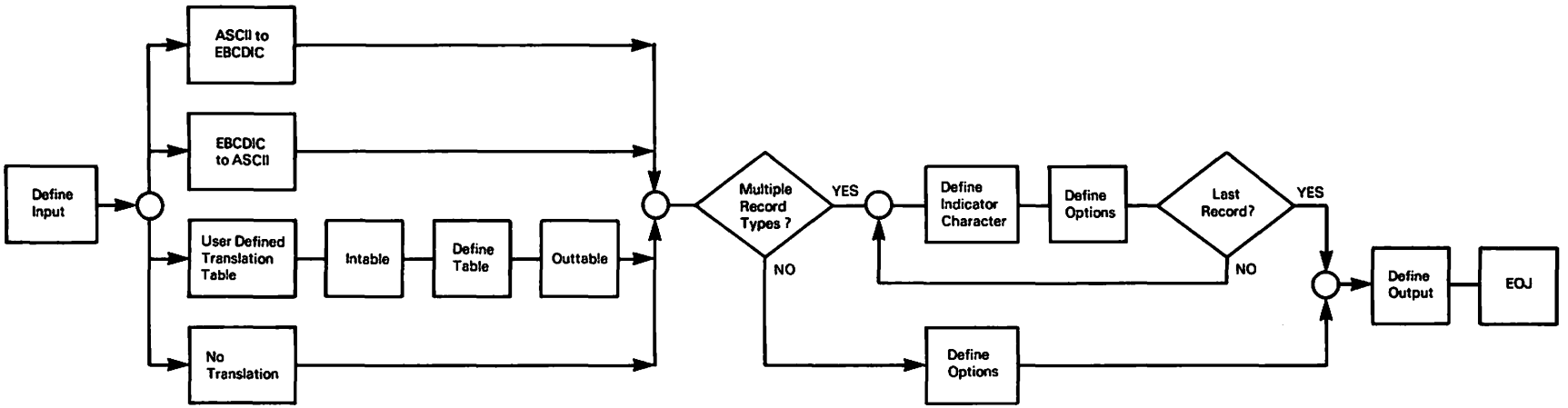


Figure 17-1. TRANSL Processing

17.1.1 Overview

TRANSL processing involves the following steps. All but the first and last are optional.

1. Specifying the input file and a TRANSL function.

2. Creating, selecting, or modifying a user-defined translation table for use in translation functions.

3. Specifying multiple record types so that the records in the input file can be processed selectively.

4. Specifying field manipulation options to edit the output records.

5. Specifying the output file.

Figure 17-1 shows an overview of TRANSL processing.

17.1.2 Running TRANSL

To begin TRANSL processing, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify TRANSL in the Program field and press ENTER.

The following sections describe TRANSL processing:

Section	Process
17.2	Specifying the Input File and Program Function
17.2.1	Standard Translation Functions
17.2.2	User-Defined Translation Tables
17.2.3	Manipulating Fields Without Translation
17.3	Defining a Translation Table
17.4	Specifying Multiple Record Types
17.5	Specifying Field Manipulation Options
17.5.1	Changing Field Order
17.5.2	Changing Field Length
17.5.3	Omitting Fields From the Output Record
17.5.4	Specifying the Primary Key for an Indexed File
17.6	Specifying the Output File
17.7	A Sample TRANSL Procedure

17.2 SPECIFYING THE INPUT FILE AND PROGRAM FUNCTION

When TRANSL processing begins, the Input File and Function screen (Figure 17-2) appears:

```
Wang VS GETPARM v 7                          Parameter Reference Name:  INPUT
                                                          Message Id:  0001
                                                           Component:  TRANSL

                        Information Required by TRANSL

    VS File Translation Utility - Version x.xx.xx  (c) Copyr. Wang 1986

                        Please supply input parameters

        FILE     = ________   LIBRARY = ________    VOLUME   = ______
            TYPES    = NO      (YES/NO for multiple record types)

        Please specify translation to be performed on the output records

            CODE     = EBCDIC    Options: ASCII   -  ASCII-to-EBCDIC
                                          EBCDIC  -  EBCDIC-to-ASCII
                                          TABLE   -  User-supplied table
                                          NONE    -  No translation

    Press ENTER to continue or select:

                                                        (16) To exit TRANSL
```

Figure 17-2. TRANSL Input File and Function Screen

Use this screen to identify the file you want TRANSL to process, to indicate whether it contains multiple record types, and to specify the process (translation or mere reformatting).

Enter information in the fields as follows:

FILE -- The name of the file you want to process.

LIBRARY -- The library in which the file resides. The default value is taken from the INLIB field of your usage constants.

VOLUME -- The volume on which the library resides. The default value is taken from the INVOL field of your usage constants.

TYPES -- Enter YES if the file you want to process contains records of more than one type.

CODE -- Specify the way the file is to be processed:

ASCII -- Translate ASCII-encoded characters to EBCDIC-encoded characters.

EBCDIC -- Translate EBCDIC-encoded characters to ASCII-encoded characters.

TABLE -- Translate specified characters by means of a user-specified translation table. You can create a table or access (and optionally modify) an existing one. This function is used for converting files to or from nonstandard character sets.

NONE -- Do no character translation. This option enables you to use the field manipulation features of the TRANSL utility without translating characters. When you specify NONE, TRANSL bypasses all translation table operations and displays the Field Options screen.

17.2.1 Standard Translation Functions

TRANSL processes both ASCII-to-EBCDIC and EBCDIC-to-ASCII conversions by using a predefined conversion table. For example, if you specify ASCII (for an ASCII-to-EBCDIC conversion) in the Code field, TRANSL creates as output a copy of the input file in which each ASCII character is converted to the corresponding EBCDIC character.

If you specify ASCII or EBCDIC for a file without multiple record types (TYPES = NO), the Field Options screen appears as described in Section 17.5. You must specify field attributes for the output file from this screen before TRANSL can create the output file and do the conversion you requested.

If you specify ASCII or EBCDIC for a file that contains multiple record types (TYPES = YES), the Record Types screen appears as described in Section 17.4 when you press ENTER from the Input File and Function screen. You must define record types before the process of specifying field attributes can begin.

17.2.2 User-Defined Translation Tables

If you enter TABLE in the Code field to indicate that you want to use or create a user-defined translation table, TRANSL displays the Intable screen (Figure 17-3). After you have specified or created the translation table, as described in Section 17.3, processing continues according to whether the input file (as specified from the Input File and Function screen) has multiple record types:

- If TYPES = NO (the input file does not contain multiple record types), the Field Options screen appears, as described in Section 17.5, so that you can specify field attributes for the output file.

- If TYPES = YES (the input file has multiple record types), the Record Types screen appears, as described in Section 17.4. You must define record types before the process of specifying field attributes can begin.

17.2.3 Manipulating Fields Without Translation

If you enter NONE in the Code field to indicate that you do not want any translation operations, all such operations are bypassed. Processing depends on whether the input file has multiple record types, as indicated by the value in the Types field of the Input File and Function screen:

- If TYPES = NO (the input file does not contain multiple record types), the Field Options screen appears, as described in Section 17.5, so that you can specify field attributes for the output file.

- If TYPES = YES (the input file has multiple record types), the Record Types screen appears, as described in Section 17.4. You must define record types before the process of specifying field attributes can begin.

As soon as you finish specifying field attributes for the output file, TRANSL creates it, modifying the records according to your specifications.

17.3 DEFINING A TRANSLATION TABLE

If you specified TABLE in the Code field of the Input File and Function screen, the Intable screen (Figure 17-3) appears when you press ENTER.

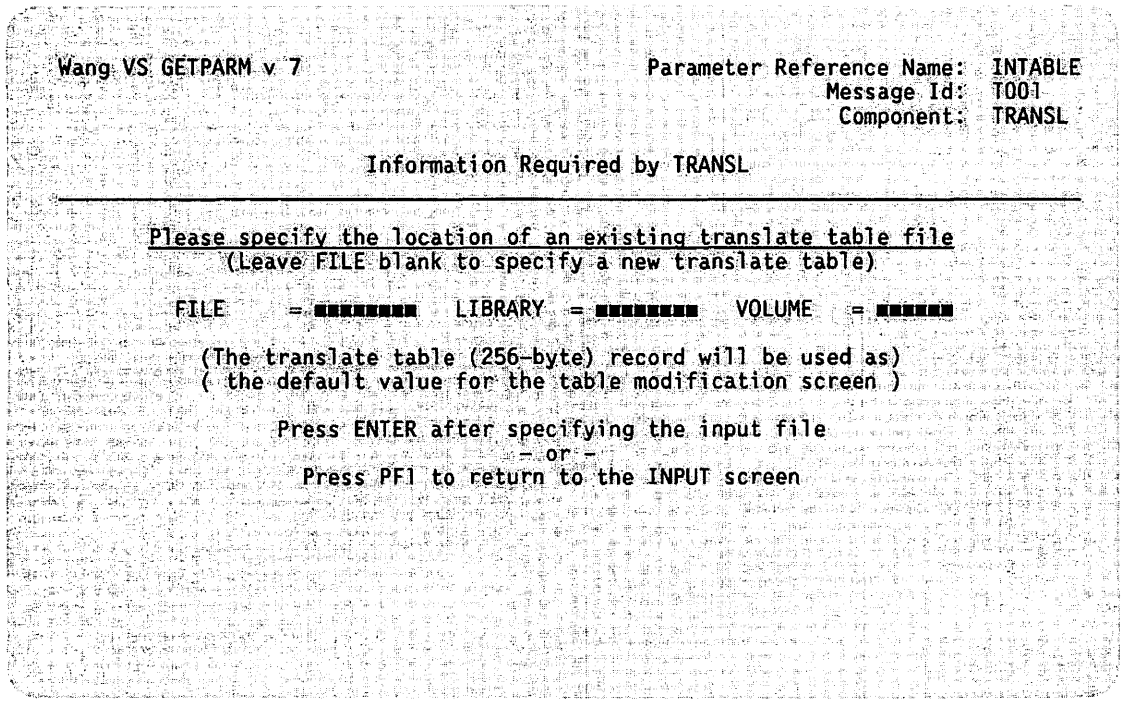


Figure 17-3. TRANSL Intable Screen

Use this screen either to indicate that you want to create a new translation table, or to identify an existing translation table you want to use or modify.

- *To create a new translation table,* leave the File, Library, and Volume fields blank and press ENTER to display a default Translation Table screen.

- *To use or modify an existing translation table,* use the File, Library, and Volume fields to specify the file that contains the translation table and press ENTER to display that table on the Translation Table screen.

The Translation Table screen (Figure 17-4) appears when you press ENTER from the Intable screen.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  TABLE
                                                          Message Id:  T002
                                                           Component:  TRANSL

                        Information Required by TRANSL

Please modify as desired and/or select from choices listed

    HEX00#0F = 00010203  04050607  08090A0B  0C0D0E0F
    HEX00#1F = 10111213  14151617  18191A1B  1C1D1E1F   ------------------
    HEX00#2F = 20212223  24252627  28292A2B  2C2D2E2F  * Please Select:   *
    HEX00#3F = 30313233  34353637  38393A3B  3C3D3E3F  *                  *
    HEX00#4F = 40414243  44454647  58595A5B  5C5D5E5F  * ENTER - Continue *
    HEX00#5F = 50515253  54555657  58595A5B  5C5D5E5F  *                  *
    HEX00#6F = 60616263  64656667  68696A6B  6C6D6E6F  * PF1 - Return to  *
    HEX00#7F = 70717273  74757677  78797A7B  7C7D7E07  *  INTABLE screen  *
    HEX00#8F = 80818283  84858687  88898A8B  8C8D8E8F  *                  *
    HEX00#9F = 90919293  94959697  98999A9B  9C9D9E9F  *                  *
    HEX00#AF = A0A1A2A3  A4A5A6A7  A8A9AAAB  ACADAEAF  * PF14 - Further   *
    HEX00#BF = B0B1B2B3  B4B5B6B7  B8B9BABB  BCBDBEBF  *     information  *
    HEX00#CF = C0C1C2C3  C4C5C6C7  C8C9CACB  CCCDCECF  *                  *
    HEX00#DF = D0D1D2D3  D4D5D6D7  D8D9DADB  DCDDDEDF   ------------------
    HEX00#EF = E0E1E2E3  E4E5E6E7  E8E9EAEB  ECEDEEEF
    HEX00#FF = F0F1F2F3  F4F5F6F7  F8F9FAFB  FCFDFEFF
```

Figure 17-4. TRANSL Translation Table Screen (Default)

A translation table is a 256-byte string used to convert one character set to another. When you are creating a new table or modifying an existing one, the Translation Table screen displays a table that contains 256 two-digit hexadecimal numbers, each of which represents one byte in the string. (For example, the first entry in the illustration shows 00010203 for the first four bytes: 00, 01, 02, and 03.)

The table displays the full set of ASCII codes in hexadecimal notation. To translate a character

1. Find the hexadecimal value of the character you want to translate

2. Enter in its place the hexadecimal value of the character you want it translated to

If, for example, you change 61 in row HEX00#6F to 41, you indicate that the character *a* is to be changed to the character A in the output file.

If you are creating a new translation table, the string that TRANSL displays is a default table in which each character has its standard ASCII hexadecimal value, as illustrated in Figure 17-4. If you used this default table without modifying it, the output file would be unchanged, since each character would simply be "translated" to itself.

Edit the translation table by specifying only the values you want to change. For example, to make a case-flip table in which all lowercase letters are converted to uppercase, change the hexadecimal values 61 through 7A (which represent the characters *a - z*) to 41 through 5A, which represent the characters *A - Z*). When this table is used to convert a file, each character in the lowercase ASCII alphabet is translated into the corresponding uppercase character. Figure 17-5 shows a Translation Table screen modified for this purpose.

```
Wang VS GETPARM v 7                          Parameter Reference Name:  TABLE
                                                          Message Id:  T002
                                                           Component:  TRANSL

                        Information Required by TRANSL

 Please modify as desired and/or select from choices listed

     HEX00#0F = 00010203  04050607  08090A0B  0C0D0E0F
     HEX00#1F = 10111213  14151617  18191A1B  1C1D1E1F   --------------------
     HEX00#2F = 20212223  24252627  28292A2B  2C2D2E2F * Please Select:      *
     HEX00#3F = 30313233  34353637  38393A3B  3C3D3E3F *                     *
     HEX00#4F = 40414243  44454647  58595A5B  5C5D5E5F * ENTER - Continue    *
     HEX00#5F = 50515253  54555657  58595A5B  5C5D5E5F *                     *
     HEX00#6F = 60414243  44454647  48494A4B  4C4D4E4F * PF1 - Return to     *
     HEX00#7F = 50515253  54555657  58595A7B  7C7D7E07 *  INTABLE screen    *
     HEX00#8F = 80818283  84858687  88898A8B  8C8D8E8F *                     *
     HEX00#9F = 90919293  94959697  98999A9B  9C9D9E9F *                     *
     HEX00#AF = A0A1A2A3  A4A5A6A7  A8A9AAAB  ACADAEAF * PF14 - Further      *
     HEX00#BF = B0B1B2B3  B4B5B6B7  B8B9BABB  BCBDBEBF *     information     *
     HEX00#CF = C0C1C2C3  C4C5C6C7  C8C9CACB  CCCDCECF *                     *
     HEX00#DF = D0D1D2D3  D4D5D6D7  D8D9DADB  DCDDDEDF   --------------------
     HEX00#EF = E0E1E2E3  E4E5E6E7  E8E9EAEB  ECEDEEEF
     HEX00#FF = F0F1F2F3  F4F5F6F7  F8F9FAFB  FCFDFEFF
```

Figure 17-5. A Sample TRANSL Translation Table Screen (Modified)

The new values have been entered in rows HEX00#6F and HEX00#7F. Note that they also appear in their original positions, rows HEX00#4F and HEX00#5F. This insures that the uppercase characters in the input file remain unchanged.

If you specified an existing translation table, TRANSL displays that table instead of the default. For every character already translated in the existing table, its new value appears on the screen in place of its standard ASCII value. (For example, if the table illustrated in Figure 17-5 were saved and then recalled, the Translation Table screen would first appear exactly as it does in the illustration.)

To use an existing table just as it is, simply press ENTER; if you want to modify it first, enter new values in the appropriate positions and press ENTER when you finish.

The Outtable screen (Figure 17-6) appears.

```
Wang VS GETPARM v 7                    Parameter reference name: OUTTABLE
                                                    Message ID: T003
                                                     Component: TRANSL

                    Information required by TRANSL

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

        Please specify an output location for this translate table file
                (Press PF16 to continue without saving table)

        FILE     = ________  LIBRARY = ________  VOLUME   = ______

        (The translate table (256-byte) record will be saved in)
        (the location specified above, for future use by TRANSL)

                Press ENTER after specifying the input file
                                - or -
                Press PF1 to return to the TABLE screen
                                - or -
                Press PF2 to replace input file CASEFLIP
```

Figure 17-6. TRANSL Outtable Screen

This screen prompts you to save a translation table you have created or modified. You have the following options:

Use the table without saving it -- If you press PF16, TRANSL goes on to translate the input file, but does not save the table in a disk file. If you have specified an existing table and have not changed it, this option makes the most sense, since the table you are using already exists in a disk file. You might also choose this option if you are modifying a standard table that is not likely to recur.

Use the table and save it -- If your modifications have produced a table you are likely to need again, enter valid file, library, and volume names in the appropriate fields and press ENTER. TRANSL saves the table under the name and location you have supplied.

Replace the existing table -- This option is available if you have been editing an existing table that you specified by file name and location. (The file name appears at the bottom of the screen where the option is announced.) If you want the edited version to replace the original version under the same file name and location, press PF2.

Before selecting any of these options, you can return to review or further modify the Translation Table screen by pressing PF1.

Editing a Translation Table Through a Procedure

If you run the TRANSL utility from a procedure one of whose functions is to modify a translation table, the procedure must specify each field of the Translation Table screen (Figure 17-4) as a whole, the way it is shown on the screen. For example, if you want to change hexadecimal 61 through 7A to hexadecimal 41 through 5A using a procedure, valid ENTER statements are written as follows:

```
ENTER TABLE HEX00#6F = "60414243 44454647 48494A4B 4C4D4E4F",
            HEX00#7F = "50515253 54555657 58595A7B 7C7D7E7F"
```

Even when you are changing only one value in a row, the ENTER statement must specify the entire row, including the default values of the other 15 bytes.

Quotation marks are required only when a space is used as a filler character. Other filler characters are acceptable. For more information about the VS Procedure Language, refer to the *VS Procedure Language Reference*.

17.4 SPECIFYING MULTIPLE RECORD TYPES

If you entered YES in the Type field of the Input File and Function screen (Figure 17-2), the Record Types screen (Figure 17-7) appears

- Immediately, if you entered ASCII, EBCDIC, or NONE in the Code field of the Input File and Function screen
- After you create, edit, or specify a translation table (as described in Section 17.3), if you entered TABLE in the Code field

```
Wang VS GETPARM v 7                          Parameter Reference Name:  TYPES
                                                          Message Id:  1007
                                                           Component:  TRANSL

                        Information Required by TRANSL

Please define the character(s) and location(s) to be used in identifying the
record types in the file.  Each character may be specified as one ASCII
character or two HEX digits.
Leave this screen blank to indicate the end of the different record types.

Record type 01 will be identified by:
                CHAR1    = ■■ in COLUMN1  = ■■■■,  SWITCH1   = YES
            and CHAR2    = ■■ in COLUMN2  = ■■■■,  SWITCH2   = YES
            and CHAR3    = ■■ in COLUMN3  = ■■■■,  SWITCH3   = YES

          The above indicators are as found in which charcter set ?
                    CHARSET  = INPUT■    (INPUT or OUTPUT)

Note:  SWITCH (1,2,3) = "YES", Means that this character's presence will
       be used to identify this record type.  SWITCH (1,2,3) = "NO", Means
       That its absence will be used to identify this record type.
```

Figure 17-7. TRANSL Record Types Screen

Since you have indicated that the input file contains multiple record types, TRANSL must be given sufficient information to identify the record types you want included in the output file. The Record Types screen prompts you for the information needed to identify one such record type. Each time you enter specifications for one record type and press ENTER, TRANSL displays another Record Types screen, until you press ENTER from an unmodified screen to indicate that you have no more record types to specify.

If the input file also contains record types you do *not* want to have included in the output file, leave them unspecified. TRANSL will ignore these records.

TRANSL identifies records by the presence or absence of up to three specific characters in specific positions. For each of these you must specify the character, its column number (i.e., its byte position in the record), and whether the record is identified by the presence or the absence of this character. (TRANSL provides three sets of fields for this information, each set identified by a numeric suffix represented by x in the list below.) You can also specify whether the identifying character or characters are to be looked for in the record before or after the translation is done.

Enter information in the fields as follows:

CHARx -- Enter an ASCII character or a two-digit hexadecimal value as an indicator character that identifies this record type.

Note: *If you want to specify an EBCDIC-encoded character, you must enter its hexadecimal value. A character entered directly is always interpreted as an ASCII value even if the record in which it is to be looked for is EBCDIC. (For example, if the character* A *is entered, TRANSL will search the input or output record for the value X'41' no matter how that record is encoded.) However, you can use your entry in the Charset field to insure that the ASCII record rather than the EBCDIC record is searched.*

COLUMNx -- Enter the byte position in the record in which this character occurs (counting from 1 as the first byte).

SWITCHx -- Indicate whether the record type can be identified by the presence or absence in the position specified in the Column field of the character specified in the Char field. YES (the default value) indicates that the record is identified if the character is present. NO indicates that it is identified if the character is absent (i.e., if any other character occurs in the specified position).

CHARSET -- Specify whether the character(s) identified in the first two fields are to be found in the input or the output record:

INPUT -- (The default value) indicates that the specified characters are present *before* the record is translated.

OUTPUT -- Indicates that the specified characters are present *after* the record is translated.

If you have specified an EBCDIC-to-ASCII translation and have entered ASCII characters or hex values in the Char field, be sure to change the value of this field to OUTPUT. Otherwise, TRANSL will search for ASCII-encoded values in the input record, which contains only EBCDIC-encoded values. (For an ASCII-to-EBCDIC translation, only the input record should be searched for ASCII characters or values.)

In theory, three characters may not be enough to define a record type precisely. In practice, however, the records in files that contain multiple record types are usually identified easily by means of flag fields whose main or only function is to identify the record type. In such cases one character is generally sufficient for a positive identification.

Note: *A file whose multiple record types cannot be distinguished by specifying three characters for each can be treated with the CONDENSE utility, which has more extensive means of identifying record types. CONDENSE can extract individual record types and place them in separate files, or combine fields taken from records of different types to create a new, single record type. These records are placed in an output file TRANSL can process. Like TRANSL, CONDENSE also reformats records. See Chapter 9 for more information about this utility.*

When you finish defining the identifying characteristics of one record, press ENTER. TRANSL displays the Field Options screen (Figure 17-8) for this record type so that you can specify field manipulation options, as described in Section 17.5. When that process is completed, TRANSL displays another Record Types screen. This screen differs from the previous one only in having a higher record type number (on the line immediately above the first set of modifiable fields). Each time you specify another record type, a Field Options screen appears so that you can specify field manipulation options for that record type.

Specify all the record types, with their field manipulation options, that you want to include in the output file. TRANSL will ignore any records it encounters whose types are not specified.

When you have no more record types to specify, press ENTER from the next Record Types screen that appears, without entering any values in the fields. This informs TRANSL that the process of specifying record types and field manipulation options is complete.

17.5 SPECIFYING FIELD MANIPULATION OPTIONS

If you have indicated that your input file contains multiple record types, TRANSL prompts you for field manipulation options for each record type you want it to include in the output file. As soon as you define the characteristics that enable the program to identify the record type (as described in the previous section), a Field Options screen (Figure 17-8) appears. The alternation of Record Types screens (Figure 17-7) and Field Options screens continues until all record types are fully specified for the output file, each record type with its own set of field manipulation options.

If your input file contains only one type of record, the Field Options screen is displayed. It appears

- Immediately after the Input File and Function screen if your entry in the Code field is ASCII, EBCDIC, or NONE
- After the translation table has been fully specified if your entry in the Code field is TABLE

```
Wang VS GETPARM v 7                    Parameter Reference Name:  OPTIONS
                                                    Message Id:  0001
                                                     Component:  TRANSL

                        Information Required by TRANSL

Press ENTER after defining the necessary fields

                                                                  (Code I,D
Field attributes:                                              or leave blank)
       Position(Input)     Length(Bytes)     Type(B,C,P OR Z)   Insert  Delete
FLD1 : POST01   = ____  LENGTH01 = ____  TYPE01   = _      SWITCH01 = _
FLD2 : POST02   = ____  LENGTH02 = ____  TYPE02   = _      SWITCH02 = _
FLD3 : POST03   = ____  LENGTH03 = ____  TYPE03   = _      SWITCH03 = _
FLD4 : POST04   = ____  LENGTH04 = ____  TYPE04   = _      SWITCH04 = _
FLD5 : POST05   = ____  LENGTH05 = ____  TYPE05   = _      SWITCH05 = _
FLD6 : POST06   = ____  LENGTH06 = ____  TYPE06   = _      SWITCH06 = _
FLD7 : POST07   = ____  LENGTH07 = ____  TYPE07   = _      SWITCH07 = _
FLD8 : POST08   = ____  LENGTH08 = ____  TYPE08   = _      SWITCH08 = _

*Options for field type: B=Binary,C=Character,P=Packed decimal,Z=Zoned decimal
Notes: If all 8 fields are used the option to define more fields will appear
  Selecting delete(D) causes this input field not to be transferred to ouput
  Selecting insert(I) creates a new field in output (Leave position zero)
```

Figure 17-8. TRANSL Field Options Screen

The Field Options screen enables you to specify field manipulation options for each record type in the output file. These options enable you to determine the order, length, and data type of each field in the output record, as well as to specify the insertion, omission, or reordering of individual fields.

The Field Options screen prompts you for three items of information (plus an optional fourth) about each field in the record. The fields in which you specify these items are arranged in eight rows of four, each row dealing with one field in the record. (The fields in each row share the same numeric suffix, indicated by *xx* in the list below.)

If you use all eight rows, TRANSL displays another Field Options screen, and, when this is filled, another, until you either reach the maximum of 25 fields or leave at least one row blank to indicate that there are no more fields to specify.

Specify the fields in the order you want them to have in the output record. This need not be the same order they have in the input record. (See Section 17.5.1.) Enter information in the fields of the Field Options screen as follows:

POSTxx -- Specify the byte position of the field *in the input record*. Its position in the output record is determined by the order in which fields are specified. For example, if the field that begins at byte 11 in the input record is to be the first field in the output record, POST01 should have a value of 11. (Count the first byte in the record as 1, not 0.) Multiple fields of the same data type, *except zoned fields*, can be grouped into a single field.

To add a new (blank) field at this point in the output record, leave the POSTxx field blank, specify the length and type of the field to be added, and enter I in the SWITCHxx field.

LENGTHxx -- Specify the length (in bytes) of the field that begins at the position specified in the POSTxx field, or the length of the blank field you want to insert in the output record.

Note: *If the input file is indexed, you have the option of changing the position of the primary key in the output file, as described in Section 17.5.5. However, you must specify its new position in the output file, which is calculated by adding 1 to the total of the lengths of the fields that precede it. This is best done while the Field Options screen is displayed, since it cannot be recalled after the Indexed File Key Options screen appears.*

TYPExx -- Specify the data type of the input field. Valid data type values are B (Binary), C (Character), P (Packed), or Z (Zoned decimal). *Binary fields are not translated.* When Zoned decimal characters are translated, the sign is included in the low-order byte of the field.

SWITCHxx -- This field is used for (1) inserting new fields in the output record, or (2) deleting input fields from the output record. An inserted field appears only in the output record; a deleted field appears only in the input record. *Do not specify a SWITCHxx value for a field that is to appear in both the input and the output records.* The default value for this field is blank.

I (Insert) -- Insert a new field (of the length specified in LENGTHxx) in the output record. The value of POSTxx must be blank or 0 (indicating that this field has no position in the input record). Since the input record contains no data corresponding to this field, it will be filled as follows:

-- Character fields are filled with blanks (X'20).
-- Binary fields are filled with nulls (X'00').
-- Packed or zoned fields are given a value of decimal 0.

D (Delete) -- Do not transfer this field from the input to the output record. *This option is significant only for fixed-length records and does not apply to variable-length records.* If you do not account for all bytes of a fixed-length input record, then the input and output record lengths differ and an error message screen appears. From this error message screen, press PF1 to redefine the field attributes. You can also press PF16 to terminate TRANSL processing.

If the input file has fixed-length records, your specifications from the Field Options screen must account for every byte of the input record. Otherwise, TRANSL detects a difference between the lengths of the input and output records and displays an error message. (From the screen that displays this message, you can either press PF1 to return to the Field Options screen and revise your specifications, or press PF16 to terminate TRANSL processing.)

Since all fields in a fixed-length input record must be specified, you cannot omit a field from the output record simply by leaving it out of the specifications. However, the SWITCHxx field provides a means of preventing it from going into the output record. See Section 17.5.3 for more information.

Sections 17.5.1 through 17.5.4 describe in detail the ways you can manipulate fields by means of your specifications from the Field Options screen. When you fill the screen or finish specifying fields, press ENTER.

- If you have filled all eight sets of fields on the screen, a new Field Options screen appears. Continue if you have more specifications to enter. When you press ENTER from a Field Options screen that contains less than eight rows of field specifications, the process is complete.

- If you indicated that the input file contains multiple record types, TRANSL displays a new Record Types screen (Figure 17-7) as soon as you finish specifying field manipulation options. If you have more record types to define, proceed as described in Section 17.4. After defining a record type, you are prompted to specify field manipulation options for records of that type, then to define the next record type, and so on. When all record types are defined, press ENTER from an empty Record Types screen to conclude the cycle of defining record types and specifying field manipulation options.

- If your input file is indexed, the Indexed File Options screen (Figure 17-9) appears after the last field manipulation options are specified, as described in Section 17.5.5. Otherwise, the Output File screen (Figure 17-10) appears, as described in Section 17.6.

17.5.1 Changing Field Order

Your specifications from the Field Options screen (Figure 17-8) can be used to change the order of fields as they are transferred from the input to the output record. This rearrangement is accomplished by manipulating two kinds of information:

The value in the POSTxx field -- This value always reflects the position of the field *in the input record.*

The order in which the output fields are specified -- This order always reflects the position of the field *in the output record.*

For example, suppose the first two fields in the input record begin at bytes 1 and 20, and you want them to change places in the output record. Enter 20 in POST01 and 1 in POST02 (and specify the lengths of both fields accurately in the LENGTH01 and LENGTH02 fields). TRANSL first reads the data beginning at byte 20 in the input record into Field 1 of the output record, then reads the data beginning at byte 1 into Field 2.

Although you do not generally have to be aware of the exact byte position a field will occupy in the output file, an exception must be made if the primary key field of an indexed file is being changed or moved. See the description of the Length field and Section 17.5.5.

17.5.2 Changing Field Length

While transferring a field from the input to the output record, you can change its length in any of the following ways:

Truncating a field -- You can shorten any field in the input record by simply specifying a shorter length in the LENGTHxx field. For example, a character field for a 9-digit zip code could be transferred to the output record as a field that contains only the first five digits by specifying the length as 5 instead of 9. TRANSL reads five bytes of data from each input record (starting at the first byte in the field, as specified in POSTxx) and thus transfers only the first five digits of each zip code to the output file.

Combining fields -- Fields of the same data type that are contiguous in the input file can be combined in the output file by specifying a length equal to the combined lengths of the contiguous fields. Take, for example, an input file containing the following character fields in succession:

Number and street (30 bytes)
City (15 bytes)
State (2 bytes)
Zip code (5 bytes)

These four fields can be transferred to the output record as a single address field. Its position (POSTxx) is specified as the first byte of the first field, and its length (LENGTHxx) as the total length of all four fields (52 bytes). TRANSL reads the 52 bytes of information starting at POSTxx into the output record as a single field.

Multiple contiguous fields can be combined in this way only if they are all of the same data type. However, *fields whose type is Z (Zoned) cannot be combined even when they are contiguous.*

Note: *Be careful that fields are not combined in such a way as to exceed the maximum length for the data type. See "Maximum Field Lengths," below.*

Extending Fields -- Although you cannot specify a greater length for a field in the output record than it has in the input record, you can provide for the addition of more data to the field by inserting an empty field after it in the output record. Suppose, for example, that your input file has a 5-byte field for zip codes, but you would like the output file to accommodate 9-digit zip codes. You could accomplish this purpose by first specifying the zip code field exactly as it is, including a length of 5 bytes in the LENGTHxx field, then by using the method described in "Inserting Fields," below, to insert a blank field four bytes long as the next field in the output record. You can then define the full nine bytes as one field in the output record, and enter 9-digit zip codes in it without fear of overwriting other data.

Note: *Be careful not to extend a field to the point where it exceeds the maximum length for the data type. See "Maximum Field Lengths," below.*

Inserting Fields

The SWITCHxx field of the Field Options screen (Figure 17-8) provides a way to insert blank fields (in which data can later be entered), into the output record. You can also use a blank field to extend the length of an input field so that it can accommodate more bytes of data, as described in the previous section.

When you want to insert a field, specify it at the point in the sequence of fields where you want it to appear in the output record. (Remember that the order in which fields are specified from the Field Options screen should reflect the desired output record, not the actual input record.) Specify the fields as follows:

POSTxx -- Enter 0 in this field, since the field you are inserting has no position in the input file.

LENGTHxx/TYPExx -- Specify the length and type you want the inserted field to have in the output record.

SWITCHxx -- Enter I (Insert) in this field.

For example, to insert a blank field that extends from byte 21 through byte 30 in the output record, first specify the field or fields that occupy Bytes 1 through 20. In the next row of fields, enter 0 in the POSTxx field, 10 in the LENGTHxx field, and I in the SWITCHxx field. (If TYPExx = C, a field of blanks is inserted; if TYPExx = B, a field with a value of Hex '00' is inserted. If TYPExx = P or Z, a field with a value of decimal 0 is inserted.)

Maximum Field Lengths

If you intend to extend the length of a field by any of the methods described in this section, be careful that the result does not exceed the maximum permissible length for a field of its data type. This problem is especially likely to arise in connection with noncharacter fields, as the following list shows:

Data Type	Maximum Length
Binary (B)	4 bytes
Packed (P)	8 bytes
Zoned (Z)	15 bytes

Character fields are less problematic, as their maximum length is limited only by the length of the record. The maximum permissible length for a record depends on file type and organization, as shown in Table 17-1.

Table 17-1. Maximum Record Lengths (in Bytes) by File Type and Organization

File Organization	File Type		
	Fixed	Variable	Compressed
Consecutive	2048	2024	2024
Indexed	2040	2024	2024
Relative	2040	2040	----[a]

[a] Records in relative files cannot be compressed.

17.5.3 Omitting Fields From the Output Record

If you do not want a particular field transferred to the output record, you can direct TRANSL to omit it. The omission of this field will depend on whether the length of the records in the input file is fixed or variable.

If the input record length is variable, you can simply omit the fields you do not want transferred from your specifications. Only the fields you specify will appear in the output record.

If the input record length is fixed, TRANSL expects your specifications to account for the entire record, and displays an error message if even a single byte is left out. You cannot, therefore, omit a field from the output record by not specifying it from the Field Options screen.

However, you can "delete" such a field from the output record by entering D in the SWITCHxx field to indicate that you do not want it placed in the output record. The field is not literally deleted, since the input record remains unchanged, but since TRANSL does not place it in the output record, the effect is similar to a deletion as far as the output file is concerned.

17.5.4 Specifying the Primary Key for an Indexed File

If your input file is indexed, the Indexed File Options screen (Figure 17-9) appears at the end of the process of specifying field manipulation options.

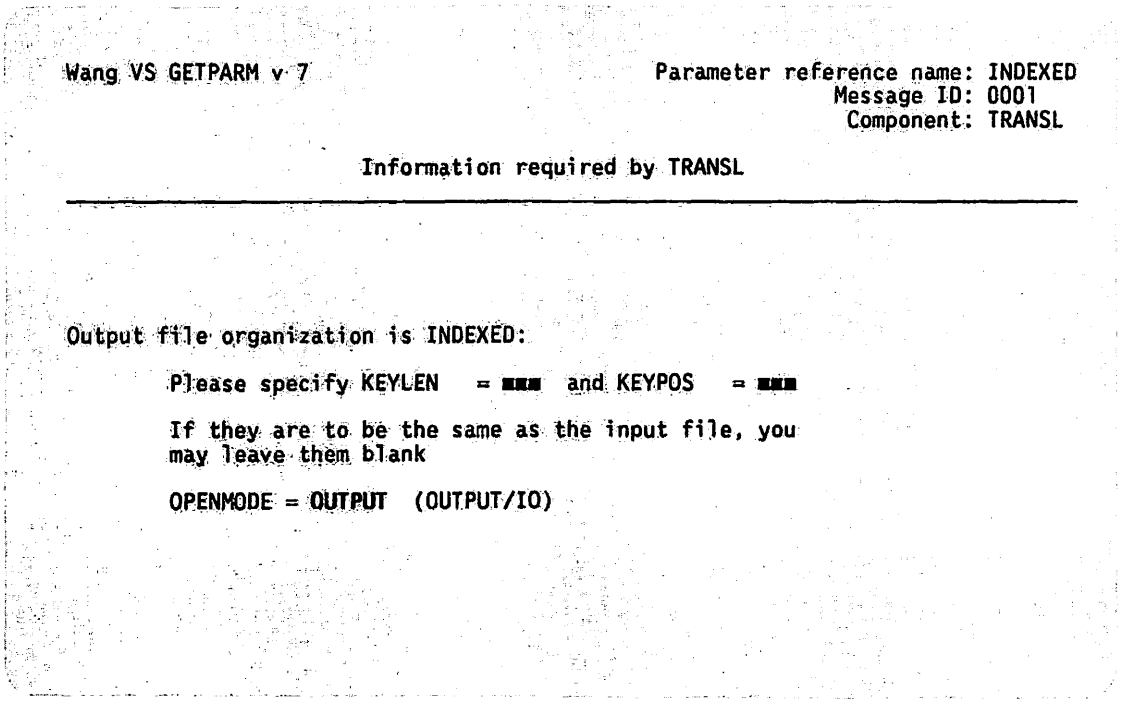


Figure 17-9. TRANSL Indexed File Options Screen

This screen prompts you to specify a length (KEYLEN) and position (KEYPOS) for the primary index key and an open mode (OPENMODE) for the output file.

If you want the output file to have the same primary key as the input file, *and if that key is in the same position in the output record,* you do not have to specify the length and position. Leave the KEYLEN (Key Length) and KEYPOS (Key Position) fields blank. If you have moved the key field to a different position by rearranging the record, however, it is your responsibility to calculate this new position and specify it from this screen. *TRANSL does not keep track of a repositioned primary key field. Failure to specify the position and length of a key field that has been moved can have unpredictable results.*

If the primary key field is to occupy a position in the output file different from its position in the key field, or if you want to use a different field as the primary key, enter the length of the field in the Key Length field of the Indexed File Options screen, and the position of the first byte in the Key Position field. (It is up to you to calculate this position. You can do so easily by summing the lengths of all fields that precede it and adding 1 to the result, but you cannot redisplay the Field Options screen at this point, so it is best to do the calculation before you exit from that screen.)

If you specify a field for the primary key of the output file that was not the primary key of the output file, it is possible that the output file may contain duplicate keys. When TRANSL encounters one of these, it displays a warning message and does not transfer the record that contains the duplicate key to the output file. (Duplicate keys can also occur if you truncate the original primary key field by specifying a shorter length than it had in the input file.)

The Open Mode field gives you two options for the mode in which TRANSL opens the output file while transferring data to it. OUTPUT, the default, is much faster, and should always be used if you know that all records are in primary key order. However, if you have changed the primary key field, it is possible that some records may not be in order. In such cases, you should use Input/Output mode (specified by entering IO -- *not* I/O -- in the Open Mode field. This mode is slower, but it permits TRANSL to sort the records into primary key order and insures that they all get into the file.

If TRANSL encounters out-of-order key values when the output file is open in Output mode, it treats them in the same way it treats duplicate keys -- it displays a warning message and omits the records whose key values are out of order from the file.

When you finish entering specifications and press ENTER, the Output File screen (Figure 17-10) appears, as described in Section 17.6.

17.6 SPECIFYING THE OUTPUT FILE

When all translation tables, record type definitions, and field manipulation options have been specified, TRANSL does the translation and/or reformatting you have requested. When TRANSL is ready to deposit the output records in a file, the Output File screen (Figure 17-10) appears.

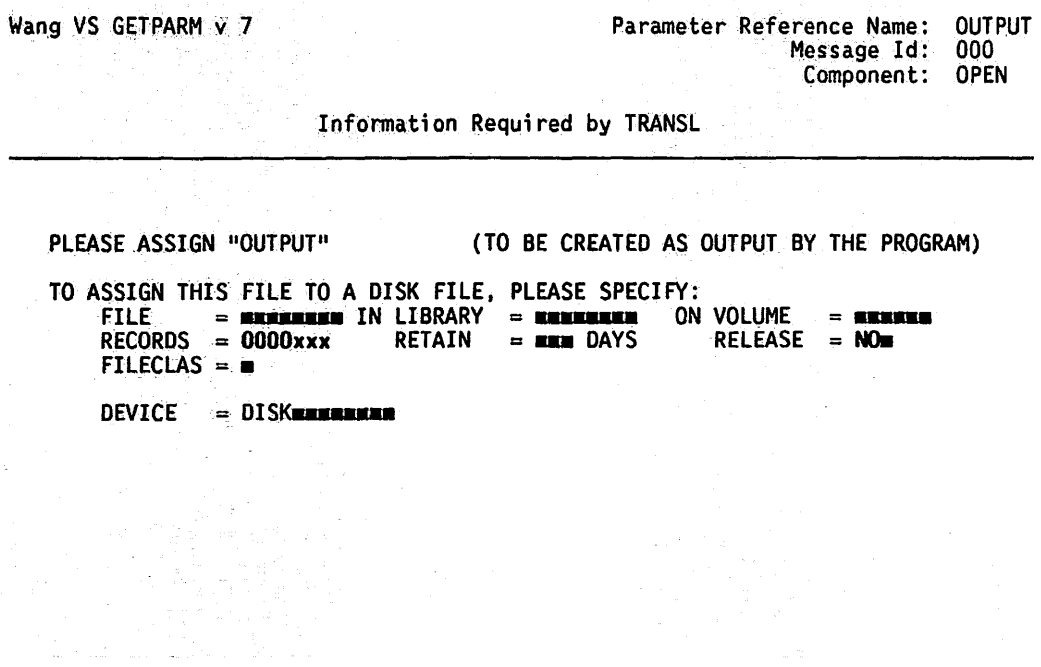


Figure 17-10. TRANSL Output File Screen

This screen prompts you to specify the name, location, and attributes of the output file in which the translated records will be placed. Enter information in the fields as follows:

FILE -- A valid name for the output file.

LIBRARY -- The library in which you want the output file to reside. The default value is taken from the OUTLIB field of your usage constants.

VOLUME -- The volume on which you want the output file to reside. The default value is taken from the OUTVOL field of your usage constants.

RECORDS -- The default value is the number of records obtained from the input file (represented by xxx in the illustration). If you plan to add more records to the output file, however, you can enter a larger number. Make the file large enough to accommodate as many records as you anticipate will be added in the future. (You should also make sure the value in the Release field is NO, so that the extra space you allocate is retained in the file.)

RETAIN -- The number of days the file should be protected from deletion. The maximum is 999 days; the minimum and default value is 0.

FILECLAS -- The protection class of the output file. The default is blank. For information about file protection classes, see the *VS System User's Introduction*.

RELEASE -- Specify YES or NO to indicate whether unused storage extents, previously allocated for data, can be released for use by other files. (Specify NO if you have allocated more space than the file requires and want the file to keep that space to accommodate future updates.) The default value for this field is identical with that of the input file.

DEVICE -- Specify the device on which the output file is to be located. The options are DISK (if TRANSL is to store the output file on disk or diskette) and PRINTER (if the output file is a print file which TRANSL is to print instead of storing. (The TRANSL utility does not copy to tape.) The default value is DISK.

When TRANSL processing is complete, the translated and reformatted records you specified are placed in the output file. The input file still exists in its original form. TRANSL displays its End of Job (EOJ) screen (not shown). From this screen you can either exit from TRANSL by pressing PF16, or rerun the utility by pressing PF1 to display a fresh Input File and Function screen (Figure 17-2).

17.7 A SAMPLE TRANSL PROCEDURE

TRANSL processing can be controlled through the VS Procedure Language. A procedure can specify the file to be translated, all TRANSL options (including a new translation table), and all details of output file organization. A complete list of TRANSL GETPARMs is provided in Appendix A. Consult the *VS Procedure Language Reference* for information about VS Procedure Language syntax.

The following sample procedure converts an indexed file with 120-byte records (containing only character fields) from the ASCII to the EBCDIC character code. The file contains only one record type. The procedure specifies the input file (DATASCII in Library TRANSIN) and the type of translation; it does not select output record reorganization. For the output, the procedure specifies the file DATEBCDI in Library TRANSOUT (doing so in a DISPLAY statement so that the user has the opportunity to redirect output to a different file or library). It then terminates the program.

```
PROCEDURE
  RUN TRANSL
    ENTER INPUT FILE=DATASCII, LIBRARY=TRANSIN, VOLUME=SYSTEM,
      CODE=EBCDIC
    ENTER OPTIONS POST01=0001, LENGTH01=0120, TYPE01=C
    ENTER INDEXED OPENMODE=OUTPUT
    DISPLAY OUTPUT FILE=DATEBCDI, LIBRARY=TRANSOUT, VOLUME=SYSTEM
    ENTER EOJ 16
RETURN
```

Part III

APPLICATION DEVELOPMENT UTILITIES

CHAPTER 18
COBGEN

18.1 INTRODUCTION

The COBGEN (COBOL Generation) utility lightens the COBOL programmer's burden by generating source code for the standard parts of a COBOL program. It is even possible, using COBGEN alone, to generate a complete program that will compile and run. However, COBGEN's product is intended primarily as a skeleton around which a program can be built. The default mode of operation is to enter the Editor automatically as soon as the source code is generated, so that you can begin augmenting it.

18.1.1 Overview

COBGEN produces at least skeletal code for all the standard parts of a COBOL program, including Identification, Environment, Data, and Procedure Divisions. It first prompts you to enter file information from which it can generate code for the appropriate sections of the program. Then it displays a menu from which you may choose any or all of the following:

- Figurative constants
- Dates
- GETPARMs
- PUTPARMs
- Linking to programs
- Sorting (using the internal COBOL sort verb)
- Sorting (using the external VS SORT utility via the SORTCALL VSSUB)

Depending on the choices you make from this menu, COBGEN presents a series of prompt screens (sometimes its own, sometimes those of other utility programs to which it links) and gathers the information necessary to generate the code for the functions you have chosen. At the conclusion of this process, COBGEN generates the source code file and (unless you direct otherwise) invokes the Editor so that you may add or modify whatever is necessary to complete your program.

18.1.2 Software Requirements

COBGEN interacts with several other programs and has the following software requirements:

- RESVFILE, which contains a list of COBOL reserved words for checking, must be in the same library as COBGEN.

- The VS CONTROL, EDITOR, EZFORMAT, and EZPRINT utilities, and the VS COBOL compiler must be in either the same library as COBGEN, or in the system library. To support the internal sort option, the COBOL compiler must be Release 3.08.04 or greater.

- The VSSUBS subroutine library should also be on the same volume as COBGEN, or on the system volume. (COBGEN sets the Editor defaults according to this expectation, but if the VSSUBS library is not found, a screen appears to prompt you for the proper location. This defect is therefore inconvenient rather than fatal, provided the VSSUBS are available on the system.)

18.1.3 Running COBGEN

Run the COBGEN program from the Command Processor as you run any other program or utility. Enter COBGEN into the program name field along with the appropriate library and volume names. (Since this utility does not reside in the system library, you must type the library and volume designations.)

The following sections describe COBGEN processing:

Section	Process
18.2	General Specifications
18.2.1	Specifying the Output Source File
18.2.2	Specifying File Control Information
18.2.3	Selecting Source Code Options
18.3	Option Specifications
18.3.1	Control File Specifications
18.3.2	Sort Specifications
18.3.3	PUTPARM Specifications
18.3.4	LINK Specifications
18.3.5	Screen Display Specifications
18.3.6	Report Format Specifications
18.3.7	Other Options
18.4	Source Code Generation
18.5	A Sample of COBGEN Code

18.2 GENERAL SPECIFICATIONS

18.2.1 Specifying the Output Source File

When COBGEN processing begins, the Source File Specification screen appears. Figure 18-1 shows a sample of this screen.

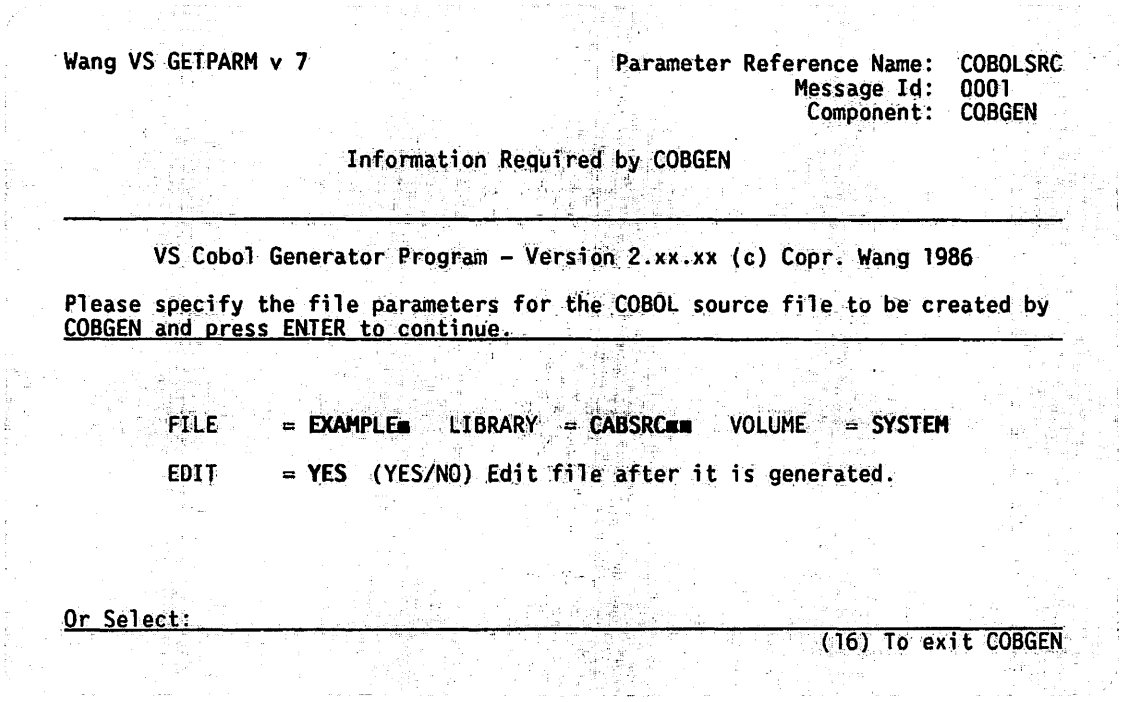


Figure 18-1. A Sample COBGEN Source File Specification Screen

Use this screen to supply file information for the COBOL source code file the program will generate. Enter information in the fields as follows:

FILE -- Supply a valid file name. It must not be a COBOL reserved word.

LIBRARY -- The library where you want the source file to reside. If the OUTLIB field of your usage constants is set, the value appears here as a default.

VOLUME -- The volume where you want the source file to reside. If the OUTVOL field of your usage constants is set, the value appears here as a default.

EDIT -- Change the default YES to NO if you do not want to enter the Editor to edit the source file when it is created.

Press ENTER when you have made your specifications. Alternatively, you can press PF16 to terminate COBGEN.

18.2.2 Specifying File Control Information

When you press ENTER from the Source File Specification screen, the File Control Information screen appears. Figure 18-2 shows a sample of this screen.

```
                    COBOL Source Code Creation Utility

 Program Id: EXAMPLE     Author: Charlie Bowen              Date: 09/24/87

 File Control Information    Name      Open Mode   Copylib    Record Key
                                                    (Y/N)
   Workstation File:         WSFILE      IO          N
   Printer File:             PRTFILE     O           N

   Disk Files:               PAYROLL     S           N
                             SUI         S           N
                             SORTPAY     IO          N
                             PERSONNL    S           Y     EMPLOYEE-NUMBER
                                         S           N
                                         S           N
                                         S           N
                                         S           N
                                         S           N

 * Open Mode :  I=Input, O=Output, IO=Input-Output, S=Shared, EX=Extend
 * Record Key:  Is Entered Only If 'Copylib = 'Y', and the File is Indexed

                    Press ENTER to Process, or PF16 to Exit
```

Figure 18-2. A Sample COBGEN File Control Information Screen

From this screen you can confirm or change program ID, author, and date information. If the program code is to generate screen displays or printed reports, specify the workstation or printer files. All data files referenced in the generated program code must also be specified from the File Control Information screen.

The Program ID and Author fields at the top of the screen show default values associated with the user, and the Date field shows the system date. All are modifiable.

Enter other information on this screen as follows:

NAME -- Enter the names of all files you intend to use for input and output. The program will not accept COBOL reserved words as file names. The workstation file must be named if you want COBGEN to generate one or more screen displays, and the printer file must be named if you want it to define one or more printed reports.

OPEN MODE -- Specify the mode in which each file is to be opened:

I Input
O Output
IO Input-Output
S Shared
EX Extend

(These values are fixed for the workstation and printer files, and can be specified only for disk files.)

COPYLIB -- If you keep COBOL file definitions in a library to be referenced by COPY statements at compile time, enter a Y in this column opposite each file name for which such a definition exists. COBGEN will access the source code in the library that describes this file and generate an appropriate SELECT statement and an FD statement that contains a COPY statement for the file layout information.

RECORD KEY -- If COPYLIB is set to Y *and* if the file in question is indexed, enter the record key in this column for use in the SELECT statement.

When you finish making specifications, press ENTER. (You can press PF16 instead to terminate COBGEN processing.)

18.2.3 Selecting Source Code Options

When you press ENTER from the File Control Information screen, the Source Creation Options menu appears. Figure 18-3 shows a sample of this menu.

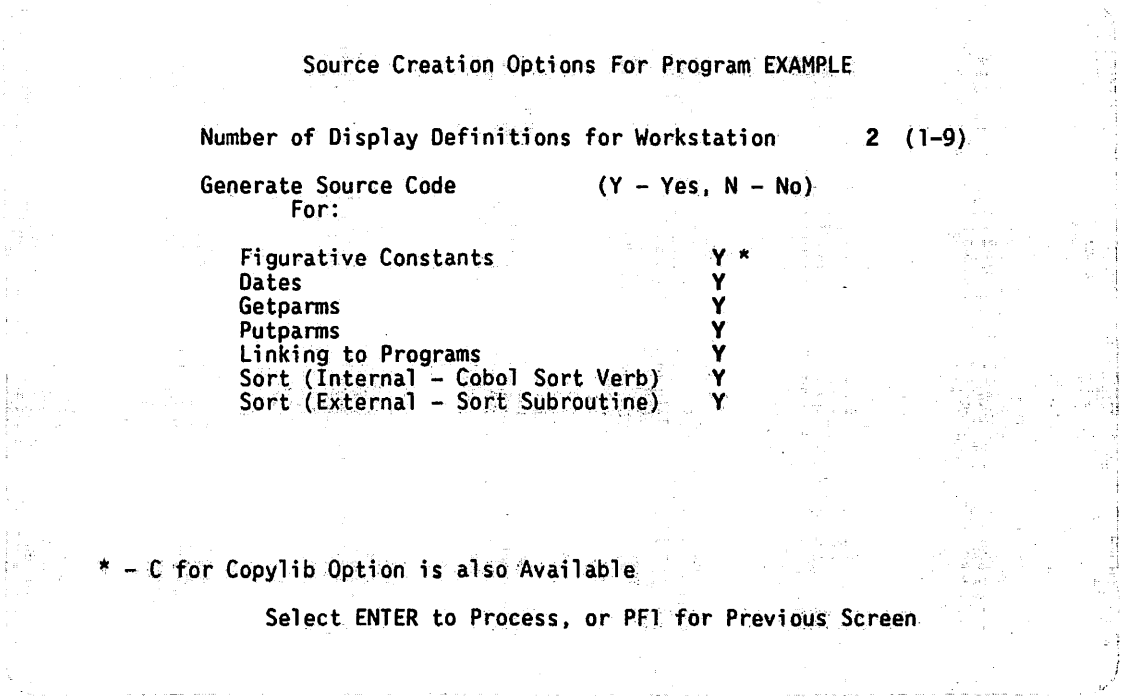


Figure 18-3. A Sample COBGEN Source Creation Options Menu

If you have named a workstation file on the File Control Information screen, you are prompted at the top of the Source Creation Options Menu screen for the number of screen displays you intend to define. (You must define at least one; the maximum is nine.)

Next, the screen lists seven code generation options. (The defaults you see depend in part on the specifications you have made from the File Control Information screen.) To request code generation for any item in the list, enter a Y; to prevent it, enter an N. You may combine any or all options from the list.

For the Figurative Constants option only, you can enter a C to indicate that you want the program to reference a FIGCON file in your copy library.

When you finish, press ENTER.

Alternatively, you can press PF1 to return to the File Control Information screen if you want to change the specifications you entered there. When you have done so, the Source Creation Options menu reappears so that you can complete the selection of options.

18.3 OPTION SPECIFICATIONS

When you press ENTER from the Source Creation Options menu, you have finished listing your requirements for the source file that is to be generated, and COBGEN begins to assemble the information it needs in order to fulfill them. It solicits information from a number of its own screens, and also links to various VS utilities which solicit information from their own screens. The specifications you enter from the File Control Information screen and the options you choose from the Source Creation Options menu determine which screens appear and what information you are asked to supply.

The following sections describe the various menu options in the order in which the related screens appear. The list below shows which sections describe the menu options and other specifications you may need to provide:

Figurative Constants	18.3.7
Dates	18.3.7
GETPARMs	18.3.7
PUTPARMs	18.3.3
Linking to Programs	18.3.4
Sort (both types)	18.3.2
Display Definitions	18.3.5
Print Format Definitions	18.3.6
Control File Specifications	18.3.1

Section 18.5 contains a sample of source code produced by COBGEN, marked to show which option caused each section of code to be generated.

18.3.1 Control File Specifications

If, from the File Control Information screen, you specified data files for which no copy library exists, the next screen to appear is a Control File Specification screen. Figure 18-4 shows a sample of this screen.

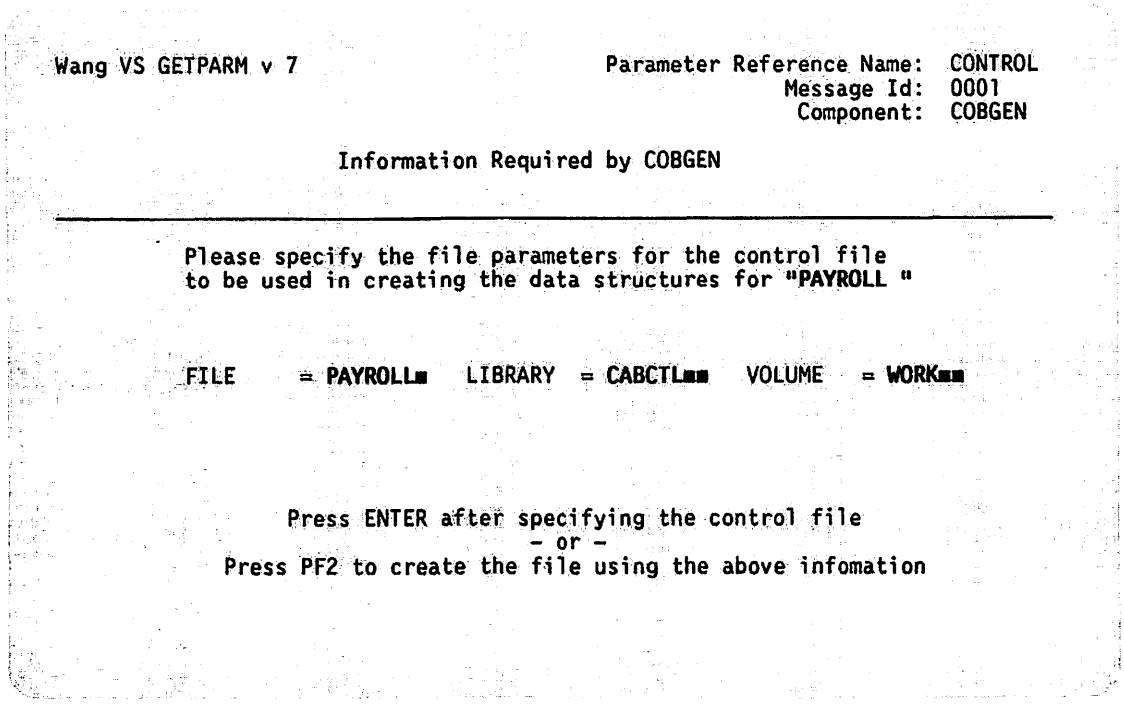


Figure 18-4. A Sample COBGEN Control File Specification Screen

COBGEN displays a Control File Specification screen for each data file you specified without a copy library from the File Control Information screen. If the program cannot access a COBOL source code description of the data file, it must obtain the necessary description of the file structure from a control file. (Chapter 2 describes control files and the way they are created through the CONTROL utility.)

The name of the data file to which this Control File Specification screen applies appears at the top of the screen. Specify the corresponding control file in the three fields below:

FILE -- Enter the name of the control file that describes the structure of this data file. If the control file does not exist, you must run the CONTROL utility to create it; you can do so by pressing PF2 from this screen. The default value provided is the name of the data file to which this screen applies.

LIBRARY -- COBGEN creates a default value by concatenating your user ID with the string *CTL*, the same default used by CONTROL.

VOLUME -- If the INVOL field of your usage constants is set, the value appears here as the default.

Enter or change the information in these fields as appropriate, then do one of the following:

- Press ENTER if the control file already exists. COBGEN links to the CONTROL utility and creates a COBOL file description from the information contained in this file. The description is used in the process of generating code.

- Press PF2 if the control file must be created. COBGEN links to the CONTROL utility (described in Chapter 2), which you must use to create a control file. When you have done this, CONTROL generates a COBOL file description for COBGEN to use.

Note: *It is your responsibility to make sure that none of the names supplied to the CONTROL utility for use in the control file (for instance, names of fields, files, or libraries) are COBOL reserved words. COBGEN cannot generate source code for the control file unless you observe this restriction.*

If another data file has been specified without a copy library, a new Control File Specification screen, with the name of that file at the top and in the FILE field, appears as soon as you have specified (and if necessary created) a control file from this one. This cycle is repeated until control files have been created or identified for all data files whose definitions cannot be referenced in a copy library.

18.3.2 Sort Specifications

If you chose either sort option, internal or external, from the Source Creation Options menu, the next screens that appear request information to define the nature of the sort. The screens differ according to the type of sort.

Note: *The advantage of using the internal COBOL SORT verb rather than the external SORT utility is that the former permits the use of input and output procedures to manipulate data before and after the sort. (COBGEN generates only the skeleton for these procedures, however; the programmer must supply the substance.) The external SORT utility lacks this feature. The external sort option, on the other hand, is capable of generating code for a "sort in place"; i.e., the input and output files can be the same. It can also reference files that have not been specified from the File Control Information screen. The internal sort option can do neither of these things.*

Internal Sort

If you specified an internal sort by means of the COBOL SORT verb, the Sort Input File Selection screen (not shown) appears. This screen displays the names of all the data files you have specified, and requests you to select the file to be sorted by placing the cursor next to the appropriate file name and pressing ENTER.

If you decide to bypass the internal sort option, you can press PF16 instead of ENTER. COBGEN will go on to process the other options you have chosen and will generate no code for an internal sort.

When you press ENTER from the Sort Input File Selection screen, the Sort Field Selection screen appears. Figure 18-5 shows a sample of this screen.

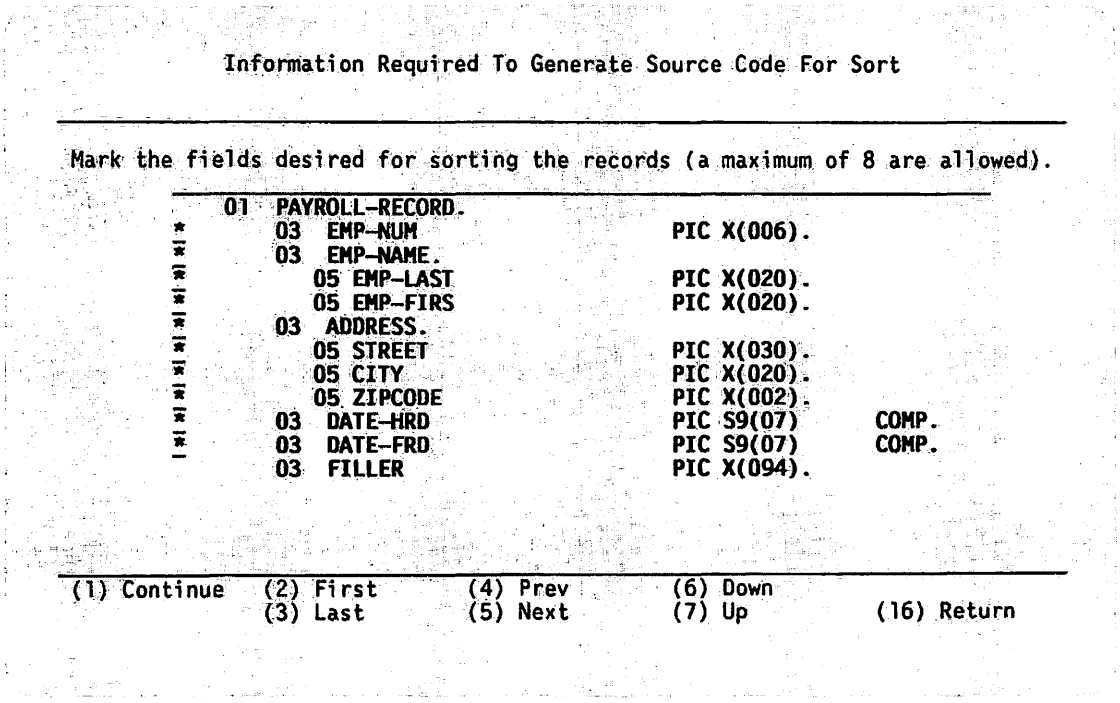


Figure 18-5. A Sample COBGEN Sort Field Selection Screen

This screen displays the field structure of the records in the file you specified from the previous screen. If the record contains too many fields to be displayed on one screen, you can move from screen to screen using the following PF keys:

PF2 Move back to the first screen
PF3 Move forward to the last screen
PF4 Move back to the previous screen
PF5 Move forward to the next screen

Mark the fields to be used in sorting the records by entering a character at the pseudoblank to the left of each. Any nonblank character selects a field, but if you enter numbers you can determine the order in which the fields are listed on the Sort Sequence screen that follows.

You can select as many as eight fields for the sort. When you have marked the sort fields and are ready to continue, press PF1. Alternatively, you can return to the Sort Input File Selection screen by pressing PF16.

After you select the sort fields and press ENTER, the Sort Sequence screen appears. Figure 18-6 shows an example of this screen.

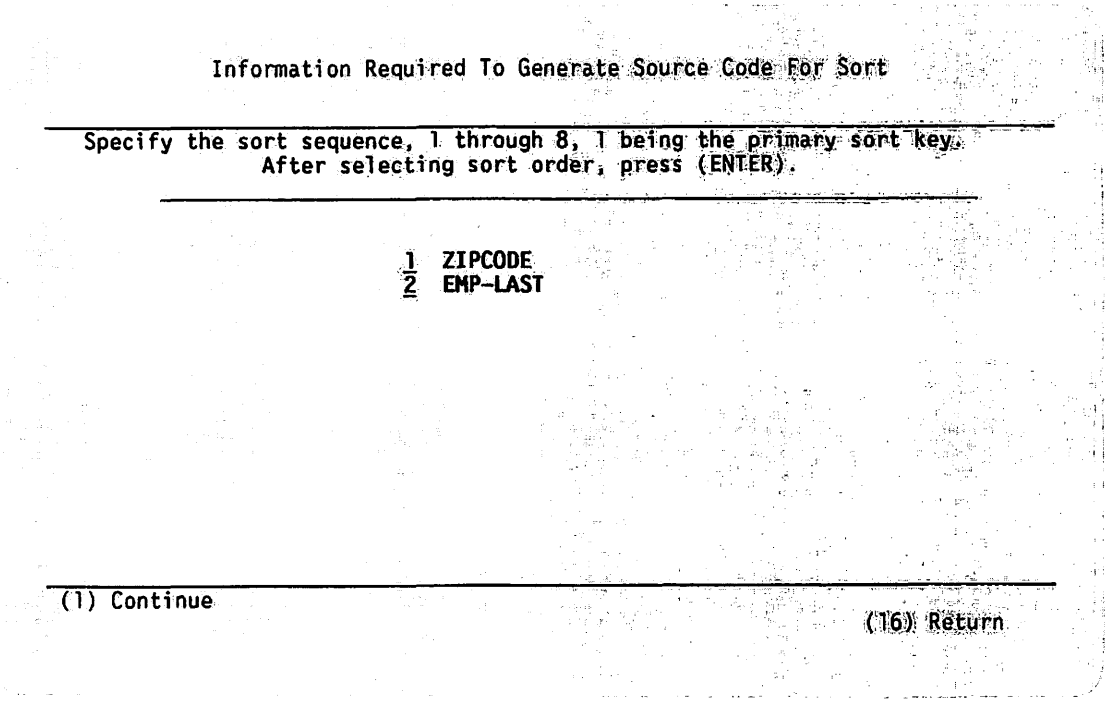


Figure 18-6. A Sample COBGEN Sort Sequence Screen

This screen displays the fields you selected. If you used numbers to mark your selections, the fields are listed in that numerical order. Otherwise, they are listed and numbered as they occur in the record.

You can rearrange the order at will, since the purpose of this screen is to establish the sequence that controls the sort. Simply enter new numbers in place of the numbers displayed, and press ENTER. COBGEN rearranges the display to correspond to the new sequence. If you want further rearrangement, enter new numbers as appropriate, and press ENTER again. You can repeat this process until you are fully satisfied with the sequence. Then press PF1 to continue.

If you want to change the set of fields you are manipulating, you can press PF16 instead to return to the Sort Field Selection screen.

When you have defined the sort sequence and pressed PF1 from the Sort Sequence screen, the Sort Direction screen appears. Figure 18-7 shows a sample of this screen.

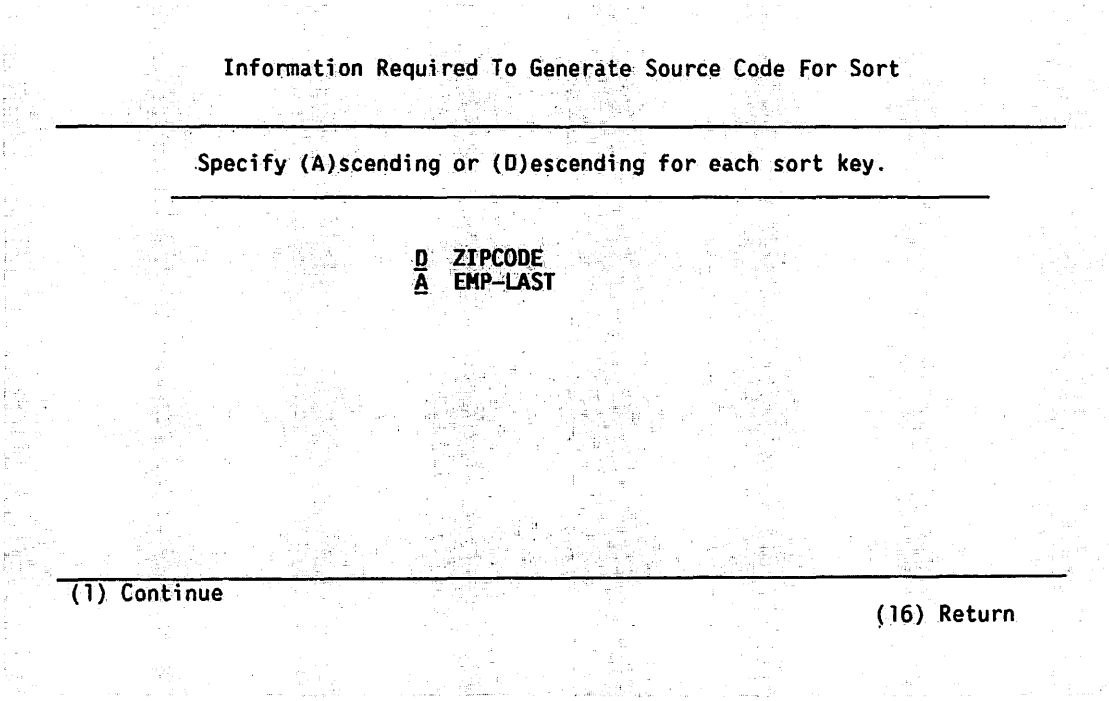


Figure 18-7. A Sample COBGEN Sort Direction Screen

This screen displays the names of the sort fields in the sequence you defined, and prompts you to enter an A or a D to the left of each name in order to determine whether that field is to be sorted in ascending or descending order. After you determine the sort direction for each field in this manner, press PF1 to continue. Pressing PF16 instead returns you to the Sort Sequence screen.

As soon as you press PF1, the Sort Output File Selection screen (not shown) appears. Like the Sort Input File Selection screen, it displays all the data files you have specified. This time you are prompted to place the cursor next to the output file and press ENTER. (Pressing PF16 instead returns you to the Sort Direction screen.)

When the output file is identified, no further specifications are required for an internal sort.

COBGEN generates code for only one internal sort. If you need more than one in your program, you must use the Editor to duplicate the code and modify it appropriately.

External Sort

If you specified an external sort, the External Sort Specification screen appears. Figure 18-8 shows a sample of this screen.

```
                Information Required To Generate Source Code For Sort

                                  Input Sort File

          FILE = PERSONNL     LIBRARY = PAYROLL■    VOLUME = SYSTEM

                                  Output Sort File

          FILE = SRTPRSNL     LIBRARY = PAYWORK■    VOLUME = WORK■■

Key Attributes:
          Position(>0)          Length In Bytes   Sort Key Type    Sort Order(A,D)
Key 1:    Post1     = 50■■      Length1 = 5■■     Type1    = C     Order1    = A
Key 2:    Post2     = 225■      Length2 = 10■     Type2    = C     Order2    = A
Key 3:    Post3     = ■■■■      Length3 = ■■■     Type3    = C     Order3    = A
Key 4:    Post4     = ■■■■      Length4 = ■■■     Type4    = C     Order4    = A
Key 5:    Post5     = ■■■■      Length5 = ■■■     Type5    = C     Order5    = A
Key 6:    Post6     = ■■■■      Length6 = ■■■     Type6    = C     Order6    = A
Key 7:    Post7     = ■■■■      Length7 = ■■■     Type7    = C     Order7    = A
Key 8:    Post8     = ■■■■      Length8 = ■■■     Type8    = C     Order8    = A

   * Key Type:   B=Binary, C=Character, D=Decimal, F=Floating, P=Packed,
                 Z=Zoned Decimal, L=Zoned Decimal Sign Leading
   * Sort Order: A=Ascending, D=Descending
```

Figure 18-8. A Sample COBGEN External Sort Specification Screen

This screen requests specifications for the input and output files and the fields in the input file that are to be sorted. This information is passed to the VS SORT utility by means of the SORTCALL subroutine. It is helpful to know these (see Chapter 14 for SORT and the *VS VSSUBS Reference* for SORTCALL), but if you know how the input file is structured you can provide the necessary information without this specific knowledge.

In the six fields at the top of the screen, specify both input and output files by name, library, and volume.

The remainder of the External Sort Specification screen contains eight sets of fields in which to specify the sort keys (i.e., the fields by which the record is to be sorted). Eight is the maximum number you can specify.

Enter information in the fields as follows:

POSITION -- The byte position at which the field begins (a number between 0 and the length of the record).

LENGTH -- The number of bytes in the field.

TYPE -- The type of data in the field. Acceptable values are

B Binary data
C Character data
D External decimal data, sign trailing separate
F Floating point data
P Packed decimal data, sign trailing included
Z Zoned decimal data, sign trailing included
L Zoned decimal data, sign leading separate

ORDER -- The order in which the field is to be sorted. Acceptable values are

A Ascending
D Descending

This screen solicits all the information COBGEN requires to generate code for an external sort. When you finish supplying specifications, press ENTER. COBGEN moves on to request information or generate code for other options you have chosen.

If you decide to bypass the external sort option, you can press PF16 instead of ENTER. COBGEN will go on to process the other options you have chosen and will generate no code for an external sort.

18.3.3 PUTPARM Specifications

PUTPARMs enable a program to supply parameters to a GETPARM issued by another program. In order to use a PUTPARM, the program you are generating must issue a LINK to another program (see Section 18.3.4), which will then issue a GETPARM.

If you have chosen the PUTPARM option from the Options screen, the next screen to appear is the PUTPARM Type Specification screen. Figure 18-9 shows an example of this screen.

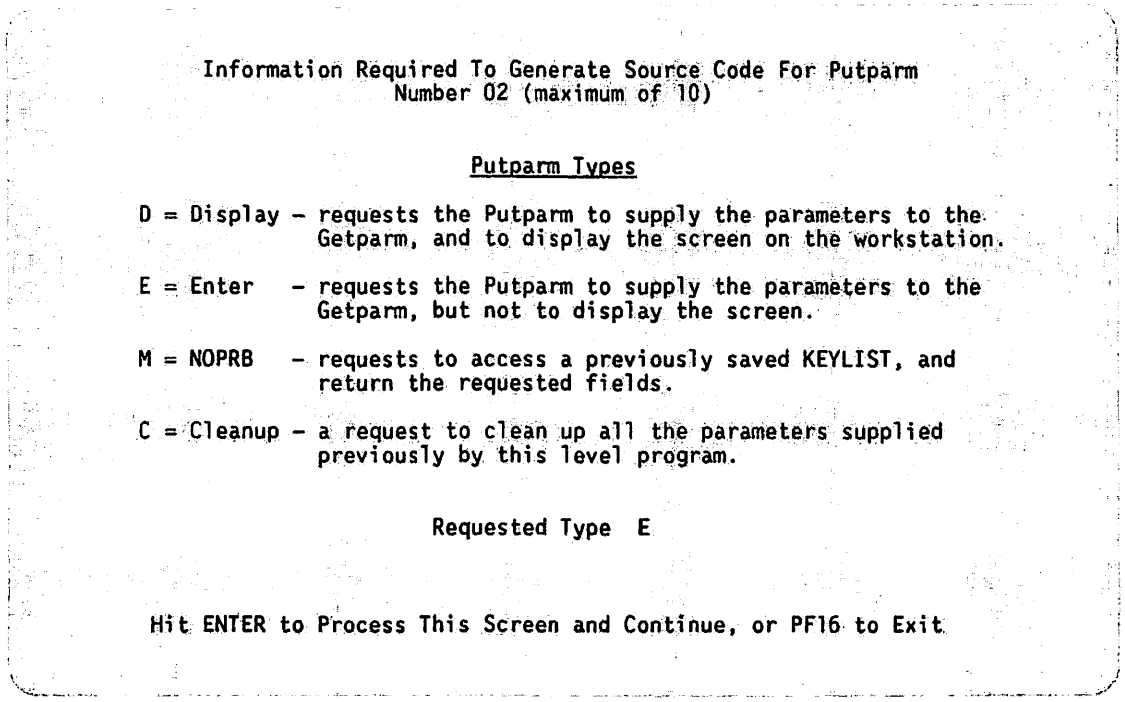


Figure 18-9. A Sample COBGEN PUTPARM Type Specification Screen

The message at the top of the screen shows the number of the PUTPARM for which COBGEN is currently requesting information. (The maximum is 5.) The screen explains the available PUTPARM type options. In the Requested Type field at the bottom, enter D, E, M, or C to specify a type for this PUTPARM. E (the type most frequently used) is the default value.

Note: *In order to supply the information requested from this screen and the next, you should be familiar with the use of the VSSUBS PUTPARM subroutine. See the* VS VSSUBS Reference *for information.*

When you have specified the type and are ready to provide parameter information, press ENTER.

If you do not want to enter PUTPARM specifications, press PF16 to continue. COBGEN goes on to request information or generate code for other options you have chosen and does not generate code for a PUTPARM.

If you press ENTER from the PUTPARM Type Specification screen, the PUTPARM Parameter screen appears. The screen contents differ according to the type of PUTPARM specified; Figure 18-10 shows a sample screen for a type E PUTPARM, which is the commonest type.

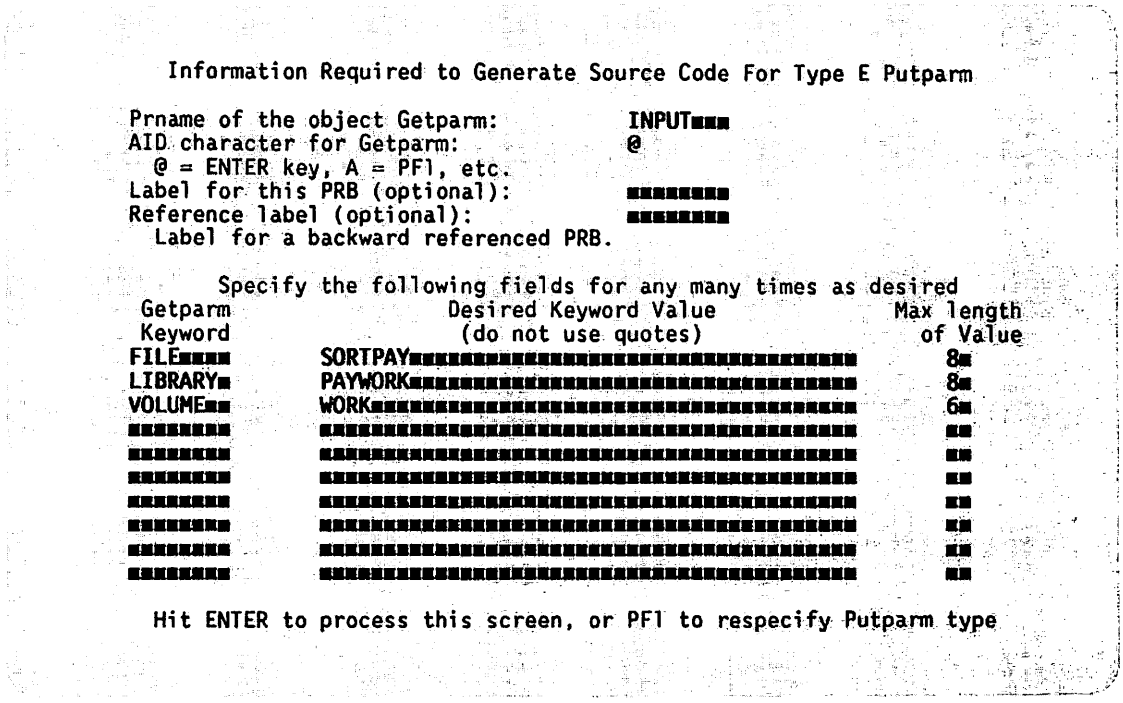


Figure 18-10. A Sample COBGEN PUTPARM Parameter Screen

To provide the necessary information for this screen, it is helpful to know the PUTPARM subroutine (see the note on the previous page). However, if you are familiar with GETPARM screens, it is not difficult to see what is required. The essential information consists of the following:

- The Parameter Reference Name (prname) for the GETPARM this PUTPARM is designed to satisfy.
- The Attention ID (AID) character that indicates which key should be "pressed" after all the GETPARM values are specified. See Table 18-1 for AID characters and values.
- A list of keywords corresponding to the GETPARM fields, with the value to be entered in each field and the maximum allowable length for each value.

If you want to change the type specification before entering parameters, press PF1 to return to the PUTPARM Type Specification screen. Otherwise, enter the parameters and press ENTER. The PUTPARM Type Specification screen reappears. You may either specify the type for the next PUTPARM, or press PF16 to move on.

Table 18-1. Attention ID (AID) Characters and Their Equivalents

AID Character	ASCII Character	AID Character	ASCII Character
ENTER key	@	PF17	a
PF1	A	PF18	b
PF2	B	PF19	c
PF3	C	PF20	d
PF4	D	PF21	e
PF5	E	PF22	f
PF6	F	PF23	g
PF7	G	PF24	h
PF8	H	PF25	i
PF9	I	PF26	j
PF10	J	PF27	k
PF11	K	PF28	l
PF12	L	PF29	m
PF13	M	PF30	n
PF14	N	PF31	o
PF15	O	PF32	p
PF16	P		

18.3.4 LINK Specifications

The VSSUBS LINK subroutine initiates the execution of another program or procedure from within the calling program. The program or procedure can specify any arguments needed to execute the linked program.

If you have chosen the LINK option from the options menu, the LINK Specification screen is the next to appear. Figure 18-11 shows a sample of that screen.

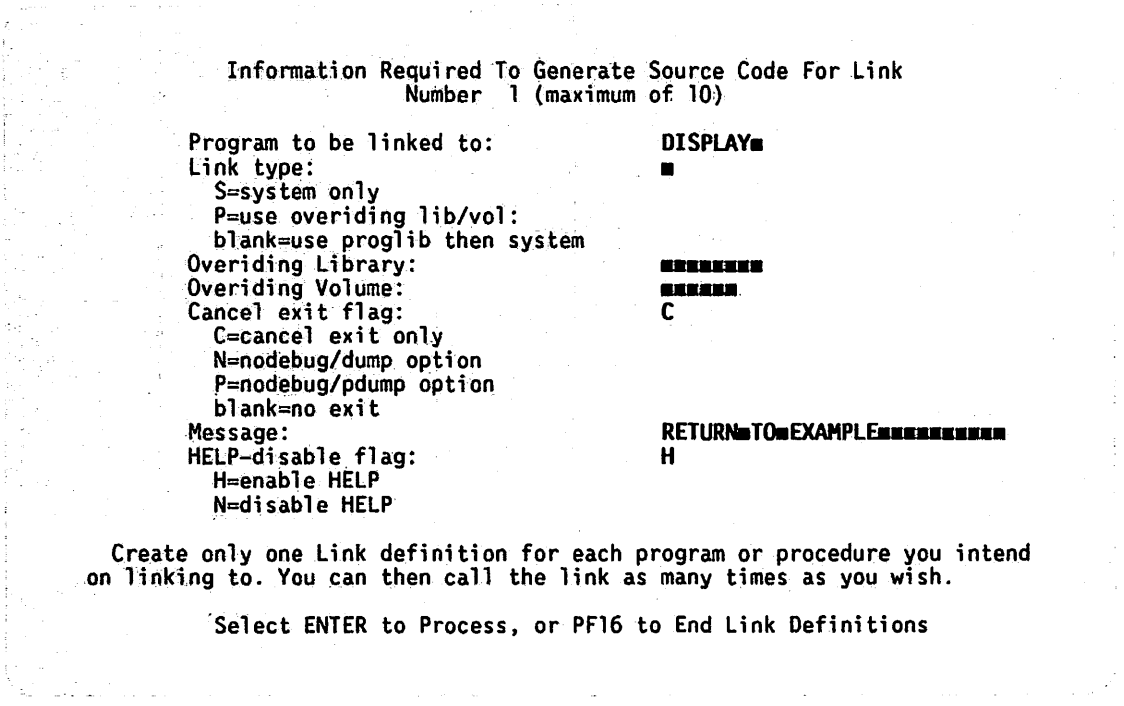


Figure 18-11. A Sample COBGEN LINK Specification Screen

The message at the top of the screen shows the number of the LINK for which the program is currently requesting information (the maximum is 10). You have to create only one LINK for each program you wish to link to; it can be reused at will.

Note: *To supply the information requested on this screen, you should be familiar with the LINK subroutine. It is described in the VS VSSUBS Reference.*

After you enter the information requested, either press ENTER to recall the Link Specification screen (if you wish to specify another LINK) or press PF16 to continue (if you have no more to specify).

18.3.5 Screen Display Specifications

If you have specified at least one screen display definition at the top of the Options menu, the Display Definition screen appears next. Figure 18-12 shows an example of this screen.

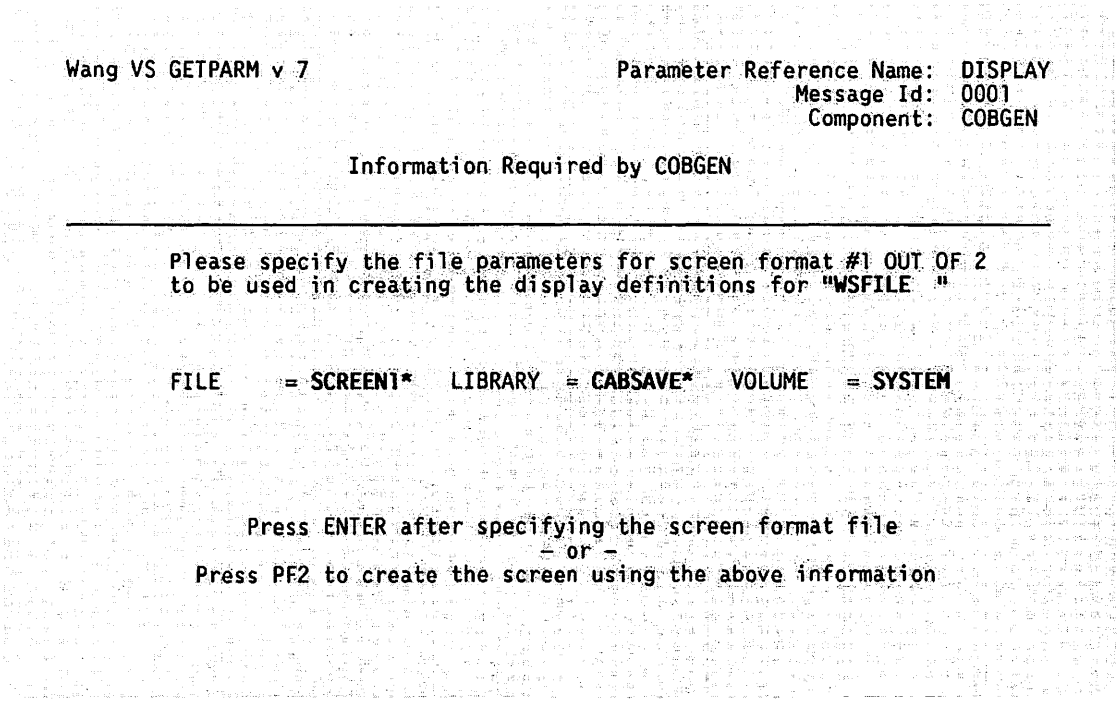


Figure 18-12. A Sample COBGEN Display Definition Screen

From this screen you can either specify an existing screen format file that defines the display in question, or link to the EZFORMAT utility to create a new screen format file. Enter information in the fields as follows:

FILE -- The name of the existing screen format file, or a valid file name for the screen format file you intend to create.

LIBRARY -- The library where the screen format file resides or will reside. COBGEN provides the same default value used by EZFORMAT: your user ID concatenated with the string *SAVE* (e.g., USRSAVE).

VOLUME -- If the INVOL field of your usage constants is set, the value appears here as a default.

When you have entered or changed the information in the fields as appropriate, press ENTER to specify an existing file, or PF2 to create a new one.

If you are creating a new file, COBGEN links to the EZFORMAT utility. Using this program, you can map out a screen display and have EZFORMAT generate the COBOL source code that enables your program to produce it. See Chapter 4 for information on the operation of EZFORMAT.

When you have either designated an existing file or created a new one through EZFORMAT, if any display definitions remain unspecified, the Display Definition screen reappears to solicit specifications for the next one. This process is repeated until the full number of display definitions you entered from the Options menu is accounted for.

18.3.6 Report Format Specifications

If you have specified a printer file from the File Control Information screen, the Print Format Definition screen (not shown) appears. Its layout is similar to that of the Display Definition screen (Figure 18-11), and like that screen it allows you the choice of designating an existing print format definition file or creating a new one. The default shown in the LIBRARY field (your user ID concatenated with the string *PRT* -- e.g., USRRPT) is the same one used by the REPORT utility.

Enter the file name (and the library and volume if you do not wish to keep the defaults). Then press ENTER to designate an existing file, or PF2 to create a new one.

If you press PF2 to create a new print format definition file, COBGEN links to the EZPRINT utility, which is used to define the report layout and to generate COBOL source code for producing it. See Chapter 20 of this manual for information on the operation of EZPRINT.

18.3.7 Other Options

Three options on the Source Creation Options menu do not require COBGEN to solicit any information. If you select any of these, the corresponding sections of source code are generated with no further action on your part.

Figurative Constants -- If you set this option to C (Copy Library), COBGEN generates code that references FIGCON in the default copy library. (If necessary, you can change this file specification in the editing process.) If you set the option to Y instead of C, COBGEN generates a set of figurative constants for various screen characteristics, cursor control, and the workstation alarm. (See the source code listing in Section 18.5.) The default for this option is Y if you have specified a workstation file; otherwise, it is N and no figurative constants are generated.

Dates -- If the value for this option is Y, COBGEN creates working storage and procedure code for getting the date from the system and printing it in a report. You can also adapt this code for other purposes. The default is Y if you have specified *either* a workstation or a printer file; otherwise it is N.

GETPARMs -- If this option is set to Y, COBGEN generates code for a "generic GETPARM," which must be edited to the user's requirements. (This code, though generated only once, can be duplicated and modified in the Editor.)

18.4 SOURCE CODE GENERATION

COBGEN now pulls together the information it has gathered and creates the source code file. If you have left the EDIT field of the Source File Specification screen (Figure 18-1) set to YES, the next screen you see is the Editor function menu. The newly created source code file is ready to be displayed and modified. When you exit from the Editor, COBGEN processing is complete, and you are returned to the Command Processor menu or to the program or procedure from which COBGEN was called.

If you specified NO in the EDIT field of the Source File Specification screen, you are returned to the Command Processor menu (or to the program or procedure from which COBGEN was called) as soon as the process of code generation is finished.

18.5 A SAMPLE OF COBGEN SOURCE CODE

The code shown on the following pages was produced directly by COBGEN, based on the specifications shown on the sample screens that illustrate this chapter. The code is unmodified except that, for the sake of clarity, line numbers and some repetitive sections are omitted. All gaps are clearly marked with asterisks and notes showing the number of omitted lines. Except for these explicit omissions, the entire source code file is reproduced.

The marginal notes to the right of the listing point out connections between particular sections of the code and the COBGEN options and specifications that caused the program to generate them.

```
*****************************************************************
*                                                               *
*                      IDENTIFICATION DIVISION                  *
*                                                               *
*****************************************************************
*
 IDENTIFICATION DIVISION.
 PROGRAM-ID.
     EXAMPLE.
 AUTHOR.
     Charlie Smith.
 DATE-WRITTEN.
     09/24/87.
*****************************************************************
*                                                               *
*                       ENVIRONMENT DIVISION                    *
*                                                               *
*****************************************************************
*
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.
     WANG-VS.
 OBJECT-COMPUTER.
     WANG-VS.
 FIGURATIVE-CONSTANTS.
     HIMOD      IS "80",
     HIMODUL    IS "A0",
     HIMODUPR   IS "81",
     HIMODUPRUL IS "A1",
     HIMODNUM   IS "82",
     HIMODNUMUL IS "A2",
     HIPRO      IS "84",
     HIPROUL    IS "A4",
     HIPROUPR   IS "85",
     HIPROUPRUL IS "A5",
     HIPRONUM   IS "86",
     HIPRONUMUL IS "A6",
     LOMOD      IS "88",
     LOMODUL    IS "A8",
     LOMODUPR   IS "89",

                * * * * * * * * * * * *

     SUMODUPRUL IS "B9",
     SUMODNUM   IS "9A",
     SUMODNUMUL IS "BA",
     SUPRO      IS "9C",
     SUPROUL    IS "BC",
     SUPROUPR   IS "9D",
     SUPROUPRUL IS "BD",
     SUPRONUM   IS "9E",
     SUPRONUMUL IS "BE",
     SOLIDS     IS "0B",
     POS-CURSOR IS "A0",
     SOUNDALARM IS "E0".
```

Identification Division code is generated automatically from the source code file name and the system user ID and date.

The Figurative Constants option causes values for a complete set of field attribute characters, plus cursor position and workstation alarm, to be generated. For a list of these values, see the VS Principles of Operation or the VS COBOL Reference.

(24 lines omitted)

```
*****************************************************************
*                                                               *
*                        INPUT-OUTPUT SECTION                   *
*                                                               *
*****************************************************************
*
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
*
     SELECT WSFILE
         ASSIGN TO "WSFILE",   "DISPLAY",
         ORGANIZATION    IS SEQUENTIAL
         ACCESS MODE     IS RANDOM
         RELATIVE KEY    IS LINENUM
         PFKEY           IS PF-KEY
         CURSOR POSITION IS CURSOR-POS
         FILE STATUS     IS FILSTAT.
*
     SELECT PRTFILE
         ASSIGN TO "PRTFILE",  "PRINTER",
         ORGANIZATION    IS SEQUENTIAL
         ACCESS MODE     IS SEQUENTIAL
         FILE STATUS IS FILSTAT-PRTFILE.
*
     SELECT PAYROLL
         ASSIGN TO "PAYROLL",  "DISK",     NODISPLAY,
         ORGANIZATION    IS INDEXED
         ACCESS MODE     IS DYNAMIC
         RECORD KEY      IS U1-EMP-NUM
         ALTERNATE RECORD KEY 01 IS U1-EMP-NAME WITH DUPLICATES
         FILE STATUS IS FILSTAT-PAYROLL.
*
               * * * * * * * * * * * * *

     SELECT SORT01
         ASSIGN TO "PAYROLL"     "DISK".
```

SELECT statements are generated for all files specified from the File Control Information Screen, starting with the workstation and printer files. (See Section 18.3.2.)

(20 lines omitted)

A SELECT statement is also generated for the internal sort when this option is chosen from the Source Creation Options menu (Figure 18-3). See the Data and Procedure Divisions also.

```
*****************************************************************
*                                                               *
*                    DATA DIVISION - FILE SECTION               *
*                                                               *
*****************************************************************
*
 DATA DIVISION.
 FILE SECTION.
*
 FD  WSFILE
     LABEL RECORDS ARE OMITTED.
 01  CRTREC                          PIC X(1924).
*
 FD  PRTFILE
     LABEL RECORDS ARE OMITTED.
 01  PRTREC                          PIC  X(132).
*
 FD  PAYROLL                                                    The code generated here reflects the file
     LABEL RECORDS ARE STANDARD                                 structure specified in the associated copy or
     VALUE OF FILENAME IS FIL-FOR-PAYROLL                       control file.  The latter can be created by
              LIBRARY  IS LIB-FOR-PAYROLL                       means of the CONTROL utility.  It must
              VOLUME   IS VOL-FOR-PAYROLL.                      contain no COBOL reserved words.  (See
 01  PAYROLL-RECORD.                                           Sections 18.3.2 and 18.4.1.)
     03  U1-EMP-NUM                  PIC  X(006).
     03  U1-EMP-NAME.
        05 U1-EMP-LAST               PIC  X(020).
        05 U1-EMP-FIRS               PIC  X(020).
     03  U1-ADDRESS.
        05 U1-STREET                 PIC  X(030).
        05 U1-CITY                   PIC  X(020).
        05 U1-STATE                  PIC  X(002).
        05 U1-ZIPCODE                PIC  X(005).
     03  U1-DATE-HRD                 PIC S9(07)      COMP.
     03  U1-DATE-TRM                 PIC S9(07)      COMP.
     03  FILLER                      PIC  X(089).
*
                * * * * * * * * * * * * *                       (38 lines omitted)

 SD  SORT01
     DATA RECORD IS SORT01-RECORD.
 01  SORT01-RECORD.
     03  S01-EMP-NUM                 PIC  X(006).              A record structure is generated for the
     03  S01-EMP-NAME.                                         internal sort.  See the Environment and
        05 S01-EMP-LAST              PIC  X(020).              Procedure Divisions for other code related to
        05 S01-EMP-FIRS              PIC  X(020).              this option.
     03  S01-ADDRESS.
        05 S01-STREET                PIC  X(030).
        05 S01-CITY                  PIC  X(020).
        05 S01-STATE                 PIC  X(002).
        05 S01-ZIPCODE               PIC  X(005).
     03  S01-DATE-HRD                PIC S9(07)      COMP.
     03  S01-DATE-FRD                PIC S9(07)      COMP.
     03  FILLER                      PIC  X(089).
```

```
*****************************************************************
*                                                               *
*                        WORKING STORAGE                        *
*                                                               *
*****************************************************************
*
 WORKING-STORAGE SECTION.
 01  DATE-INFO.
     03  DATEIN                    PIC 9(06).
     03  DATEGROUP                 REDEFINES DATEIN.
         05  YEAR-DIGITS           PIC 9(02).
         05  MONTH-DIGITS          PIC 9(02).
         05  DAY-DIGITS            PIC 9(02).
     03  RPT-DATE.
         05  RPT-MONTH             PIC 9(02).
         05  FILLER                PIC X(01)  VALUE "/".
         05  RPT-DAY               PIC 9(02).
         05  FILLER                PIC X(01)  VALUE "/".
         05  RPT-YEAR              PIC 9(02).
*****************************************************************
*                   FILE CONTROL STATUS SWITCHES                *
*****************************************************************
 01  FILSTAT.
     03  ERR-FLAG                  PIC X(01).
     03  PFK-BYTE                  PIC X(01).
 01  FILSTAT-PRTFILE.
     03  ERR-FLAG-PRTFILE           PIC X(1)  VALUE "0".
         88 DUPLICATE-ON-PRTFILE       VALUE "2".
     03  FILLER                     PIC  X(1).
 01  FILSTAT-PAYROLL.
     03  ERR-FLAG-PAYROLL           PIC X(1)  VALUE "0".
         88 MORE-PAYROLL               VALUE "0".
         88 RECORD-FOUND-ON-PAYROLL    VALUE "0".
         88 NO-MORE-PAYROLL            VALUE "1".
         88 NO-RECORD-FOR-PAYROLL      VALUE "2".
         88 DUPLICATE-ON-PAYROLL       VALUE "2".
     03  FILLER                     PIC  X(1).

               *  *  *  *  *  *  *  *  *  *  *  *

*****************************************************************
*              EXTERNAL FILE LOCATION DEFINITIONS               *
*****************************************************************
 77  REC-COUNT                     PIC 9(4)  VALUE 500.
 77  FIL-FOR-PAYROLL               PIC X(8)  VALUE SPACES.
 77  LIB-FOR-PAYROLL               PIC X(8)  VALUE SPACES.
 77  VOL-FOR-PAYROLL               PIC X(6)  VALUE SPACES.
 77  FIL-FOR-SUI                   PIC X(8)  VALUE SPACES.
 77  LIB-FOR-SUI                   PIC X(8)  VALUE SPACES.
 77  VOL-FOR-SUI                   PIC X(6)  VALUE SPACES.

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

This section of code is generated automatically when the Dates option is chosen from the Options Menu.

File control status switches are generated for all files specified.

(24 lines omitted)

These definitions are generated for all disk files specified from the File Control Information screen.

(6 lines omitted)

```
*****************************************************************
*            WORKSTATION FILE CONTROL DESCRIPTIONS              *
*****************************************************************
 77  LINENUM                        PIC 9(02) VALUE 1.
 77  PF-KEY                         PIC 9(02) VALUE 0.
 01  CURSOR-POS.
     03  MOD-COL                    BINARY.
     03  MOD-ROW                    BINARY.
 01  REDEF-CURS-POS                 REDEFINES CURSOR-POS.
     03  FILLER                     PIC  X(01).
     03  CURS-COL                   PIC  X(01).
     03  FILLER                     PIC  X(01).
     03  CURS-ROW                   PIC  X(01).
*****************************************************************
*                                                               *
*                   DISPLAY DEFINITIONS                         *
*                                                               *
*****************************************************************
*
 01  DISPLAY-REC1 USAGE IS DISPLAY-WS.
     05  FILLER          PICTURE IS  X(28)       ROW 03 COLUMN 28
         VALUE IS "Quarterly Payroll Processing".
     05  FILLER          PICTURE IS  X(21)       ROW 06 COLUMN 30
         VALUE IS "PF 1    Calculate FUI".
     05  FILLER          PICTURE IS  X(21)       ROW 08 COLUMN 30
         VALUE IS "PF 2    Calculate SUI".
     05  FILLER          PICTURE IS  X(21)       ROW 10 COLUMN 30
         VALUE IS "PF3     Print Reports".
     05  FILLER          PICTURE IS  X(23)       ROW 12 COLUMN 30
         VALUE IS "PF16    Exit Processing".
*****************************************************************
*                                                               *
*              DISPLAY DEFINITION FOR DISPLAY-REC 2             *
*                                                               *
*****************************************************************
*
 01  DISPLAY-REC2 USAGE IS DISPLAY-WS.
     05  FILLER          PICTURE IS  X(28)       ROW 02 COLUMN 28
         VALUE IS "Quarterly Payroll Processing".
     05  FILLER          PICTURE IS  X(19)       ROW 04 COLUMN 32
         VALUE IS "Generate SUI Report".
     05  FILLER          PICTURE IS  X(16)       ROW 09 COLUMN 32
         VALUE IS "Enter State Code".
     05  ROW09-COL51     PICTURE IS  X(02)       ROW 09 COLUMN 51
         RANGE IS FROM "AA"           TO "ZZ"
         SOURCE IS STATE-CODE     OBJECT IS STATE-CODE
     05  FILLER          PICTURE IS  X(26)       ROW 11 COLUMN 32
         VALUE IS "PF16  Terminate Processing".
 77  STATE-CODE                PIC X(02)       VALUE SPACES.
```

This code is generated whenever a workstation file is specified.

These lines create the workstation display definitions specified from the Source Creation Options menu and created with EZFORMAT. (See Sections 18.3.3 and 18.4.5.)

```
*****************************************************************
*                                                               *
*                     PRINT LINE DEFINITIONS                    *
*                                                               *
*****************************************************************
*
 01  PRINT-LINENUM        BINARY                    VALUE 55.
*
 01  LINE001.
     03  FILLER           PICTURE IS X(016)         VALUE SPACES.
     03  LINE001-FIELD01  PICTURE IS X(028)         VALUE
     "Payroll Quarterly SUI Report".
 01  LINE003.
     03  FILLER           PICTURE IS X(004)         VALUE SPACES.
     03  LINE003-FIELD01  PICTURE IS X(013)         VALUE
     "Employee Name".
     03  FILLER           PICTURE IS X(006)         VALUE SPACES.
     03  LINE003-FIELD02  PICTURE IS X(005)         VALUE
     "State".
     03  FILLER           PICTURE IS X(005)         VALUE SPACES.
     03  LINE003-FIELD03  PICTURE IS X(010)         VALUE
     "Qtr. Wages".
     03  FILLER           PICTURE IS X(005)         VALUE SPACES.
     03  LINE003-FIELD04  PICTURE IS X(010)         VALUE
     "SUI Amount".
 01  LINE005.
     03  LINE005-FIELD01  PICTURE IS X(020)         VALUE SPACES.
     03  FILLER           PICTURE IS X(004)         VALUE SPACES.
     03  LINE005-FIELD02  PICTURE IS X(002)         VALUE SPACES.
     03  FILLER           PICTURE IS X(007)         VALUE SPACES.
     03  LINE005-FIELD03  PIC 99,999.99             VALUE ZERO   .
     03  FILLER           PICTURE IS X(007)         VALUE SPACES.
     03  LINE005-FIELD04  PIC 9,999.99              VALUE ZERO   .
```

This code defines the report file layout, as created with EZPRINT. (See Section 18.4.6.)

```
*****************************************************************
*                                                               *
*                        GETPARM DEFINITIONS                    *
*                                                               *
*****************************************************************
*
 01  GETPARM-TYPE                PIC  X(2).
 01  GETPARM-FORM                PIC  X(3).
 01  GETPARM-PRNAME              PIC  X(8).
 01  GETPARM-AID-CHAR            PIC  X(1).
 01  GETPARM-MSG-NO              PIC  X(4).
 01  GETPARM-MSG-ID              PIC X(6)  VALUE "EXAMPL".
 01  GETPARM-MESSAGE             PIC X(78).
 01  GETPARM-MSG-LENGTH.
     03  FILLER                  BINARY    VALUE ZERO.
     03  GETPARM-MSG-LEN         BINARY    VALUE 78.
 01  GETPARM-KEYWORD1-CODE       PIC X(01).
 01  GETPARM-KEYWORD1            PIC X(08).
 01  GETPARM-KEYWORD1-VAL        PIC X(08).
 01  GETPARM-VAL1-LENGTH.
     03  FILLER                  BINARY    VALUE ZERO.
     03  GETPARM-VAL1-LEN        BINARY    VALUE ZERO.
 01  GETPARM-KEYWORD1-ROW.
     03  FILLER                  BINARY    VALUE ZERO.
     03  GETPARM-KEY1-ROW        BINARY    VALUE ZERO.
 01  GETPARM-KEYWORD1-COL.
     03  FILLER                  BINARY    VALUE ZERO.
     03  GETPARM-KEY1-COL        BINARY    VALUE ZERO.
 01  GETPARM-KEYWORD1-TYPE       PIC X(01).
 01  GETPARM-KEYWORD2-CODE       PIC X(01).
 01  GETPARM-KEYWORD2            PIC X(08).
 01  GETPARM-KEYWORD2-VAL        PIC X(08).

               *  *  *  *  *  *  *  *  *  *  *  *

 01  GETPARM-TEXT3-COL.
     03  FILLER                  BINARY    VALUE ZERO.
     03  GETPARM-TXT3-COL        BINARY    VALUE ZERO.
 01  GETPARM-PFKEY-CODE          PIC X(1)  VALUE "P".
 01  GETPARM-PFKEYS.
     03  GETPARM-PF1-16          BINARY    VALUE ZERO.
     03  GETPARM-PF17-32         BINARY    VALUE ZERO.
 01  GETPARM-DISABLE-ENTR        PIC X(1)  VALUE "E".
```

This code defines a "Generic GETPARM" with three keywords and three text messages. It is generated when the GETPARM option is chosen from the Source Creation Options Menu. See the Procedure Division also.

(51 lines omitted)

```
*****************************************************************
*                                                               *
*                  DEFINITIONS FOR LINK CALL(S)                 *
*                                                               *
*****************************************************************
*
 77  LINK-TO-NAME                PIC X(08) VALUE SPACES.
 77  LINK-TYPE                   PIC X(01) VALUE SPACE.
 77  LINK-LIBRARY                PIC X(08) VALUE SPACES.
 77  LINK-VOLUME                 PIC X(06) VALUE SPACES.
 01  LINK-PCOUNT.
     03  FILLER                  BINARY    VALUE ZERO.
     03  LINK-PCOUNT-NO          BINARY    VALUE ZERO.
 77  LINK-CEXIT-FLAG             PIC X(01) VALUE SPACE.
 77  LINK-CEXIT-MSG              PIC X(27) VALUE SPACES.
 01  LINK-CEXIT-MSG-LEN.
     03  FILLER                  BINARY    VALUE ZERO.
     03  FILLER                  BINARY    VALUE 27.
 77  LINK-HELP-FLAG              PIC X(1)  VALUE SPACE.
 77  LINK-RESERVED               PIC X(02) VALUE LOW-VALUES.
 77  LINK-CANCEL-RCVR            PIC X(128) VALUE SPACES.
 01  LINK-CANCEL-RCVR-LEN.
     03  FILLER                  BINARY    VALUE ZERO.
     03  FILLER                  BINARY    VALUE 128.
 01  LINK-CODE.
     03  FILLER                  BINARY    VALUE ZERO.
     03  LINK-CODE-VAL           BINARY    VALUE ZERO.
 01  LINK-RETURN-CODE.
     03  FILLER                  BINARY    VALUE ZERO.
     03  LINK-RTN-CODE-VAL       BINARY    VALUE ZERO.
```

Code for a LINK is generated when this option is chosen from the Source Creation Options menu. It uses parameters entered from the LINK Specification screen. (See both Section 18.4.4 and the Procedure Division.)

```
*****************************************************************
*                                                               *
*                      PUTPARM DEFINTIONS                       *
*                                                               *
*****************************************************************
*
*****************************************************************
*            DEFINITIONS FOR PUTPARM 01 (INPUT   )              *
*****************************************************************
 01  PUTPARM01-TYPE                PIC X(01) VALUE "E".
 01  PUTPARM01-PRNAME              PIC X(08) VALUE "INPUT   ".
 01  PUTPARM01-KEYCOUNT.
     03  FILLER                    BINARY    VALUE ZERO.
     03  FILLER                    BINARY    VALUE 03.
 01  PUTPARM01-AID                 PIC X(01) VALUE "@".
 01  PUTPARM01-LABEL               PIC X(08) VALUE "        ".
 01  PUTPARM01-REF-LABEL           PIC X(08) VALUE "        ".
 01  PUTPARM01-KEYWORD01           PIC X(08) VALUE "FILE    ".
 01  PUTPARM01-VALUE01             PIC X(08) VALUE SPACES.
 01  PUTPARM01-VAL-LEN01.
     03  FILLER                    BINARY    VALUE ZERO.
     03  FILLER                    BINARY    VALUE  08.
 01  PUTPARM01-KEYWORD02           PIC X(08) VALUE "LIBRARY ".
 01  PUTPARM01-VALUE02             PIC X(08) VALUE SPACES.
 01  PUTPARM01-VAL-LEN02.
     03  FILLER                    BINARY    VALUE ZERO.
     03  FILLER                    BINARY    VALUE  08.
 01  PUTPARM01-KEYWORD03           PIC X(08) VALUE "VOLUME  ".
 01  PUTPARM01-VALUE03             PIC X(06) VALUE SPACES.
 01  PUTPARM01-VAL-LEN03.
     03  FILLER                    BINARY    VALUE ZERO.
     03  FILLER                    BINARY    VALUE  06.
 01  PUTPARM-RTN-CODE.
     03  FILLER                    BINARY    VALUE ZERO.
     03  PUTPARM-RETURN-CODE       BINARY    VALUE ZERO.
*****************************************************************
*            DEFINITIONS FOR PUTPARM 02 (EOJ     )              *
*****************************************************************
 01  PUTPARM02-TYPE                PIC X(01) VALUE "E".
 01  PUTPARM02-PRNAME              PIC X(08) VALUE "EOJ     ".
 01  PUTPARM02-KEYCOUNT.
     03  FILLER                    BINARY    VALUE ZERO.
     03  FILLER                    BINARY    VALUE 00.
 01  PUTPARM02-AID                 PIC X(01) VALUE "p".
 01  PUTPARM02-LABEL               PIC X(08) VALUE "        ".
 01  PUTPARM02-REF-LABEL           PIC X(08) VALUE "        ".
```

This code constructs the specific PUTPARMs that were specified from the two PUTPARM screens after this option was chosen from the Source Creation Options menu. See Section 18.3.4; see also the Procedure Division.

```
*****************************************************************
*                      SORT CALL PARAMETERS                     *
*****************************************************************
 01  SORT-PARAMETERS.
     03  SORT-IN-FIL              PIC X(08) VALUE SPACES.
     03  SORT-IN-LIB              PIC X(08) VALUE SPACES.
     03  SORT-IN-VOL              PIC X(06) VALUE SPACES.
     03  SORT-OUT-FIL             PIC X(08) VALUE SPACES.
     03  SORT-OUT-LIB             PIC X(08) VALUE SPACES.
     03  SORT-OUT-VOL             PIC X(06) VALUE SPACES.
     03  SORT-FIELDS              OCCURS 8 TIMES.
         05  SORT-FLD-POS         PIC  Z(04).
         05  SORT-FLD-LEN         PIC  Z(03).
         05  SORT-FLD-TYPE        PIC  X(01).
         05  SORT-ORDER           PIC  X(01).
 01  SORT-RTN-CODE.
     03  FILLER                   BINARY    VALUE ZERO.
     03  SORT-RETURN-CODE         BINARY    VALUE ZERO.
```

These definitions are generated when an external sort is chosen from the Options Menu. The parameters were entered from the External Sort Specification Screen — see Section 18.4.2. See the Procedure Division also.

```
*****************************************************************
*                                                               *
*                        PROCEDURE DIVISION                     *
*                                                               *
*****************************************************************
*
 PROCEDURE DIVISION.
 START-PROGRAM.
     PERFORM INITIALIZATION.
     PERFORM MAIN-PROCESS.
*      UNTIL NO-MORE-filename OR PF-KEY IS EQUAL TO "xx".
     PERFORM TERMINATION.
 EXIT-PROGRAM.
     STOP RUN.

*****************************************************************
*                        INITIALIZATION                         *
*****************************************************************
*
 INITIALIZATION.
     ACCEPT DATEIN FROM DATE.
     MOVE YEAR-DIGITS TO RPT-YEAR.
     MOVE MONTH-DIGITS TO RPT-MONTH.
     MOVE DAY-DIGITS TO RPT-DAY.
*    MOVE RPT-DATE TO variable-name.
     PERFORM LOAD-GETPARM-PARAMETERS.
     PERFORM CALL-GETPARM.
     PERFORM SORT-PAYROLL.
     PERFORM LOAD-PARAMETERS-AND-SORT.
     OPEN I-O    WSFILE.
     OPEN OUTPUT PRTFILE.
     OPEN SHARED PAYROLL.
     OPEN SHARED SUI.
     OPEN I-O    SORTPAY.
     OPEN SHARED PERSONNL.
```

This skeletal section is the same for all programs. Note that the Procedure Division contains many lines of commented-out code at points where program expansion is likely.

This code is generated if the dates option is chosen from the Source Creation Options Menu. (The dates option default is automatically set to Y when you specify a printer or workstation file.)

These two lines execute the GETPARM call.

This line executes the internal sort.
This line executes the external sort.

These lines open the files specified from the File Control Information screen (Figure 18-2).

```
*****************************************************************
*                        MAIN PROCESS                           *
*****************************************************************
*
 MAIN-PROCESS.
     PERFORM DISPLAY-AND-READ-DISPLAY-REC1.
*    IF PF-KEY IS EQUAL TO "00" OR "xx"
     PERFORM DISPLAY-AND-READ-DISPLAY-REC2.
*    IF PF-KEY IS EQUAL TO "00" OR "xx"
     PERFORM READ-A-REC-FROM-PAYROLL.
     IF MORE-PAYROLL  AND RECORD-FOUND-ON-PAYROLL
     PERFORM REWRITE-A-REC-TO-PAYROLL.
     PERFORM DELETE-RECORD-FROM-PAYROLL.
     PERFORM WRITE-A-REC-TO-PAYROLL.
     PERFORM READ-A-REC-FROM-SUI.

               * * * * * * * * * * * * *

     PERFORM WRITE-A-PRINT-LINE.
     PERFORM CALL-PUTPARM01-FOR-INPUT.
     PERFORM CALL-PUTPARM02-FOR-EOJ.
     PERFORM LINK-TO-DISPLAY.

*****************************************************************
*                         TERMINATION                           *
*****************************************************************
*
 TERMINATION.
     CLOSE WSFILE.
     CLOSE PRTFILE.
     CLOSE PAYROLL.
     CLOSE SUI.
     CLOSE SORTPAY.
     CLOSE PERSONNL.
```

(13 lines omitted)

Code is automatically generated for closing all opened files.

```
*****************************************************************
*                                                               *
*                      INPUT OUTPUT ROUTINES                    *
*                                                               *
*****************************************************************
*
*****************************************************************
*                    DISPLAY AND READ DISPLAY REC1              *
*****************************************************************
*
 DISPLAY-AND-READ-DISPLAY-REC1.
     DISPLAY AND READ DISPLAY-REC1 ON WSFILE.
*          PFKEY xx, xx
*       ON PFKEY xx, xx imperative-statement.

               * * * * * * * * * * * *

*****************************************************************
*                    WRITE A PRINT LINE                         *
*****************************************************************
*
 WRITE-A-PRINT-LINE.
     ADD 1 TO PRINT-LINENUM.
     IF PRINT-LINENUM IS GREATER THAN 55
        PERFORM PRINT-PAGE-HEADERS.
*    WRITE PRTREC FROM LINE0xx.
*
 PRINT-PAGE-HEADERS.
     MOVE ZERO TO PRINT-LINENUM.
*    WRITE PRTREC FROM LINE001 AFTER ADVANCING PAGE.
*    WRITE PRTREC FROM LINE00x AFTER ADVANCING 2 LINES.
*    WRITE PRTREC FROM LINE00x AFTER ADVANCING 1 LINE.
*    WRITE PRTREC FROM LINE00x AFTER ADVANCING 1 LINE.
```

A routine is generated for each input and output statement in the Main Process.

(118 lines omitted)

(The last output routines.)

Print file I/O routines.

```
*****************************************************************
*                                                               *
*                      LOAD GETPARM ROUTINE                     *
*                                                               *
*****************************************************************
*
 LOAD-GETPARM-PARAMETERS.
     MOVE "I"                          TO  GETPARM-TYPE.
     MOVE "REQ"                        TO  GETPARM-FORM.
     MOVE "XXXXXXXX"                   TO  GETPARM-PRNAME.
     MOVE "@"                          TO  GETPARM-AID-CHAR.
     MOVE "0001"                       TO  GETPARM-MSG-NO.
     MOVE SPACES                       TO  GETPARM-MESSAGE.
     MOVE "K"                          TO  GETPARM-KEYWORD1-CODE.

               * * * * * * * * * * * *

*****************************************************************
*                                                               *
*                      CALL GETPARM ROUTINE                     *
*                                                               *
*****************************************************************
*
 CALL-GETPARM.
     CALL "GETPARM" USING
         GETPARM-TYPE,
         GETPARM-FORM,
         GETPARM-PRNAME,
         GETPARM-AID-CHAR,
         GETPARM-MSG-NO,
         GETPARM-MSG-ID,
         GETPARM-MESSAGE,
         GETPARM-MSG-LENGTH,
         GETPARM-KEYWORD1-CODE,

               * * * * * * * * * * * *


*****************************************************************
*                           SORT PAYROLL                        *
*****************************************************************
 SORT-PAYROLL.
     SORT SORT01
         ON DESCENDING KEY S01-ZIPCODE
         ON  ASCENDING KEY S01-EMP-LAST
         WITH DUPLICATES IN ORDER
*        INPUT  PROCEDURE IS SORT01-IN-PROC THRU SORT01-IN-EXT
         USING PAYROLL
*        OUTPUT PROCEDURE IS SORT01-OT-PROC THRU SORT01-OT-EXT
         GIVING SORTPAY.
```

These lines prepare "generic" parameters for the GETPARM. Ordinarily, specific parameters would be substituted before the program is compiled.

(35 lines omitted)

These lines call the GETPARM subroutine and pass the parameters to it.

(38 lines omitted)

This code executes the sort that uses the internal sort verb. Parameters were entered from the series of screens that request information for this type of sort. (See Section 18.4.2.)

```
*****************************************************************
*                                                               *
*              LOAD PARAMETERS AND CALL SORT ROUTINE            *
*                                                               *
*****************************************************************
*
 LOAD-PARAMETERS-AND-SORT.
     MOVE "PERSONNL"                  TO  SORT-IN-FIL.
     MOVE "PAYROLL "                  TO  SORT-IN-LIB.
     MOVE "SYSTEM"                    TO  SORT-IN-VOL.
     MOVE "SRTPRSNL"                  TO  SORT-OUT-FIL.
     MOVE "PAYWORK "                  TO  SORT-OUT-LIB.
     MOVE "WORK    "                  TO  SORT-OUT-VOL.
     MOVE 0050                        TO  SORT-FLD-POS(1).
     MOVE 005                         TO  SORT-FLD-LEN(1).
     MOVE "C"                         TO  SORT-FLD-TYPE(1).
     MOVE "A"                         TO  SORT-ORDER(1).
     MOVE 0225                        TO  SORT-FLD-POS(2).
     MOVE 010                         TO  SORT-FLD-LEN(2).
     MOVE "C"                         TO  SORT-FLD-TYPE(2).
     MOVE "A"                         TO  SORT-ORDER(2).
     MOVE 0000                        TO  SORT-FLD-POS(3).
     MOVE 000                         TO  SORT-FLD-LEN(3).
     MOVE "C"                         TO  SORT-FLD-TYPE(3).
     MOVE "A"                         TO  SORT-ORDER(3).
     MOVE 0000                        TO  SORT-FLD-POS(4).
     MOVE 000                         TO  SORT-FLD-LEN(4).
     MOVE "C"                         TO  SORT-FLD-TYPE(4).
     MOVE "A"                         TO  SORT-ORDER(4).
     MOVE 0000                        TO  SORT-FLD-POS(5).
     MOVE 000                         TO  SORT-FLD-LEN(5).
     MOVE "C"                         TO  SORT-FLD-TYPE(5).
     MOVE "A"                         TO  SORT-ORDER(5).
     MOVE 0000                        TO  SORT-FLD-POS(6).
     MOVE 000                         TO  SORT-FLD-LEN(6).
     MOVE "C"                         TO  SORT-FLD-TYPE(6).
     MOVE "A"                         TO  SORT-ORDER(6).
     MOVE 0000                        TO  SORT-FLD-POS(7).
     MOVE 000                         TO  SORT-FLD-LEN(7).
     MOVE "C"                         TO  SORT-FLD-TYPE(7).
     MOVE "A"                         TO  SORT-ORDER(7).
     MOVE 0000                        TO  SORT-FLD-POS(8).
     MOVE 000                         TO  SORT-FLD-LEN(8).
     MOVE "C"                         TO  SORT-FLD-TYPE(8).
     MOVE "A"                         TO  SORT-ORDER(8).
*****************************************************************
*                        CALL SORT                              *
*****************************************************************
      CALL "SORTCALL" USING SORT-PARAMETERS, SORT-RTN-CODE.
      IF SORT-RETURN-CODE IS NOT EQUAL TO ZERO
         DISPLAY "RETURN CODE FROM SORT IS ", SORT-RETURN-CODE
         GO TO EXIT-PROGRAM.
```

This section of code prepares parameters for the subroutine that invokes the external SORT utility. The parameters were entered from the External Sort Specification screen (Figure 18-7).

These lines call the subroutine.

```
*****************************************************************
*                                                               *
*                     CALL PUTPARM ROUTINES                     *
*                                                               *
*****************************************************************
*
*****************************************************************
*                   CALL PUTPARM01 FOR INPUT                    *
*****************************************************************
 CALL-PUTPARM01-FOR-INPUT.
     MOVE "SORTPAY"                    TO PUTPARM01-VALUE01.
     MOVE "PAYWORK"                    TO PUTPARM01-VALUE02.
     MOVE "WORK"                       TO PUTPARM01-VALUE03.
     CALL "PUTPARM" USING  PUTPARM01-TYPE,
                           PUTPARM01-PRNAME,
                           PUTPARM01-KEYCOUNT,
                           PUTPARM01-KEYWORD01,
                           PUTPARM01-VALUE01,
                           PUTPARM01-VAL-LEN01,
                           PUTPARM01-KEYWORD02,
                           PUTPARM01-VALUE02,
                           PUTPARM01-VAL-LEN02,
                           PUTPARM01-KEYWORD03,
                           PUTPARM01-VALUE03,
                           PUTPARM01-VAL-LEN03,
                           PUTPARM01-AID,
                           PUTPARM01-LABEL,
                           PUTPARM01-REF-LABEL,
                           PUTPARM-RTN-CODE.
     IF PUTPARM-RETURN-CODE IS NOT EQUAL TO ZERO
        DISPLAY "RETURN CODE FROM PUTPARM = ", PUTPARM-RETURN-CODE
        GO TO EXIT-PROGRAM.
*****************************************************************
*                   CALL PUTPARM02 FOR EOJ                      *
*****************************************************************
 CALL-PUTPARM02-FOR-EOJ.
     CALL "PUTPARM" USING  PUTPARM02-TYPE,
                           PUTPARM02-PRNAME,
                           PUTPARM02-KEYCOUNT,
                           PUTPARM02-AID,
                           PUTPARM02-LABEL,
                           PUTPARM02-REF-LABEL,
                           PUTPARM-RTN-CODE.
     IF PUTPARM-RETURN-CODE IS NOT EQUAL TO ZERO
        DISPLAY "RETURN CODE FROM PUTPARM = ", PUTPARM-RETURN-CODE
        GO TO EXIT-PROGRAM.
```

This section of code calls the PUTPARM subroutine for each of the PUTPARMs specified, and passes the appropriate set of parameters to it. (See Section 18.4.3.)

```
*****************************************************************
*                                                               *
*                    LINK TO DISPLAY  ROUTINE                   *
*                                                               *
*****************************************************************
*
 LINK-TO-DISPLAY.
     MOVE "DISPLAY "                     TO  LINK-TO-NAME.
     MOVE " "                            TO  LINK-TYPE.
     MOVE "        "                     TO  LINK-LIBRARY.
     MOVE "      "                       TO  LINK-VOLUME.
     MOVE  0                             TO  LINK-PCOUNT-NO.
     MOVE "C"                            TO  LINK-CEXIT-FLAG.
     MOVE "RETURN TO EXAMPLE           " TO  LINK-CEXIT-MSG.
     MOVE "H"                            TO  LINK-HELP-FLAG.
     CALL "LINK" USING LINK-TO-NAME,
                       LINK-TYPE, LINK-LIBRARY, LINK-VOLUME,
                       LINK-PCOUNT,
                       LINK-CEXIT-FLAG,
                       LINK-CEXIT-MSG, LINK-CEXIT-MSG-LEN,
                       LINK-HELP-FLAG, LINK-PFKEY-MASK,
                       LINK-CANCEL-RCVR, LINK-CANCEL-RCVR-LEN,
                       LINK-CODE, LINK-RETURN-CODE.
     IF LINK-RTN-CODE-VAL IS GREATER THAN ZERO
        DISPLAY "RETURN CODE FROM LINK IS ", LINK-RTN-CODE-VAL.
*****************************************************************
*                             SORT01 IN PROC                    *
*****************************************************************
 SORT01-IN-PROC.
     READ PAYROLL  NEXT AT END
         GO TO SORT01-IN-EXT.
*    Manipulate the Input File
     RELEASE SORT01-RECORD.
     GO TO SORT01-IN-PROC.
 SORT01-IN-EXT.
     EXIT.
*****************************************************************
*                             SORT01 OT PROC                    *
*****************************************************************
 SORT01-OT-PROC.
     RETURN SORT01 AT END
         GO TO SORT01-OT-EXT.
*     Manipulate the Output File
     PERFORM WRITE-A-REC-TO-SORTPAY.
     GO TO SORT01-OT-PROC.
 SORT01-OT-EXT.
     EXIT.
```

This is the code that calls the LINK subroutine, passing parameters entered from the LINK Specification Screen (Figure 18-9.)

This last section of code provides optional skeletal input and output routines for the internal sort. It is generated whenever this sort option is chosen. (The routines as generated have no effective content: this must be added.)

CHAPTER 19
COMPILE

19.1 INTRODUCTION

The COMPILE utility provides a convenient way to compile and link programs. By merely specifying the appropriate libraries, you can compile all the source files in a single library and then link all the object files produced by the compiler. The compiling and linking functions can also be done separately.

19.1.1 Overview

To begin compiling, you specify a *source library* containing program source files, all of which must be written in the same language. COMPILE supports COBOL, RPG II, BASIC, Assembler, FORTRAN, PL/I, and C.

The object files, as they are produced by the compiler, go into an *object library* which you also specify. If you are linking only, the object library provides the input for the Linker, so you do not specify a source library.

If you request linking, you can also specify a *load library* to be the destination of the linked object files produced by the Linker. If you do not specify a load library, these files go to the object library, where they replace the object files that were used as input for the Linker.

Note: *Table 19-1 in Section 19.2 shows which libraries to specify for each type of COMPILE operation.*

All standard compiler and linker options are available. You can compile and link either all or some of the files in a given library. You can also choose to have existing object or print files that might conflict with program output automatically scratched.

19.1.2 Software Requirements

COMPILE cannot compile unless the appropriate language compiler resides in the system library (@SYSTEM@), and it cannot link unless the Linker resides there.

19.1.3 Running COMPILE

Run the COMPILE utility from the Command Processor as you run any other program or utility. Enter COMPILE into the program name field, the appropriate library name into the program library field, and the appropriate volume name into the program volume field. (Since this utility does not reside in the system library, you must type the library and volume designations.)

The following sections describe COMPILE processing:

Section	Process
19.2	Specifying Libraries and Options
19.2.1	Selecting Individual Files for Processing
19.3	Final Specifications and Program Output

19.2 SPECIFYING LIBRARIES AND OPTIONS

When COMPILE processing begins, the Library and Options Specification screen appears (Figure 19-1).

```
Wang VS GETPARM v 7                    Parameter Reference Name:  LIBRARY
                                                    Message Id:  0001
                                                     Component:  COMP

                    Information Required by COMPILE

   VS Compile/Link Files Utility - Version  x.xx.xx (c) Copr. Wang 1986

This program compiles and/or links programs from a library. Please supply the
following information and press ENTER to continue:

LANGUAGE = COBOL      (ASSEMBLER, BASIC, CC, COBOL, FORTRAN, PLI, RPGII)
LINK     = NO         (YES, NO, ONLY)

Source library:                    SRCLIB   =          SRCVOL   =
Object library (compiled programs): OBJLIB  =          OBJVOL   =
Load library                       LOADLIB  =          LOADVOL  =
 (specify only if LINK = YES/ONLY, and if different from the object library)

Select files from input library ?  SELECT   = NO (uses OBJLIB if LINK=ONLY)
Scratch old objects and prints ?   SCRATCH  = YES

Or Select:
                                        (13) Information  (16) To exit COMPILE
```

Figure 19-1. COMPILE Library and Options Specification Screen

Enter information in the fields as follows:

LANGUAGE -- Enter the language in which the source programs are written. The screen lists all valid choices. Note that an unconventional spelling is sometimes required: *CC* instead of C, *PLI* instead of PL/I, and *RPGII* instead of RPG II.

LINK -- Enter YES to compile and link all the programs. Enter ONLY to link without compiling. NO (the default) causes the program to compile without linking.

Note: *The next six fields are used to specify source, object, and load libraries. You need not specify all three types of library for every COMPILE function. See the comments as well as Table 19-1 below.*

SRCLIB -- Specify the source library containing the program files to be compiled. This field is ignored when you are not compiling (i.e., when LINK = ONLY).

SRCVOL -- Identify the volume containing the source library, if one is specified.

OBJLIB -- Specify the object library. When you are compiling (LINK = YES or NO), this library receives the compiled object code. When you are linking (LINK = YES or ONLY), this library provides the input for the linker.

OBJVOL -- Identify the volume containing the object library.

LOADLIB -- Specify the library to receive the linker output. You should do this only when LINK = YES or ONLY, and when you want the load library to be different from the object library (OUTLIB). A load library is always optional.

LOADVOL -- Identify the volume containing the load library, if one is specified.

Table 19-1. Libraries Specified for COMPILE Operations

Operation	Source Library	Object Library	Load Library
Compile only	Yes	Yes	No
Link only	No	Yes	Optional
Compile and link	Yes	Yes	Optional

SELECT -- Your response determines whether the specified operation affects some or all of the files in the library. Enter YES if you wish to select only some program files within the library. The default NO includes the whole library in the operation.

SCRATCH -- The default YES causes any object, link, and print files that already exist in the library to be scratched without notification. Enter NO to preserve these files. If SCRATCH = NO, and an existing file is in danger of being overwritten, you are notified and given the options of (1) pressing PF3 to scratch the original file and replace it with the new one, or (2) giving the new file a different name.

When you have entered all your specifications, press ENTER to continue. Alternatively, you can press PF13 for information or PF16 to exit from COMPILE without processing any files.

19.2.1 Selecting Individual Files for Processing

If you have chosen to select individual files instead of compiling the whole contents of the library (by setting the value of SELECT to YES from the Library and Options Specification screen, Figure 19-1), COMPILE lists the files in the library on a File Selection screen (Figure 19-2).

```
Place a non-blank character to the left of the files you want to process, and
press ENTER to continue.

■ TESTPRG1    ■ TESTPRG2    ■TESTPRG3    ■TESTPRG4

                        4 files in TESTLIB  on VOL111

ENTER - Continue      2 - First screen      16 - Exit without selecting
```

Figure 19-2. A Sample COMPILE File Selection Screen

Indicate the files you wish to compile or link by entering any nonblank character at the pseudoblank to the left of the file name.

Press ENTER to continue processing. Alternatively, you can press PF2 to return to the Library and Options Specification screen, where you can change specifications, or you can press PF16 to terminate COMPILE processing without selecting files.

19.3 FINAL SPECIFICATIONS AND PROGRAM OUTPUT

If you specified compiling, a Compiler Options screen appears when you press ENTER from the Library and Options Specification screen or the File Selection screen. It shows the compilation options for the language compiler you are using. If you are unfamiliar with these, see the appropriate appendix in the language reference manual. (The Compiler Options screen appears only once. The options you select are applied to all files compiled in this run of the program.) When you finish specifying options, press ENTER to begin the compiling process.

If you specified linking only, COMPILE displays the Linker Options screen when you press ENTER from the Library Definition and Options screen or the File Selection screen.

If you specified both compiling and linking, the COMPILER Options screen appears when you press ENTER from the Library Definition and Options screen or the File Selection screen, and the Linker Options screen appears immediately after the first program is compiled.

The Linker Options screen shows the available link options and allows you to change them. See the *VS Linker Reference* for a detailed account of Linker options.

The Linker Options screen, like the Compiler Options screen, appears only once. The options you select are applied to all files linked in this run of the COMPILE utility.

The last screen displayed is the COMPILE End of Job (EOJ) screen. It gives you the options of pressing PF1 to return to the first COMPILE screen, so that you can compile another library, or PF16 to terminate the program.

In addition to the compiled or linked output files, COMPILE generates print files and a statistics log. All are placed in the user's default spool library (as set with PF2 from the Command Processor menu). These files are named as follows:

File	Name
Source listing print files	Same as source program name
Linker print files	LINKxxxx (xxxx represents a unique number generated by the operating system)
Compile statistics	CLOGxxxx (xxxx represents a unique number generated by the operating system)

CHAPTER 20
EZPRINT

20.1 INTRODUCTION

The EZPRINT utility provides a method of creating or modifying report formats for forms up to 180 characters wide, and then generating source code from the report format in either BASIC, COBOL, RPG II, or Assembly Language. The code contains program statements capable of reproducing the format features you have designed.

EZPRINT's code cannot be compiled or executed as it stands, but you can save much tedious effort by incorporating the statements in a larger program.

20.1.1 Overview

On entering EZPRINT, you provide the name of a *report format file*. This is a file, created through EZPRINT, that contains specifications for formatting a particular report. You can specify either an existing format file you want to modify or a new one you want to create. The program then displays a screen (blank if you are creating a new format file) on which you can use an extensive menu of commands to design or redesign the report format. You can check your work by printing and displaying it at any point in the design process. When you finish, EZPRINT creates or updates the format file, requests a name, location, and language for the source code file, and uses the format file to generate the latter.

20.1.2 Software Requirements

For EZPRINT to function fully, the VS DISPLAY utility must be present in the system library (@SYSTEM@).

20.1.3 Running EZPRINT

Run the EZPRINT utility from the Command Processor as you run any other program or utility. Enter EZPRINT into the program name field, the appropriate library name in the library field and the appropriate volume name in the volume field.

The following sections describe EZPRINT processing:

Section	Process
20.2	Specifying a Format File
20.3	Designing or Modifying a Report Format
20.3.1	Entering Format Information
20.3.2	Menu Commands
20.3.3	Copying, Moving, and Merging
20.3.4	The Repeat Function
20.3.5	Arranging and Printing Sample Reports
20.3.6	The Carriage Control Column and RPG II Code
20.4	Specifying EZPRINT Output

20.2 SPECIFYING A FORMAT FILE

When EZPRINT processing begins, the Format File Specification screen (Figure 20-1) appears.

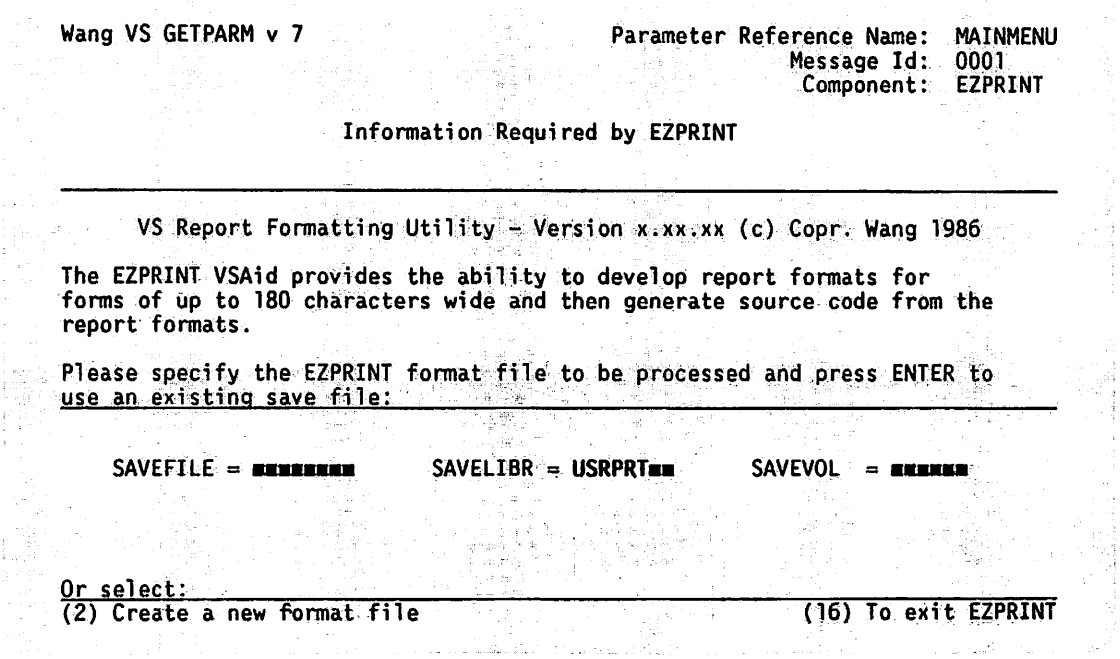


Figure 20-1. EZPRINT Format File Specification Screen

Use the SAVEFILE field on this screen to specify a format file -- either an old one you want to modify or a new one you want to create. EZPRINT saves an image of a report format in this file and uses it to create source code.

When you have entered the name, library, and volume for the format file, press ENTER if you are using an existing file or PF2 if you are creating a new file. (If you supply a file name beginning with # or ## and press PF2 to create a new file, EZPRINT creates a temporary file that is scratched when you log off the system or return to the Command Processor. Ordinarily, this is done only after the file has served its purpose in the generation of source code.)

Press PF16 if you want to exit EZPRINT without specifying a format file.

20.3 DESIGNING OR MODIFYING A REPORT FORMAT

When you press either ENTER or PF2 from the Format File Specification screen, the Report Formatting screen (Figure 20-2) appears.

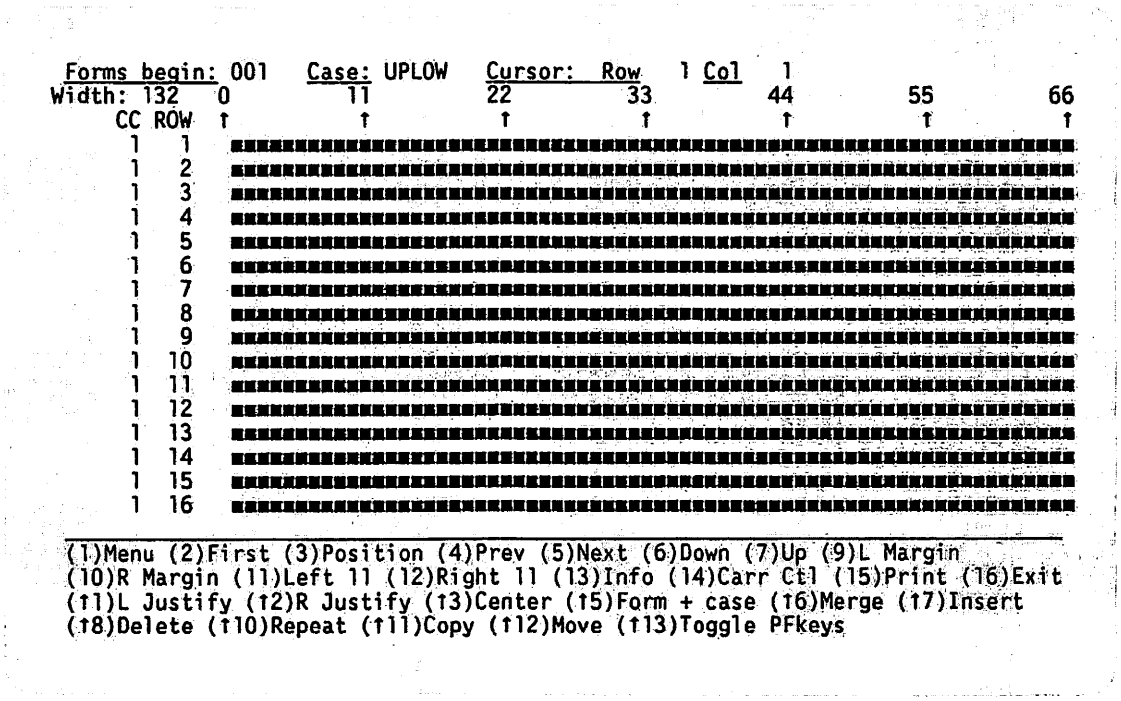


Figure 20-2. EZPRINT Report Formatting Screen

The top of the screen shows the form width, cursor position, and column numbers. The row numbers and carriage control column (see Section 20.2.5 to learn more about the latter) are at the left, and a condensed menu appears in the bottom four lines.

A "window" representing a portion of the report page occupies the rest of the screen. If you are creating a new format, the window is filled with pseudoblanks, but if you are using an existing format definition, the window contains a representation of the first part of it.

Although it can be expanded to 180 columns, the default width of the report page is 132 columns, half of which is shown in the window. PF keys can be used as in the DISPLAY utility (Chapter 11) to move the window on the page, and they can also be used to manipulate the contents in various ways. The condensed menu at the bottom of the screen summarizes these PF key commands; when you press PF1, an expanded version of the menu replaces the entire formatting screen (see Figure 20-3). Section 20.3.2 describes all the menu commands individually.

20.3.1 Entering Format Information

Using a portion of the formatting screen as the facsimile of an equivalent portion of the printed report page, you create or modify the report format by moving the cursor to the point where you want a particular item to appear and entering characters to represent it.

Although you could use the screen to represent an entire report page, you would be unlikely to do so, since you typically use the generated code in units that describe no more than a few lines at a time. A single format file is capable of representing as many as 999 lines, and there is no good reason to limit it to one page.

Text can be entered in the visible portion of the screen only, and the window does not automatically shift to the right when you reach the last visible column in a row. Instead, the text wraps to the first visible column in the next row. For example, if the screen is showing columns 0 - 66, and you enter the words "Monthly Subtotals" beginning at column 63, "Mont" will go into columns 60 - 66 and "ly Subtotals" into columns 0 - 11 of the next row. To avoid this kind of inconvenience, it is best before entering an item to shift the window to a point where it can accommodate the entire item, or as much of it as possible.

Not every text-manipulation function is limited to the visible portion of the screen, however. The sections that follow describe those functions in detail.

How Report Items Are Represented on the Formatting Screen

There are three kinds of report items to represent on the screen:

Literal Text -- Alphanumeric text that does not vary, e.g., headings, labels, and the like. Enclose this text in double quotation marks when you enter it on the screen. (The quotation marks will not appear in the report.) Place the text exactly where you want it. For example, a character placed in column 12 on the formatting screen will appear in column 12 on the report page, regardless of the position of the quotation marks that enclose it.

Variable Character Fields -- Data fields that will be filled with information expressed in alphanumeric characters (such as names) when the report is processed. Enter uppercase Xs in the exact position, and to the maximum length, that you want each field to occupy in the report.

Variable Numeric Fields -- Data fields that will be filled with information expressed in numbers when the report is processed. Enter 9s in the exact location, and to the maximum length, that you want each field to occupy in the report. Numeric fields may also contain the following variants:

- You may insert edit characters acceptable to the syntax of the language you are using in the places where you want them to appear in the report (e.g., *$999,999.99*). See the appropriate language reference for the characters that can be used in this way.

- To suppress leading zeros in a given position, enter a *Z* instead of a 9. For example, the value 6 would appear as *0006* in a field represented by *9999*, as *06* in a field represented by *ZZ99*, and as *6* in a field represented by *ZZZ9*.

- Dollar signs may also be used to represent digits in a numeric field. This provides the effect of a floating dollar sign: it both suppresses leading zeros and places the dollar sign in the rightmost eligible position, i.e., the rightmost position in which the representation of the field contains a dollar sign. For example, the value 6.5 appears as *$6.50* in a field represented by *$$$9.99*, as *$ 6.50* in a field represented as *$ZZ9.99*, and as *$006.50* in a field represented as *$999.99*.

Some further examples:

On the Formatting Screen	**In the Report**
"DATE:" Z9/99/99	DATE: 1/29/87 DATE: 12/01/88
"Subtotal" $ZZ9,999.99	Subtotal $ 7,862.47 Subtotal $386,591.25 Subtotal $ 0,016.32
"REVENUE" $$$9,999.99	REVENUE $1,850.00 $0,772.33 $322,454.79
"City" XXXXXXXXXXXX	City PITTSBURGH LOS ANGELES WEST SPRINGF
"City" XXXXXXXXXXXXXXXXXXXX	City PITTSBURGH LOS ANGELES WEST SPRINGFIELD

(The only difference between the last two examples, apart from the alignment of the literal text "City," is that the first limits the field to 12 characters, truncating one of the records.)

20.3.2 Menu Commands

When you press PF13 from the Report Formatting screen, the Report Manipulation menu (Figure 20-3) appears. It is an expanded version of the menu at the bottom of the Report Formatting screen.

```
                        Report Manipulation Options

(1) Display menu of report options      (↑1) Left justify target row
(2) Display the first 16 lines          (↑2) Right justify target row
(3) Update the cursor row and column    (↑3) Center target row

(4) Display the previous 10 lines       (↑5) Define print form and case
(5) Display the next 10 lines           (↑6) Merge two rows
(6) Move the display back 1 line        (↑7) Insert 1 character *
(7) Move the display forward 1 line     (↑8) Delete 1 character *

(9) L Margin display first 66 columns   (↑10) Repeat character/string
(10) R Margin display last 66 columns   (↑11) Copy a row **
(11) Move the display left 11 columns   (↑12) Move a row **
(12) Move the display right 11 columns  (↑13) Toggle shifted PFkeys

(13) Display the information screen     Target row is the row that the cursor
(14) Specify sample report CC           is currently positioned at.
(15) Print and display sample report
(16) Exit report definition

  *  Functions on the entire line, not just the characters displayed.
 **  Source and destination row numbers will appear in the top right corner.
          Press ENTER to continue, or PF15 to print this screen
```

Figure 20-3. EZPRINT Report Manipulation Menu

You can print a convenient copy of this menu by pressing PF15 while it is displayed. To return to the Report Formatting screen, press ENTER.

The basic commands for moving the window, which involve PF keys 2 to 12, unshifted, are essentially similar to the DISPLAY utility commands that employ the same keys. (EZPRINT does not use PF8, since it has no equivalent to DISPLAY's Search feature.)

The following list describes the PF key functions one at a time. Functions that need more extensive discussion are treated in the sections that follow the list.

In the list, an arrow preceding a key number (e.g., PF↑7) indicates that the key is shifted.

PF Key	Function
1	Displays the Report Manipulation menu (Figure 20-3).
2	Moves the window to the first 16 lines of the screen, putting the cursor on Line 1 in the same column as when the command was given.
3	Updates the display at the top of the screen to show the current cursor column and row. (This display does not change automatically when you move the cursor.)
4	Moves the window back 10 rows (text scrolls downward). If there are fewer than 10 rows to the beginning of the text, the window moves back to Row 1 and the workstation alarm sounds.
5	Moves the window forward to show the next 10 rows (text scrolls upward). After Row 999, Row 1 reappears.
6	Moves the window back one row (text scrolls downward). If Row 1 is the top row already, the workstation alarm sounds.
7	Moves the window forward one row (text scrolls upward). If 999 is the bottom row when the command is given, Row 1 appears at the bottom of the window.
9	Moves the window to its leftmost position, showing the first 66 characters in the displayed rows. (If the form width has been defined as less than 66 characters, only the columns that fall within the defined width contain pseudoblanks; the part of the screen to the right of this area is dark.)
10	Moves the window to its rightmost position, showing the last 66 characters in the displayed rows. If the window is already at the right margin, or if the width is 66 or less, the workstation alarm sounds.
11	Moves the window 11 columns to the left (text moves to the right). If, for instance, the window shows Columns 22 through 88 before the command, afterwards it shows Columns 11 through 77. If the left margin is already displayed, the workstation alarm sounds.
12	Moves the window 11 columns to the right (text moves to the left). If, for instance the window shows Columns 22 through 88 before the command, afterwards it shows Columns 33 through 99. If the right margin is already displayed, the workstation alarm sounds.

PF Key	Function
13	Displays an information screen summarizing methods and options.
14	Modifies carriage control for printing format samples or generating RPG II code. See Section 20.3.5.
15	Prints a sample report, using the formatted lines as a template, and then displays the printed results by means of the VS DISPLAY utility. See Section 20.3.5.
16	Closes the report definition phase of the program: this is the way to exit from the Report Formatting screen when you finish the designing process.
↑1 (17)	Left-justifies the row the cursor is on by moving all text to the left until the first character is in Column 1. Shifts the window if necessary to show the left-hand margin. This command affects *all* characters in the row, not just the visible ones.
↑2 (18)	Right-justifies the row the cursor is on by moving all text to the right until the last character is in the last defined column.[1] Shifts the window if necessary to show the right-hand margin. This command affects *all* characters in the row, not just the visible ones.
↑3 (19)	Centers the contents of the row the cursor is on.[1] Affects *all* characters in the row, not just the visible ones.
↑5 (21)	Displays a screen on which you can specify the width of the report form (maximum 180 columns, minimum 11, default 132)[1], and the input mode for letters. The default for the latter is UPLOW, which permits the entry of uppercase and lowercase letters; this may be changed to UPPER, which translates all letters to uppercase as they are entered. (Changes of width and input mode affect text entry only: nothing previously entered is changed.)
↑6 (22)	Merges one row into another. See Section 20.3.3.

[1] If you intend to print a report whose width varies from the default 132 columns, be sure you use PF↑5 to set the width *before* you use either PF↑2 for right justification or PF↑3 for centering. Otherwise these operations are based on the default right margin, and must be repeated after the new width is set. Existing lines that are right-justified or centered are *not* automatically adjusted when you change the form width.

PF Key	Function
↑7 (23)	Inserts one blank character at the cursor position. Since EZPRINT has no "insertion mode," the best way to insert characters into an already formatted line is first to insert the required number of blanks and then to type new characters over them. As you insert blanks, the text already in the line (including the characters not visible in the window) moves toward the right until a nonblank character reaches the right margin. At that point the function is disabled, and no more characters can be inserted until you delete some text from the end of the line.
↑8 (24)	Deletes one character at the cursor position, and moves all characters from there to the end of the row one column to the left. (This includes characters not currently visible in the window.) This command affects only one row.
↑10 (26)	Repeats a character or string. This function is described in Section 20.3.4.
↑11 (27)	Copies a row. This function is described in Section 20.3.3.
↑12 (28)	Moves the contents of one row to another. This function is described in Section 20.3.3.
↑13 (29)	This toggle has three positions. The first time the key is pressed, it blanks out the last two rows of the menu at the bottom of the screen; the second time, it blanks out the first two rows. (These actions do not disable the functions involved, but merely remove them from the menu, making the remaining lines easier to read.) When the key is pressed a third time, all four rows of the menu are redisplayed.

20.3.3 Copying, Moving, and Merging

Copying, moving, and merging are three functions that transfer text from one line to another.

- *Copying* reproduces text on a new line while leaving it unchanged on the original line.

- *Moving* places text on a new line and leaves the original line blank.

  Both obliterate the previous contents of the new line.

- *Merging*, or overlaying, is a less destructive kind of copy; that is, it leaves the text on the original line, but (depending on the option you choose) it can also leave all or part of the new line's previous contents unchanged.

The operations of the three functions are similar. Figure 20-4 shows the screen for a merge, but is also sufficient to illustrate the copying and moving functions.

Copy or Move

Place the cursor on the row you want to copy or move (this is not mandatory, but it is convenient), and press SHIFT and PF11 to copy or SHIFT and PF12 to move. The screen background darkens as the pseudoblanks disappear. One of the following messages appears in the upper right-hand corner:

```
COPY ROW 001 to 002

MOVE ROW 006 to 007
```

(The row numbers shown are examples.) The default source row is the one the cursor is on when you give the command; the default destination row is simply that row plus 1. Both row numbers are highlighted and the cursor is on the first, so they are easy to change. You can, if you wish, enter the numbers of rows that are not displayed in the screen window.

When you have the source and destination lines properly specified -- with the text displayed on the screen this is easy to check -- press ENTER to carry out the copy or move. (If you change your mind about proceeding with the operation, you can return to the formatting screen by pressing PF1 instead of ENTER.)

Merge

A merge (or overlay) is initiated in much the same way as a copy or move (using the SHIFT key with PF6), but you have one additional option. It is explained on the Merge screen. Figure 20-4 shows an example of this screen.

```
Forms begin: 001   Case: UPLOW   Cursor:  Row  11 Col   1 OVERLAY 011 TO 012
Width: 132   0         11        22        33        44        55        66
     CC ROW  ↑         ↑         ↑         ↑         ↑         ↑         ↑
      1   1
      1   2
      1   3
      1   4
      1   5
      1   6
      1   7
      1   8
      1   9
      1  10
      1  11    "LITERAL TEXT STRING"
      1  12             "Numeric field:" $ZZ9,999,999.99
      1  13
      1  14
      1  15
      1  16
_____________________________________________________________________________
(1)Return          (4)Non-destructive overlay          (5)Destructive overlay
A non-destructive overlay will not destroy any non-blank characters on the
DESTINATION line; whereas, a destructive overlay will copy all non-blank
characters from the source line to the DESTINATION line.
```

Figure 20-4. A Sample EZPRINT Merge Screen

As on the Copy or Move screen, the background darkens as the pseudoblanks disappear. The upper right corner contains the message "OVERLAY **xxx** to **xxx**" where the first (source) row is the one the cursor was on when the command was given, and the second (destination) row is the one below the first.

Modify the source and destination row numbers, if necessary, exactly as you do in copying or moving. At that point, however, you must choose between PF4 and PF5 for a nondestructive or a destructive overlay.

Unlike the Copy and Move functions, which affect whole lines, the Merge function affects individual characters.

A *nondestructive overlay* copies characters in the source line only if they correspond to blanks in the destination line. It thus preserves all the characters in the destination line except the blanks. The Carriage Control number (see Section 20.3.5) of the destination line remains unchanged.

A nondestructive overlay on the sample screen would produce

```
"LITERAL TEXT STRING"                              (Row 11)
"LITERAL "Numeric field:" $ZZ9,999,999.99          (Row 12)
```

A *destructive overlay* copies all nonblank characters in the source line to the destination line. It thus preserves only the characters in the destination line that correspond to blanks in the source line and destroys the rest. A destructive overlay also changes the Carriage Control number (see Section 20.3.5) of the destination line to that of the source line.

A destructive overlay on the sample screen would produce

```
"LITERAL TEXT STRING"                              (Row 11)
"LITERAL TEXTeSTRING"ld:" $ZZ9,999,999.99          (Row 12)
```

Your choice depends, therefore, on whether you want the contents of the destination line or the source line to predominate in the merged line. For the first, press PF4 (nondestructive overlay); for the second, press PF5 (destructive overlay).

If you decide not to proceed with the merge, press PF1 instead of PF4 or PF5. You are returned to the formatting screen.

Note: *Keep the following points in mind when working with merges:*

- *As the examples show, a merge may produce a line containing unpaired quotation marks. It is your responsibility to check for these.*

- *When a copy, move, or merge is completed, the formatting screen returns with the destination row in the top position, no matter which row occupied that position when the command was first given. Forgetting this can cause confusion when you check results.*

20.3.4 The Repeat Function

This function is particularly useful for drawing boxes on the screen (and, ultimately, in the printed report). It permits you to specify a character (including a blank) and have this character entered in as many columns across a row or as many rows down a column as you require. You can even choose both functions at once and have a rectangular area of the screen filled with iterations of the specified character.

The Repeat function can also be used to reproduce any string of characters as many times as you specify in a vertical column on the screen.

To choose the Repeat function, place the cursor at the point where you want the repetition to begin and press SHIFT and PF10. The Repeat Option screen appears. Figure 20-5 shows a sample of this screen.

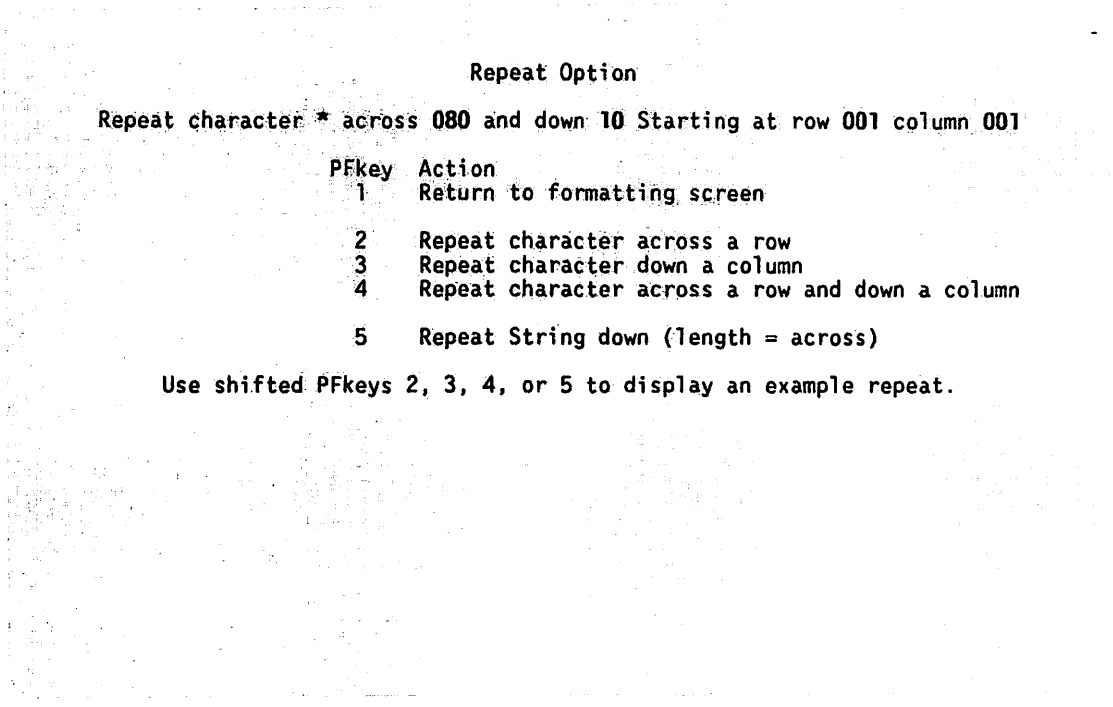


Figure 20-5. A Sample EZPRINT Repeat Option Screen

The line at the top of the Repeat Option screen shows five highlighted defaults which you may change to specify the conditions of the repetition. The cursor is positioned at the first, i.e., the character to be repeated. The cursor position when the command is given is the default starting position; the other defaults echo the entries made the last time the command was given. The sample screen shows entries that could be used to draw a box 80 columns wide by 10 columns long, using asterisks.

The PF key commands that actually place characters on the screen are listed in the middle of the Repeat Option screen. If you press any of these PF keys together with the SHIFT key, a simple "before-and-after" example of its effect is displayed at the bottom of the screen.

The commands work as follows:

- To draw a horizontal row (80 asterisks in the example), press PF2. The formatting screen returns with the new characters in place. (The row in which they are drawn is now the top row in the window.) The specifications are taken from the line at the top of the Repeat Option screen; this command ignores the number specified for vertical iterations.

- To draw a vertical column (10 asterisks in the example), press PF3. The formatting screen returns with the new characters in place; the row containing the first is now the top row in the window. This command ignores the number entered at the top of the screen for horizontal iterations.

- To fill in a solid area on the screen (80 columns by 10 rows of asterisks in the example), press PF4. The formatting screen returns with the new characters in place. The top row of characters is at the top of the window.

Note: *These are the commands used to draw boxes. The usual way is to fill a solid area with characters and then fill a smaller area inside it with blanks, using only PF4. The program's "memory" for defaults makes the process fairly convenient (and certainly much less tedious than writing program code for this purpose).*

When the formatting screen reappears after a repeat command, the leftmost column it shows is the leftmost column affected by the command. This means that when the screen returns after you have blanked out the inside of a box, you must shift the window to the left in order to see the left side of the box.

- To repeat a string, first place the cursor on the initial character and press SHIFT and PF10. When the Repeat Option screen appears, the first default in the line at the top shows the character you placed the cursor on. In the next position ("across"), enter the number of characters in the string you want to repeat. In the third ("down"), enter the number of rows downward. Then press PF4. The string is repeated in a vertically aligned column, its length the number of rows you have specified.

If you decide to skip the Repeat function, press PF1 to get back to the formatting screen.

20.3.5 Arranging and Printing Sample Reports

As a way to check the appearance of the format during the design process, EZPRINT provides a command for printing out a sample at any time. Whenever you press PF15, the program sends an image of the format as currently designed to the printer, and also displays it on the screen by means of the VS DISPLAY utility. These services enable you to examine your results and if necessary make adjustments without having to leave EZPRINT.

Because the Report Formatting screen may be crowded with many different format lines designed to go into different parts of a report, or even into different reports, it may be desirable to separate these lines in the printout. So that you will not have to leave blank lines on the formatting screen for this purpose (wasting some of the 999 available lines, and increasing the amount of window-shifting necessary), EZPRINT provides the Carriage Control column at the beginning of each row. Numbers entered in this column cause blank lines to be generated in the printed sample. Alternatively, an E can be entered to cause a page eject.

Suppose, for example, that Row 9 marks the beginning of a new section in your format definition, and you want to keep it clearly separate in the sample printout from the section that ends with Row 8. Place the cursor on Row 9 (the first row of the new section, not the last row of the old). Then press PF14. The screen background darkens as the pseudoblanks disappear; the window shifts to the left if necessary, and the cursor moves to the highlighted Carriage Control (CC) column of Row 9. (This command is specific to the row, so you must place the cursor on the right row before pressing PF14.)

You can now enter any number from 1 to 9 in the CC column, and an equivalent number of blank lines will later appear in the printed sample between the text in Row 8 and the text in Row 9 on the screen. The blank lines are generated immediately above the row where the number is placed.

If you would like the format lines beginning at a particular row to appear at the top of a page, you can enter an E in the CC column for that row. The result is a new page in the printout, with the designated text at the top of it.

After you enter a number from 1 to 9, or E, in the CC column, press ENTER to record the change. The formatting screen reappears.

Line moves and merges affect the CC column as well as the text in the line. See Section 20.3.3 for details.

20.3.6 The Carriage Control Column and RPG II Code

The commands you place in the CC column affect the sample printout. With one exception, they have no effect on the source code that EZPRINT generates. That exception is code generated in RPG II. In that language, when the CC column contains a number greater than 1, EZPRINT generates an output specification that causes the appropriate number of lines to be skipped before the line in question is printed. If the CC column for a particular line contains an E, EZPRINT generates an output specification that causes the line to be treated as a header.

RPG II programmers can save themselves inconvenience by making sure that any values they have entered in the CC column for the purpose of printing sample reports are, if necessary, changed to the values they want in the generated code statements before allowing the generation process to begin.

20.4 SPECIFYING EZPRINT OUTPUT

When you have no more report format lines to define or modify, press PF6. The Output Specification screen appears (Figure 20-6).

```
Wang VS GETPARM v 7                          Parameter Reference Name:  EXITMENU
                                                          Message Id:  0002
                                                           Component:  EZPRNT

                        Information Required by EZPRINT

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

      VS Report Formatting Utility - Version 2.00.02 (c) Copr. Wang 1986

Please specify the source file and language option then,
press ENTER to create source code:

    SRCFILE  = ********     SRCLIBR  = ********     SRCVOL   = ******
    LANGUAGE = C  Language option: 'B'ASIC, 'C'OBOL, 'R'PGII, 'A'SSEMBLER

Or select:
                                        (16) To exit without creating source file
```

Figure 20-6. EZPRINT Output Specification Screen

Use the fields of this screen to specify

- A name, library, and volume for the source file you want EZPRINT to create.
- The language in which you want the source code to be generated. Your choices for the latter are BASIC (B), COBOL (C), RPG II (R), and VS Assembly Language (A).

When you have specified the file and language, press ENTER to generate the source code.

If you decide at this point to exit without creating source code, press PF16 instead of ENTER. The format file you created or modified preserves your report format specifications. If you gave it a temporary file name, however, as described in Section 20.2, it will disappear when you log off the system. This is why it is generally unwise to use a temporary file name unless you intend to complete the process by generating source code in the same EZPRINT session.

When you press ENTER from the Output Specification screen, the source file is created and the EZPRINT End-of-Job (EOJ) screen (not shown) appears. From the EOJ screen you have the following options:

PF Key	Option
1	**Specify another savefile** -- Pressing PF1 returns you to the Format File Specification screen (Figure 20-1), where you can specify a new report format file and begin EZPRINT processing again.
16	**End Processing** -- Press PF16 to exit from EZPRINT and return to the VS Command Processor or to the program or procedure from which EZPRINT was called.

CHAPTER 21
PROCGEN

21.1 INTRODUCTION

The PROCGEN (Procedure Generation) utility provides a convenient way to create or modify VS Procedure Language routines. By running PROCGEN, making selections from its menu, and providing the requested parameter information, you can, without writing a line of code, generate a procedure that performs any of the system call statements and runs any of the programs in the following list:

Programs				Call Statements
ASSEMBLE	CREATE	FLOPYDUP	SORT	DISMOUNT
BACKUP	DATENTRY	INQUIRY	TAPECOPY	MOUNT
BASIC	DISKINIT	LINKER	TAPEINIT	RUN
COBOL	DISPLAY	PLI	TRANSL	SCRATCH
COPY	EDITOR	REPORT	VERIFY	SET
COPYWP	EZFORMAT	RPGII		

By means of the RUN statement, a PROCGEN procedure can run user programs or other system programs, although it cannot provide programs run in this way with parameter information.

PROCGEN's output is a standard-format VS Procedure Language file which can be modified by means of the Editor.

Modification may be necessary. PROCGEN is not intended as a substitute for Procedure Language and cannot eliminate entirely the need for users to understand procedures. PROCGEN does not generate Procedure Language code for any of the following:

- ENTER statements for user programs, for utilities not listed above, or for default GETPARMs
- DISPLAY statements

- RENAME statements
- Testing and branching statements
- Return codes
- Labelling
- Logoffs

You must use the Editor to write code for these purposes.

Besides creating new procedure files, PROCGEN can be used to extend old ones. It can only append new code to the end of an existing file; more complex modifications require the use of the Editor. The menu includes a "free format" option, however, which allows you to add up to 24 lines of text to the file without having to use the Editor.

21.1.1 Software Requirements

PROCGEN assumes that the programs listed on its menu are present in the system library (@SYSTEM@). If they are elsewhere, the procedures it creates will have to be modified to reflect their actual locations.

21.1.2 Running PROCGEN

Run the PROCGEN utility from the Command Processor as you run any other program or utility. Enter PROCGEN into the program name field, the appropriate library name in the library field and the appropriate volume name in the volume field. (Since this utility does not reside in the system library, you must enter the library and volume designations.)

The following sections describe PROCGEN processing:

Section	Process
21.2	Specifying the Procedure File
21.2.1	Specifying a Temporary Procedure
21.3	Selecting Procedure Options

Section 21.4 presents a sample procedure created by PROCGEN, and Section 21.5 is a guide to documentation for the various procedure options PROCGEN offers.

21.2 SPECIFYING THE PROCEDURE FILE

When PROCGEN processing begins, the Output Definition screen appears. Figure 21-1 shows a sample of this screen.

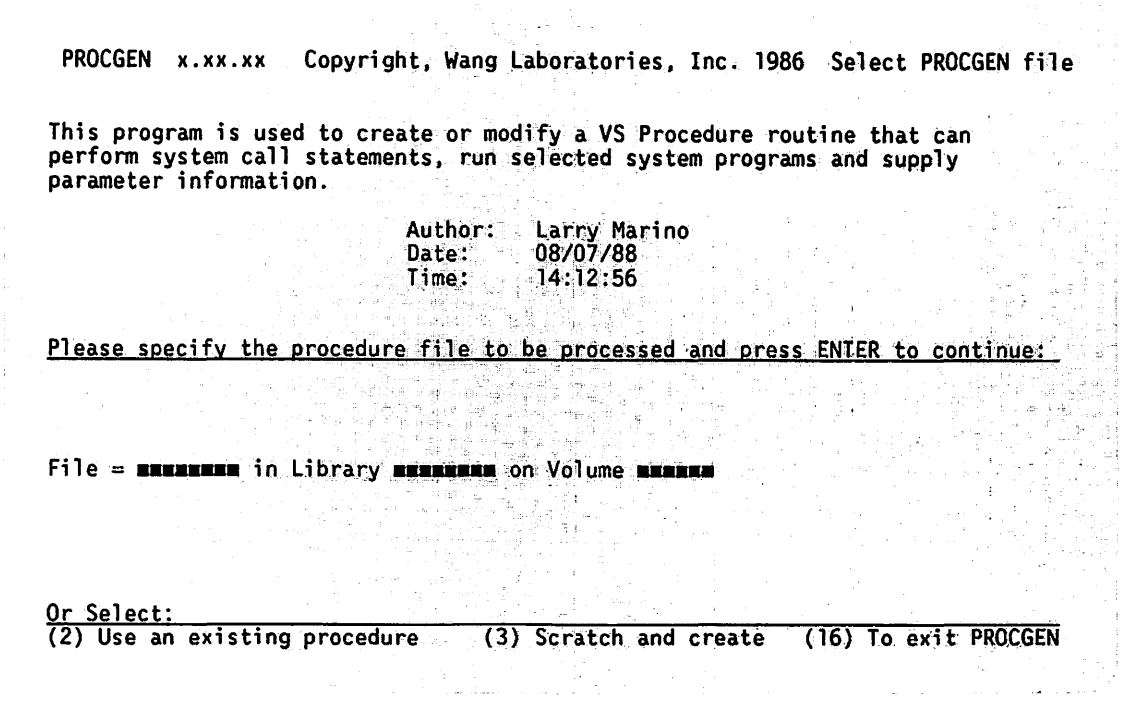


Figure 21-1. A Sample PROCGEN Output Definition Screen

The screen shows your name and the date and time. These are placed at the head of the procedure file as a comment. Immediately below, you are prompted to supply information as follows:

FILE -- If you are creating a new procedure, enter a valid name for the file. If you are extending an existing procedure, enter its name.

LIBRARY -- The library in which the procedure file resides, if it exists, or will reside, if it is to be created. If the INLIB field of your usage constants is set, its value appears here as the default.

VOLUME -- The volume on which the procedure file resides or will reside. If the INVOL field of your usage constants is set, its value appears here as the default.

Enter or change the values in these fields as appropriate, and press ENTER to continue. You also have these options from the Output Definition screen:

PF Key	Option and Description
2	**Use an existing procedure** -- Press PF2 to extend an existing procedure. (Its name and location must be correctly specified above.) PROCGEN adds the code it generates to the end of this procedure. Your name, the date, and the time appear at the end as comments.
3	**Scratch and create** -- If you have specified an existing procedure, but want the new procedure to replace it, press PF3 to scratch the named procedure and create a new one with the same name. (If you press ENTER after specifying a file that already exists, a warning appears on your screen, and you must either press PF3 or change the output file specification before you can proceed.)
16	**Exit PROCGEN** -- Press PF16 to terminate PROCGEN processing without creating any procedure code.

Newly created files assume the default output file protection class, and (for releases of the VS Operating System that support an access list) the default access list.

21.2.1 Specifying a Temporary Procedure

If you want to create a temporary procedure that will be destroyed when you finish using it, you can do so by supplying a name that begins with a pound sign (#), for the procedure file. If you enter a name beginning with # in the FILE field of the Output Definition screen, PROCGEN creates a temporary procedure routine with a system-supplied name in your work library and work volume. All temporary files are scratched when you log off the system or return to the Command Processor.

21.3 SELECTING PROCEDURE OPTIONS

When you have specified a procedure from the Output Definition screen and pressed either ENTER, PF2, or PF3 as appropriate, the Procedure Options menu (Figure 21-2) appears.

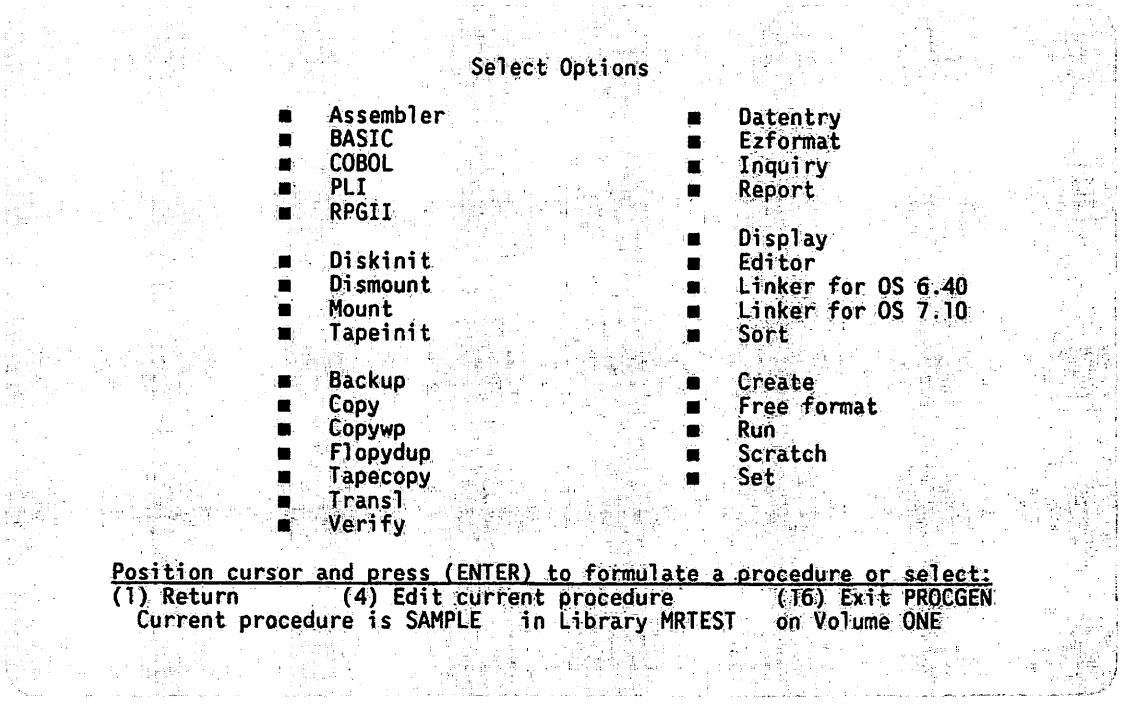


Figure 21-2. PROCGEN Procedure Options Menu

To generate Procedure Language statements for any of the calls or programs shown, move the cursor to the pseudoblank next to its name and press ENTER. PROCGEN simulates the program or call statement by displaying simulated GETPARM screens like those the program or call would display. The values you enter from these screens are embedded in ENTER clauses that follow the RUN statement for the program or call. The sequence of screens that appears depends on the individual program or call, and you should consult the appropriate documentation for information about supplying values. See Section 21.5 for references.

When you return (by means of the usual program exit procedure) from the simulated program to the PROCGEN Options menu, the box appears dimmed, but it remains available for use. You may use as many menu selections as you choose, repeating at will, in any order that makes sense.

Your other options from the Procedure Options menu are as follows:

PF Key	Option and Description
1	**Return** -- If you are finished with this procedure file and want to generate another, press PF1 to return to the Output Definition screen (Figure 21-1). The procedure you have created is saved, and the fields on the screen are cleared in the expectation that you will specify another file. If you want to scratch your first attempt and start again, simply enter the same file specifications and press PF3 from the Output Definition screen.
4	**Edit current procedure** -- If you want to use the Editor to modify the file you have generated, press PF4. (When the Editor is invoked in this way, its Restart and Utilities options are unavailable. See the *VS Editor Reference* for information on using the Editor. For VS Procedure Language syntax, see the *VS Procedure Language Reference*.)
16	**Exit PROCGEN** -- If you are finished with this procedure file and do not want to make another, press PF16. The file you have generated is saved.

21.4 A SAMPLE PROCGEN PROCEDURE

This section describes step-by-step the process of creating a simple procedure that runs the DATENTRY and REPORT utilities.

1. From the Output Definition screen, supply the file, library, and volume names for the procedure file. Press ENTER.

2. From the PROCGEN Options menu, position the cursor at the Datentry option and press ENTER.

3. From the DATENTRY Options screen, select PF3 (Add records to a data file).

4. Supply a path name if the file is indexed, and press ENTER.

5. From the PROCGEN Options menu, move the cursor to the Report option and press ENTER.

6. From the REPORT Function screen, select PF4 (Print a report).

7. From the REPORT Options screen, supply the Report ID (a file name), and press ENTER.

8. From the REPORT Print screen, supply the print file, library, and volume names, and press ENTER.

9. From the REPORT Function screen, select PF16, Exit REPORT.

10. From the PROCGEN Options menu, select PF16 (Exit PROCGEN).

If you follow these steps, PROCGEN generates a procedure file like the sample in Figure 21-3.

```
                PROCEDURE SAMPLE                                        00010
*                                                                       00020
************************************************************************00030
*                                                                      *00040
* AUTHOR: Larry Marino                                                 *00050
* CREATION DATE: 08/07/88                                              *00060
* CREATION TIME: 16:44:01                                              *00070
*                                                                      *00080
************************************************************************00090
*                                                                       00100
*                                                                       00110
        RUN DATENTRY                                                    00120
         ENTER INPUT                                                    00130
         FILE="TESTZ",      LIBRARY="MRTEST",   VOLUME="ONE",        00140
         CTLFILE="TESTZ",   CTLLIB="MRCTL",     CTLVOL="ONE"         00150
         ENTER OPTIONS            03                                    00160
         ENTER PATH                                                     00170
         PATH="FIELD1"                                                  00180
         ENTER OPTIONS 16                                               00190
         ENTER INPUT 16                                                 00200
*                                                                       00210
        RUN REPORT                                                      00220
         ENTER FUNCTION           04                                    00230
         ENTER OPTIONS                                                  00240
         ID="TESTRPT",      DEVICE="PRINTER",   FILES="NO",          00250
         OPTION="NO",       ONLY="NO",          PAGE="55",           00260
         LINES="123"                                                    00270
         ENTER PRINT                                                    00280
         FILE="TESTPRT",    LIBRARY="MRTEST",   VOLUME="ONE"         00290
         ENTER FUNCTION           16                                    00300
```

Figure 21-3. A Sample Procedure File Generated by PROCGEN

21.5 CALLS AND PROGRAMS

The following list shows all the call statements and programs available from PROCGEN's menu and the manual in which to find further information about each one.

Call Statements

DISMOUNT	*VS Procedure Language Reference*
MOUNT	
RUN	
SCRATCH	
SET	

Assemblers and Compilers

ASSEMBLER	*VS Assembly Language Reference*
BASIC	*VS BASIC Language Reference*
COBOL	*VS COBOL 85 Language Reference*
PLI	*VS PL/I Language Reference*
RPGII	*VS RPG II Language Reference*

Utility Programs

CREATE	Chapter 10 of this manual
DATENTRY	Chapter 3
DISPLAY	Chapter 11
EZFORMAT	Chapter 4
INQUIRY	Chapter 5
REPORT	Chapter 6
SORT	Chapter 14
TRANSL	Chapter 17
COPY	*VS Media, Transfer, and Device Utilities Reference*
COPYWP	
DISKINIT	
FLOPYDUP	
TAPEINIT	
TAPECOPY	
VERIFY	
BACKUP	*VS System Operator's Guide*
EDITOR	*VS Editor Reference*
LINKER	*VS Linker Reference*

CHAPTER 22
RPTGEN

22.1 INTRODUCTION

The RPTGEN (Report Generation) utility takes a Report Definition file (RDF) created by the REPORT utility and uses it to generate a COBOL source program. This program, when compiled and run, produces essentially the same results as the REPORT utility, but with distinct advantages for the programmer:

- The code can be customized to suit a particular purpose.
- It can be converted to a subroutine to be called or linked by another program.
- When printing large reports, a RPTGEN program can run as much as three times faster than REPORT.

22.1.1 Overview

RPTGEN begins by requesting the name of the Report Definition file, or RDF, which must exist before you can run RPTGEN. (If you need to create an RDF or modify an existing one, you can go directly to REPORT from the RPTGEN input screen.) After supplying the RDF, you next specify output options for the report to be produced, and finally a name and location for the COBOL source file that RPTGEN creates. If you choose, you can enter the Editor to work on this source file as soon as it is created. Information screens provide assistance for each step in the process.

22.1.2 Software Requirements

For RPTGEN to run properly, the REPORT utility and the VS Editor must reside in the system library.

As mentioned previously, RPTGEN requires an existing RDF, and this in turn requires for its creation the existence of a control file and a data file. For information about RDFs and the way REPORT is used to produce them, see Chapter 6. For the creation of control files through the CONTROL utility, see Chapter 2.

22.1.3 Running RPTGEN

Run the RPTGEN utility from the Command Processor as you run any other program or utility. Enter RPTGEN in the program name field, the appropriate library name in the library field, and the appropriate volume name in the volume field.

The following sections describe RPTGEN processing:

Section	Process
22.2	Specifying a Report Definition File
22.3	Specifying Report Options
22.4	Specifying the Source File

22.2 SPECIFYING A REPORT DEFINITION FILE

When RPTGEN processing begins, the RDF Specification screen (Figure 22-1) appears.

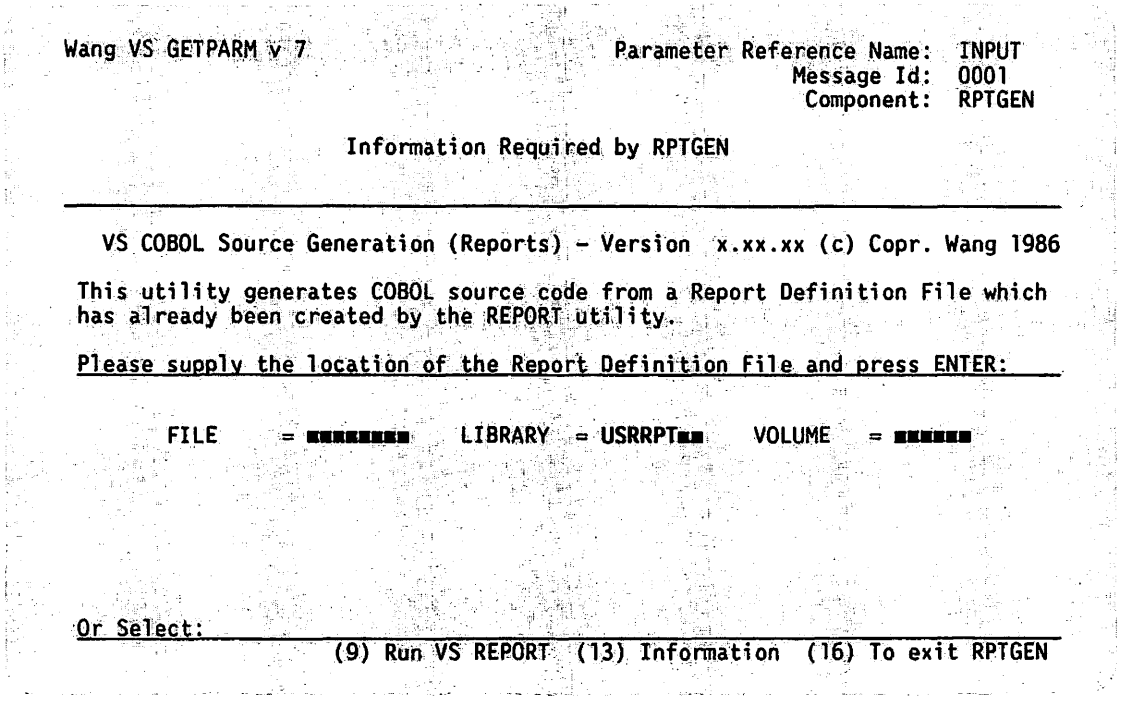


Figure 22-1. RPTGEN RDF Specification Screen

Use this screen to specify the RDF on which the report is to be based by supplying file information in the appropriate fields.

FILE -- Enter the name of an existing RDF, or a valid file name for the RDF you intend to create by running the REPORT utility.

LIBRARY -- RPTGEN supplies a default value by concatenating your user ID with the string *RPT* (e.g., USRRPT). This is the same default library used by REPORT.

VOLUME -- If the INVOL field of your user constants is set, the value from that field appears here as a default.

Enter or change the information in the fields as necessary and press ENTER to continue. You also have the following options from the RDF Specification screen:

- Press PF9 to run the REPORT utility. Do this if you intend to create a new RDF or modify an existing one. (If the RDF you have specified does not exist, you must create it through REPORT before you can continue.) When you terminate REPORT, you are returned to this screen.
- Press PF13 for general information about RPTGEN.
- Press PF16 to terminate RPTGEN without creating source code.

22.3 SPECIFYING REPORT OPTIONS

When you press ENTER from the RDF Specification screen, the Report Options screen (Figure 22-2) appears.

```
Wang VS GETPARM v 7                    Parameter Reference Name:  OPTIONS
                                                    Message Id:  0004
                                                     Component:  RPTGEN

                    Information Required by RPTGEN

 VS COBOL Source Generation (Reports) - Version  x.xx.xx (c) Copr. Wang 1986

 Please supply the following options and press ENTER to continue:

        PROGRAM  = ********  (The program name, any valid COBOL name)
        DEVICE   = PRINTER   (PRINTER/DISPLAY, Output device)
        COUNT    = NO        (YES/NO, Count option)
        LINES    = 55        (05 - 99, Lines per page)


 Or Select:
 (1) Return to INPUT                    (13) Information
```

Figure 22-2. RPTGEN Report Options Screen

Use the Report Options screen to name the program RPTGEN will generate and to specify other output options. Enter the information as follows:

PROGRAM -- This name (which must be a valid COBOL program name) is used by RPTGEN to identify the program it generates. The default value is the name of the RDF you specified.

DEVICE -- Your choice determines whether the reports generated by the COBOL program will be printed or displayed on the screen. The default is PRINTER.

COUNT -- If you want the reports to include the number of input records processed, change the default NO to YES.

LINES -- This field, the number of lines to a page, applies principally to printed reports, but it also affects the screen display. The default is 55, but you may enter any number from 5 to 99. A new heading is generated whenever this number of lines is exceeded, no matter which type of output you select. If the report is to be displayed, you may want to insure that the heading is repeated on every screen or that it always appears at the top of a screen. If so, you must calculate the number for this field accordingly.

When you have finished specifying options, press ENTER. You may also press PF13 for information about the option fields or PF1 to return to the Input Definition screen.

22.4 SPECIFYING THE SOURCE FILE

When you have specified the report options and pressed ENTER, the Source File Specification screen appears. Figure 22-3 shows an example of this screen.

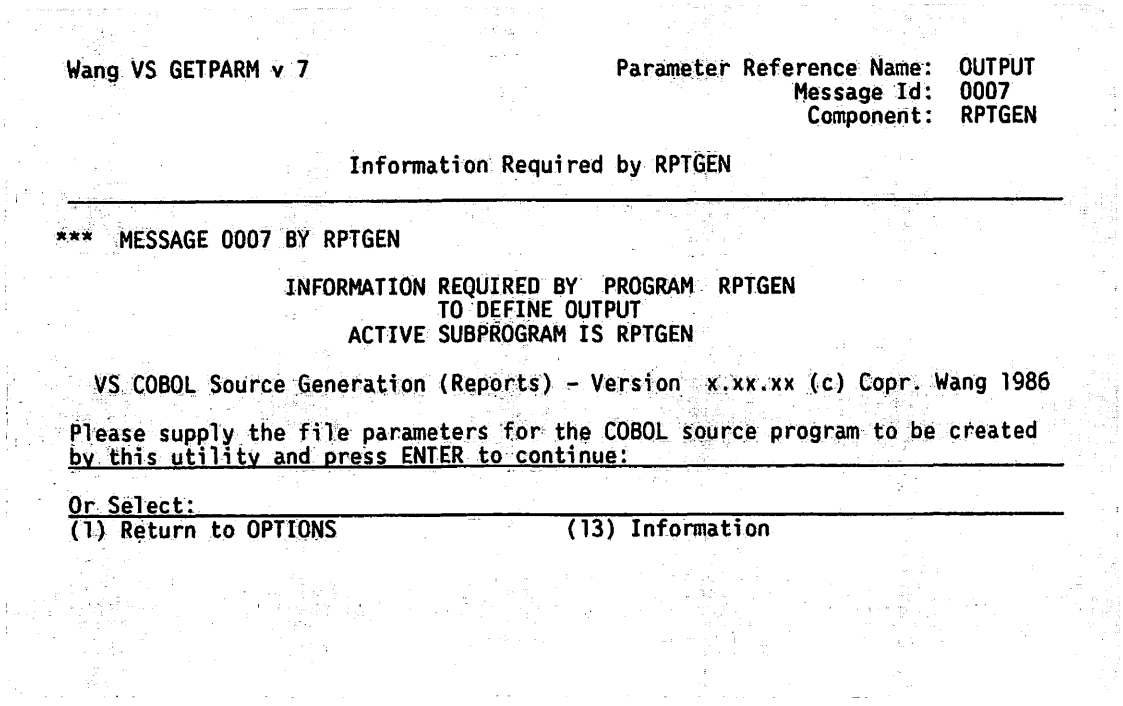


Figure 22-3. A Sample RPTGEN Source File Specification Screen

RPTGEN uses the file name, library, and volume you enter from this screen for the source file it generates. The file name default is the program name you entered from the Options screen. This name is used internally for the program, but it need not be the file name under which the source code is saved. You may enter any valid file name in its place. (It need not be a valid COBOL program name.)

The remaining field is EDIT, whose default value is YES. If you do not change this to NO, the Editor is set up to edit the source file as soon as it is created.

You can also press PF13 for information or PF1 to return to the Options screen.

When you press ENTER from the Source File Specification screen, RPTGEN creates the COBOL source file. If you specified YES in the EDIT field, the Editor menu appears next. See the *VS Editor Reference* for information on using the Editor.

Note: *The generated source code may contain calls to the LINK, PUTPARM, SORTCALL, or SCRATCH VSSUBS. (See the* VS VSSUBS Reference *for information about these subroutines.) In order to resolve the external references when compiling the source code, you must reference the VSSUBS library when you link the object code.*

The RPTGEN End-of-Job (EOJ) screen (not shown) appears when you exit from the Editor, or, if you have not chosen this option, as soon as the source file is created. It offers you the choice of pressing PF1 to rerun the program or pressing either PF16 or ENTER to terminate it.

APPENDIX A
FILE MANAGEMENT UTILITY GETPARMS

A.1 INTRODUCTION TO GETPARMS

The VS Operating System supports a supervisor call routine (SVC), the "GETPARM" SVC, which solicits and accepts runtime parameter information, and displays and awaits acknowledgment of runtime messages. GETPARM-generated prompts are displayed on the workstation screen during normal execution. These prompts solicit parameter information from the user or from a controlling procedure. Values entered from either source are verified for validity. If the values entered are not acceptable, the GETPARM SVC responds with an error message.

GETPARM processing is distinguished from other methods of obtaining runtime information primarily because GETPARM processing can interface with a procedure (see the *VS Procedure Language Reference*). A procedure is the preferred source of information for a GETPARM request; thus, GETPARM prompts never appear on the workstation screen when they are satisfied by a Procedure Language ENTER statement. Therefore, the interaction of the user with a program at runtime can be precisely controlled by the procedure writer. GETPARM requests are used wherever possible by the VS system programs to solicit parameter information. This enables the user to run system programs with little or no user interaction by supplying most or all of the required program parameters from procedures.

A.2 THE STRUCTURE OF A GETPARM

Each GETPARM request in a program is identified with a parameter reference name (*prname*). The prname for each request is, in general, unique within that program. System programs generally observe certain conventions when identifying GETPARM requests with prnames. For example, a GETPARM request soliciting information for an input file is usually identified with the prname INPUT, while a GETPARM soliciting parameters for an output file is named OUTPUT.

Many GETPARM requests for information contain one or more modifiable fields into which the user (or a procedure) can enter information. Each of these fields is labelled with a name called a *keyword*. When a GETPARM request is displayed, the keyword for each field provides a description of the information to be supplied for that field.

Again, certain conventions are commonly used in keyword naming; for example, a request for a file name often uses the keyword FILE. Also, many GETPARM requests solicit a PF key response (such as 16 = Exit Program). There is no keyword associated with a PF key choice; only the PF key number itself is specified.

A.3 ASSOCIATING A PROCEDURE WITH A GETPARM

Within a procedure, each ENTER or DISPLAY statement supplies parameters for a single GETPARM request. To associate a given ENTER or DISPLAY statement with a specific GETPARM request, the prname of the request must be specified in that statement. (GETPARM requests issued by a user program may, of course, be assigned any prname desired by the programmer. For user-defined GETPARM requests, modifiable fields and the keywords identifying them are specified by the GETPARM issuer.)

When parameters are supplied by a procedure, keywords must be used in the ENTER or DISPLAY statement to associate the specified values with the fields to which they are to be assigned in the GETPARM request. In this case, values associated with keywords in the procedure statement are passed to the corresponding keyword-identified fields in the GETPARM request. (Keywords must, of course, be correctly spelled in the procedure statement.) Fields that are not assigned new values retain their default values.

An example of a procedure that runs the DATENTRY utility is provided below. If your compare this procedure to the list of prnames and keywords for DATENTRY, you will see that certain default keyword values have been used, since those keywords are not listed (e.g., CTLLIB and CTLVOL for prname INPUT). Also, PF key options have been selected for certain other prnames (e.g., for prname OPTIONS: PF2, Create a Data File, was selected).

```
PROC RUN DATENTRY
RUN DATENTRY
ENTER INPUT FILE = TEST, LIBRARY = IDCTL,
 VOLUME = SYSTEM, CTLFILE = TEST
ENTER OPTIONS  2
ENTER OPTIONS 16
RETURN
```

Refer to the *VS Procedure Language Reference* for more information on the Procedure Language and the use of GETPARM requests.

A.4 FILE MANAGEMENT UTILITY GETPARM REQUESTS

The following pages list the prnames, keywords, options, and default values used by GETPARM requests for the VS File Management Utilities. The GETPARMS are listed by utility, and the utilities are organized alphabetically. Please note that when PF keys are used as options for a prname, no keyword is listed. Similarly, when the ENTER key is used as a PF key option, "Ъ" is listed; otherwise use of the ENTER key is assumed.

A.4.1 CONDENSE

prname	Keyword	Length	Options	Defaults
PARMFILE	PARMFILE	8		
	PARMLIB	8		User id&COND(USRCOND)
	PARMVOL	6		
	PF keys[a]		Ъ = Continue 16 = Exit CONDENSE	
FUNCTION	PF keys[a]		2 = Create parameter file 3 = Modify existing parameter file 4 = Create condensed data file 5 = Run REPORT 16 = Exit to respecify parameter file	
USEREXIT	FILE	8		
	LIBRARY	8		
	VOLUME	6		
CONTROL	CTLFILE	8		
	CTLLIB	8		User id&CTL(USRCTL)
	CTLVOL	8		
INPUT	INFILE	8		
	INLIB	8		
	INVOL	6		
OUTPUT	OUTFILE	8		
	OUTLIB	8		
	OUTVOL	6		

[a] The keyword is not required.

A.4.2 CONTROL

prname	Keyword	Length	Options	Defaults
LIMITED	FILE	8		
	LIBRARY	8		
	VOLUME	6		
	CTLFILE	8		
	CTLLIB	8		
	CTLVOL	6		
	MESSAGE	68		
CTLFIL	FILE	8		
	LIBRARY	8		User id&CTL(USRCTL)
	VOLUME	6		INVOL
	PF keys[a]		↵ = Continue	
			2 = Create new control file	
			9 = Print detail for all files	
			16 = EXIT	
OPTIONS	PF keys[a]		3 = Add new fields	
			4 = Modify existing fields	
			↑4 = Modify field header	
			5 = Delete fields	
			6 = List header and fields	
			7 = Create or maintain table fields	
			8 = Create source from the control file	
			12 = Modify the field sequence for DATENTRY	
			16 = Exit to list or respecify the control file	9 = Run DATENTRY
				10 = Run REPORT
				↑10 = Run INQUIRY
			↑16 = Exit CONTROL	11 = Run EZFORMAT
HEADER	RECLEN	4	1 - 2024 (variable/ compressed)	
			1 - 2040 (fixed/ relative)	
			1 - 2048 (consecutive)	

[a] The keyword is not required.

A.4.2 CONTROL (continued)

prname	Keyword	Length	Options	Defaults
HEADER	FILETYPE	F, V, C		
	FILEORG	C, I, R		
	KEYFIELD	8		
	USEREXIT	8	Ъ, USER1 - USER10	Ъ
	UPDATE	1	0, 1	0
	DELETE	1	0, 1	0
	REPORT	1	0, 1	
	OPENSHAR	1	Y, N	Y
	TIMEOUT	000 - 255	In seconds	000
	DEFAULT DATA			
	FILE	8		
	LIBRARY	8		
	VOLUME	6		
	COMMENT1	60		
	COMMENT2	60		
	COMMENT3	60		
	PF keys[a]		Ъ = Continue 16 = Exit to respecify the control file	

[a] The keyword is not required.

A.4.2 CONTROL (continued)

prname	Keyword	Length	Options	Defaults
SOURCE	PF keys[a]		1 = COBOL 3 = RPGII 5 = PLI 16 = Exit to CONTROL menu	
CPYLIB	FILE	8		
	LIBRARY	8		
	VOLUME	6		
	PF keys[a]		0 = Continue 16 = Exit	
PATHS	PATH1	8		1st alternate key
	PATH2	8		2nd alternate key
	.	.		
	.	.		
	.	.		
	PATH 16	8		16th alternate key
	DUPS1	3	YES, NO	NO
	.	.	.	.
	.	.	.	.
	.	.	.	.
	DUPS16	3	YES, NO	NO
	PF keys[a]		0 = Make changes 1 = Return to header screen 7 = Compress unused paths	
CTLSCR	DEVICE	7	SCREEN, PRINTER	SCREEN
	PF keys		0 = Continue 16 = Exit to CONTROL menu	

Note: *If you choose to display information on the screen instead of sending it to the printer, press PF6 (from the CONTROL central menu). Refer to Section 2.4.5.*

[a] The keyword is not required.

A.4.3 DATENTRY

prname	Keyword	Length	Options	Defaults
INPUT	FILE	8		
	LIBRARY	8		
	VOLUME	6		
	CTLFILE	8		
	CTLLIB	8		User id&CTL(USRCTL)
	CTLVOL	6		Input volume
	PF keys[a]			
			ƀ = Continue/list of files	
			2 = Create a new data file	
			16 = EOJ	
OPTIONS	PF keys[a]		3 = Add records	
			4 = Modify records	
			5 = Delete records	
			6 = Display data file	
			7 = Modify field display attributes	
			8 = Generate EZFORMAT screen	
			9 = Run INQUIRY	
			16 = Exit to respecify files	
			↑16 = Exit DATENTRY	
INDFILE	FILE	8		Data file name
	LIBRARY	8		Data file library
	VOLUME	6		Data file volume
	RECORDS	7		512
	RETAIN	3		
	RELEASE	3	YES, NO	NO
	FILECLAS	1	A - Z, ƀ, $, $	FILECLAS of usage constants
	DEVICE	11	DISK	DISK

A.4.3 DATENTRY (continued)

prname	Keyword	Length	Options	Defaults
PATH	PATH	8		
	PF keys[a]		Ø = Continue 1 = Corresponds to the primary key 1 - 16 = Corresponds to the alternate indexed keys ↑16 = Exit	
SPACING	SPACING	1,2		2
	PF keys[b]		Ø = Continue 16 = Exit	
FAC	ULINE	3	YES, NO	
	DISPLAY	7		
	UPLOW	3	YES, NO	
	PF keys[b]		Ø = Set display attributes or continue 1 = Return to spacing 2 = First field 4 = Previous field 5 = Next field 8 = Find field 16 = Exit to DATENTRY menu	

[a] See Section A.5.

[b] The keyword is not required.

A.4.4 EZFORMAT

prname	Keywork	Length	Options	Defaults
OPTIONS	LANGUAGE	10	ASSEMBLER, A, COBOL, C, BASIC, B, RPG, R, D, MENU, M	
	PROCEDUR	3	YES, NO	YES
	PF keys[a]		2 = New screen format 3 = Existing screen format 16 = Exit without generating format definition	
INSCREEN	FILE	8		
	LIBRARY	8		User id&SAVE(USRSAVE)
	VOLUME	6		
	DEVICE	11	DISK, NONE	DISK
FUNCTION	PF keys[a]		1 = Return to function selection without saving screen contents 2 = Save generated output only 3 = Save screen contents only 4 = Save screen contents and generated output 16 = Exit without saving any files	
SCREEN	FILE	8		
	LIBRARY	8		User id&SAVE(USRSAVE)
	VOLUME	6		
			0 = Continue	
CONTROL	FILE	8		
	LIBRARY	8		User id&CTL(USRCTL)
	VOLUME	6		
	PF keys[a]		0 = Continue 1 = Switch to COBOL option 2 = Invoke CONTROL 16 = Exit	

A.4.4 EZFORMAT (continued)

prname	Keyword	Length	Options	Defaults
COPYLIBR	FILE	8		
	LIBRARY	8		User id&COPY(USRCOPY)
	VOLUME	6		
PROGRAM	FILE	8		
	LIBRARY	8		
	VOLUME	6		
	PF keys[a]		0 = Continue 2 = Create field name file	
SOURCE	PF keys[a]	1 = COBOL 3 = RPG 16 = Exit		

[a] The keyword is not required.

A.4.5 INQUIRY

prname	Keywork	Length	Options	Defaults
INPUT	FILE	8		
	LIBRARY	8		
	VOLUME	6		
	MODE	5	INPUT, SHARED	INPUT
	CHANGE	3	YES, NO	NO
	OPTION	7	DISPLAY, EXTRACT	DISPLAY
	PF keys[a]		Ø = Continue 16 = Exit	
CONTROL	FILE	8		
	LIBRARY	8		User id&CTL(USRCTL)
	VOLUME	6		
	PF keys[a]		Ø = Continue 1 = Return to input data selection	
QUERY	DISPLAY		YES, NO	YES
	TITLE	40		
	LINE 1A	40		
	LINE 1B	39		
	LINE 2A	40		
	LINE 2B	39		
	LINE 3A	40		
	LINE 3B	39		
	LINE 4A	40		
	LINE 4B	39		
	LINE 5A	40		
	LINE 5B	39		
	LINE 6A	40		
	LINE 6B	39		
	LINE 7A	40		
	LINE 7B	39		
	PF keys[a]		(First Time) Ø = Continue 1 = Reselect query files 2 = Set lowercase on 14 = Review explanation 15 = View valid field names	

[a] The keyword is not required.

A.4.5 INQUIRY (continued)

prname	Keyword	Length	Options	Defaults
			(Second Time) 1 = Reselect query files 2 = Set lowercase on 14 = Review explanation 15 = View valid field names 16 = Continue	
OUTPUT	FILE	8		
	LIBRARY	8		
	VOLUME	6		
	RECORDS	7		No. of records in interrogated file
	CONSEC	3	YES, NO	NO
EOJ	PF keys[a]		1 = New query 2 = Save query 3 = Create RDF 16 = Exit	

[a] The keyword is not required.

A.4.6 REPORT

prname	Keyword	Length	Options	Defaults
FUNCTION	PFKEYS		2 = Define report 3 = Modify definition 4 = Print report 16 = EOJ	
CONTROL	FILE	8		Primary control file name
	LIBRARY	8		User id&CTL(USRCTL)
	VOLUME	6		Primary control file volume
CONTROL2	FILE	8		Secondary control file name
	LIBRARY	8		User id&CTL(USRCTL)
	VOLUME	6		Secondary control file volume
RPTDEF	FILE	8		Report Definition file name
	LIBRARY	8		Report Definition file library
	VOLUME	6		Report Definition file volume
INPUT1	FILE	8		Primary Data file name
	LIBRARY	8		Primary Data file library
	VOLUME	6		Primary Data file volume
	MODE	6	INPUT or SHARED	
INPUT2	FILE	8		Secondary Data file name
	LIBRARY	8		Secondary Data file library
	VOLUME	6		Secondary Data file volume
	MODE	6	INPUT or SHARED	
OPTIONS	ID	8		
	DATE	8		Current date
	DEVICE	7	PRINTER/DISPLAY	PRINTER
	FILES	3	YES or NO	
	OPTION	3	YES/NO or 001 - 999	
	ONLY	3	YES/NO	NO
	PAGE	2	05 - 99	55
	LINES	3	Combination of 1, 2, and 3	

A.4.6 REPORT (continued)

prname	Keyword	Length	Options	Defaults
	SPACING	3	Combination of 0 - 5.	000
	PF keys[a]		Ø = Continue 8 = Change ID defaults 14 = Explanations 16 = Return to REPORT menu	
PRINT[b]	(PRINT default GETPARM)			

[a] The keyboard is not required.

[b] Refer to Section A.5.

A.5 DEFAULT GETPARMS

Default GETPARMs are parameter requests that are not normally displayed to the user because the default GETPARMs are already supplied default values. These default values can be supplied either by the system, as in the case of a user's print library name, or by values specified in the Set Usage Constants function of the Command Processor. To change the default values, specify the appropriate prname and keywords in a procedure.

prname	Keyword	Length	Options	Defaults
OUTPUT	FILE	8		System generated
	LIBRARY	8		# logon id&WORK
	VOLUME	6		System disk
	RECORDS	7		Number of records
	RETAIN	3		0
	RELEASE	3	YES or NO	YES
	FILECLAS	1		#
	DEVICE	11	DISK, TAPE	DISK
PRINT	FILE	8		System generated
	LIBRARY	8		# logon id&PRT
	VOLUME	6		System disk
	RECORDS	7		Number of records
	RETAIN	3		0
	RELEASE	3		YES
	FILECLAS	1		#
	DEVICE	11	DISK, PRINTER, TAPE	DISK
	PRTCLASS	1		A
	FORM#	3		000

APPENDIX B
CONTROL FILE RECORD FORMATS

B.1 INTRODUCTION

The control file is an indexed file with the key in bytes 3 - 10 and a record length of 130 bytes. It has fixed-length records and is stored in a user-defined library or a default library whose name comprises user id&CTL(USRCTL).

The control file contains File Descriptor records, Alternate Key records, Comment records, and Field Descriptor records. A File Descriptor record consists of the File Name, File Type, Key Field, Report Code, Update Code, Delete Code, Record Length, User Exit (optional), and the Number of Alternate Keys. If an alternate key is specified, the Alternate Key record contains the Alternate Key Name and indicates whether or not this key allows duplicates. An optional Comment record consists of the File Description. A Field Descriptor record consists of the Field Name, Internal Format, Internal Length, Starting Position, Occurrences, Zero Suppress Code, Decimal Insert Code, Sign Control Code, Dollar/Comma Code, External Length, Report Code, Update Code, Decimal Positions Code, Binary Edit Code, Update Sequence, Validation Type, Table Name, Low Range, High Range, Packed Digits, Cumulative Field Name, Field Alias, Display Code, and Report Field Length.

B.2 FILE DESCRIPTOR RECORD

The format of the File Descriptor Record is as follows:

Byte Number	Field Name	Permissible Values
2	Not used	ASCII 0s
3 - 10	Header label	Must be "␣HEADER␣"
11	File type V - Variable length records C - Compressed records	F - Fixed-length records
12 - 19	Key field	Name of key field for indexed file. If blank, consecutive file assumed.
20	Report code	0 = Allowed 1 = Not allowed
21	Update code	0 = Allowed 1 = Not allowed
22	Deletion code	0 = Allowed 1 = Not allowed
23 - 26	Record length	Maximum record length of data file. (Length dependent upon file organization.)
27 - 29	Timeout clause	Timeout value in seconds (000 - 255). Used only when processing shared files.
30 - 37	User exit	Name of user exit subroutine, USER1 - USER10.

Byte Number	Field Name	Permissible Values
38 - 39	Number of alternate	Total number of alternate keys in data file. Numeric values, 0 - 16
40 - 47	Not used	
48	Data entry spacing 2 = Double spacing	1 = Single spacing
49	File organization	I = Indexed file R = Relative file C = Consecutive file Ø = Consecutive file
50	Default open mode	Ø = IO Y = Shared
51 - 58	Default data file	Default data file name
59 - 66	Default data library	Default data library
67 - 72	Default data volume	Default data volume
73 - 130	Unused	

B.3 ALTERNATE KEY RECORD

The format of the Alternate Key record is as follows.

Byte Number	Field Name	Permissible Values
1 - 2	Not used	
3 - 10	Record name	Must be "bKEY1bbb" for first record. Must be "bKEY2bbb" for first record.
11 - 18	Alternate key name 1	Valid field name
19	Allow duplicates 1	Y (YES) N (NO)

The above fields are repeated seven times, with their byte positions as follows:

20 - 27	Alternate key name 2
28	Allow duplicates 2
29 - 36	Alternate key name 3
	Allow duplicates 3
38 - 45	Alternate key name 4
	Allow duplicates 4

Byte Number	Field Name	Permissible Values
47 - 54	Alternate key name 5	
55	Allow duplicates 5	
56 - 63	Alternate key name 6	
64	Allow duplicates 6	
65 - 72	Alternate key name 7	
73	Allow duplicates 7	
74 - 81	Alternate key name 8	
82	Allow duplicates 8	
83 - 100	Not used	

Note: *There may be up to two of these records. The second record will reference alternate keys 9 through 16.*

B.4 COMMENT RECORD

The format of the Comment record(s) is as follows:

Byte Number	Field Name	Permissible Values
1 - 2	Not used	
3 - 10	Comment descriptor	Must be "!␣COMM␣", "!2␣COMM␣", or "!3␣COMM␣".
11 - 70	Comment	Alphanumeric data
71 - 130	Not used	

Note: *There may be up to three of these records.*

B.5 FIELD DESCRIPTOR RECORDS

The format of the Field Descriptor record is as follows:

Byte Number	Field Name	Permissible Values
1 - 2	Not used	
3 - 10	Field name	1- to 8-character field name. Must begin with alphabetic character and not contain any embedded blanks.
11	Internal format	B - Binary C - Character P - Packed decimal Z - Zoned decimal
12 - 14	Internal length	Number of bytes required for internal representation (1 - 132).
15 - 18	Starting position	Starting position of field. Must be less than record length.
19 - 20	Occurrences	Number of occurrences of this field (01 - 99).
21	Zero-suppress code (used by REPORT)	0 - No zero suppression 1 - Suppress leading zeros
22	Decimal insert code (used by REPORT)	0 - 9 - Number of decimal positions to print out for REPORT utility
23	Sign control code (used by REPORT)	0 - No sign control 1 - Trailing minus sign 2 - CR on minus (trailing) 3 - DB on minus (trailing)
24	Dollar/Comma Code (used by REPORT)	0 - No dollar or comma edit 1 - Comma edit for numeric field 2 - Dollar edit for numeric field 3 - Dollar and comma edit for numeric field

Byte Number	Field Name	Permissible Values
25 - 27	External length	1 - 132, calculated by CONTROL program from internal byte length, or entered by the user. If character field, external field length = number of disk positions. If zoned decimal field (format = Z), external field length = number of disk positions + 1 (for sign) + 1 (for decimal point if number of decimal positions not equal to 0). If packed decimal field (format = P, external field length - (number of disk positions * 2) + 1 (for decimal point if number of decimal positions not = 0).

Byte Number	Field Name	Permissible Values
25 - 27 (continued)	External length (continued)	If binary field (format = B), and binary edit code is equal to 0 (see below) for hexadecimal field; external field length = number of disk positions * 2. If binary field (format = B), and binary edit code is = 1 (see below), for conversion to binary field. If number of disk positions = 1, external field length = 2. If number of disk positions = 2, external field length = 3. If number of disk positions = 3, external field length = 5. If number of disk positions = 4, external field length = 8.
28	Report code	0 - This field can be reported on using REPORT. 1 - This field cannot be reported on using REPORT.
29	Update code	0 - This field can be updated using DATENTRY. 1 - This field cannot be updated using DATENTRY.
30	Decimal positions code	Number of decimal positions; numeric 0 - 9
31	Binary edit code	Used only if field format = B. 0 - This binary field is represented in hexadecimal. 1 - This binary field is represented as decimal.

Byte Number	Field Name	Permissible Values
32	Not used	
33 - 34	Update sequence	Sequence in which this field will (Used by DATENTRY) appear for formatted screen in DATENTRY.
35 - 36	Validation type	Validation criteria for field for DATENTRY. T - An external table is used. R - This field is checked for being within a range. CF - This field is accumulated in a predefined field. Both this field and the predefined field must be numeric (format = P or Z). DF - This field is a date field. DATENTRY will display the system date in the format (MMDDYY) for this field when records are added to the data file.

Byte Number	Field Name	Permissible Values
35 - 36 (continued)		GD - This field is a date field. DATENTRY will display the system date in the format (YYMMDD) for this field when records are added to the data file. DS - This field is a day field. DATENTRY will display the system date in the format (YYDDD) for this field when records are added to the data file. TS - This field is a time field. DATENTRY will display the system time in the format (HHMMSSHH) for this field when records are added to the data file. CS - This field is a creation date/time field. DATENTRY will display the system date and time in the format (YYMMDDHHMMSSHH) for this field when records are added to the data file. CM - This field is a modification date/time field. DATENTRY will display the system date and time in the format (YYMMDDHHMMSSHH) for the field when records are modified.
37 - 42	Table name	1- to 6-character name of external validation table for this field. Used only if validation type = Table.
43 - 58	Low range	Low range for this field. Used for range checking by DATENTRY. Used only if validation type = Range.
59 - 74	High range	High range for this field. Used for range checking by DATENTRY. Used only if validation type = Range.

Byte Number	Field Name	Permissible Values
75 - 76	Packed Digits	Number of packed digits (if internal format = P). This is the (number of disk positions * 2) - 1.
77 - 78	Cumulative field name	Name of cumulative object field. Used if validation type = Cumulative. The cumulative field name must be an existing numeric field defined on the control file.
85 - 115	Field alias	An alternate name for the fields to be used by the INQUIRY program.
116 - 117	Table field info	For Table fields: This contains the number of component fields within a single occurrence of the table. For Component fields: This contains the occurrence count for the Table field.
118 - 120	Table field info	For Table fields: This contains the starting location of the first Component field. For Component fields: This contains the length of a single occurrence of the table field. (A total of all components fields in a single occurrence.)
121 - 124	Not used	
125	Display code	0 - Pseudoblanks appear 1 - Data remains on screen 2 - New data added, but no modification

Byte Number	Field Name	Permissible Values
126	Report field length	Total number of field positions required for report format (includes external length, signs, symbols, etc.).
129	Default FAC character	Any valid field attribute Character (refer to the *VS Principles of Operation*).
130	Not used	

APPENDIX C
REPORT FILE RECORD FORMATS

C.1 INTRODUCTION

The Report Definition file is an indexed file with the key in bytes 1 - 13 and a record length of 87 bytes. It has compressed records and is stored in the library USERID + RPT (unless the user overrides this default).

The Report Definition file consists of

- A Header record
- A Data Files record (containing information concerning the input data file(s))
- A Control Files record (containing information about the control file(s))
- Sort records (containing the fields by which the data is sorted)
- A Control Fields record (containing control break information)
- Title Descriptor records
- Data Limits records (containing data selection criteria)
- New Field records (containing information on any new fields that are specified)
- Field Descriptor records (containing the printing specifications for individual fields)
- A User Exit record (containing the subroutine name and its file, library, and volume names)

Many of these records, or parts of them, are unnecessary for specific reports and, therefore, are not created. This is true of the Control Files record (file location may be obtained from the Data Files record, library may be obtained from IDCTL, and volume may be obtained from Set Usage Constants). Also unnecessary for specific reports are the Sort records (if no sorting is performed), the Control Fields record (if no control breaks are selected), some of the Title Descriptor records (only the first report title is required), and the New Field records (if no new fields are specified). Note also that if only part of a record is required, the remainder is blank.

C.2 HEADER RECORD

The format of the Header record is as follows:

Byte Number	Field Name	Permissible Values
1 - 13	Key to Header record	Must be "ƀƀHEADERƀƀƀƀƀ."
14	Secondary File Switch	1 - Secondary file is used 0 - No secondary file is used
15	Sort Switch	1 - Sorting performed 0 - No sorting performed
16	Data Limits Switch	1 - Data limits specified 0 - No data limits specified
17	Control Fields Switch	1 - Control fields used 0 - No control fields used
18	Column Subheadings Switch	1 - Column subheadings specified 0 - No column subheadings specified
19	New Fields Switch	1 - New fields specified 0 - No New fields specified
20 - 21	Print Line Length	Packed format, numeric value. Legal range 1 - 132.
22 - 23	Number of Fields	Packed format, numeric value. Legal range 1 - 40. Total number of fields chosen for REPORT (includes nonprinted fields).
24 - 25	Number of Print Fields	Packed format, numeric value. Legal range 1 - 40. Total number of fields selected for printing.
26 - 27	Second Print Line Length	Packed format, numeric value. Range 1 - 132.
28 - 29	Third Print Line Length	Packed format, numeric value. Range 1 - 132.
30	User Exit Switch	1 - User exit used 0 - No user exit used
31 - 87	Not used	

C.3 DATA FILES RECORD

The format of the Data Files record is as follows:

Byte Number	Field Name	Permissible Values
1 - 13	Key to Data Files record	Must be "ƀƀFILESƀƀƀƀ."
14 - 21	Primary Data File Name	1- to 8-character file name. Must begin with an alphabetic character and cannot contain any embedded blanks.
22 - 29	Primary Data File Library	1- to 8-character library name. Must begin with an alphabetic character and cannot contain any embedded blanks.
30 - 35	Primary Data File Volume	1- to 6-character volume name. Must be an existing volume name.
36 - 43	Secondary Data File Name	1- to 8-character file name. Must begin with an alphabetic character and cannot contain any embedded blanks.
44 - 51	Secondary Data File Library	1- to 8-character library name. Must begin with an alphabetic character and cannot contain any embedded blanks.
52 - 57	Secondary Data File Volume	1- to 6-character volume name. Must be an existing volume name.
58 - 65	Key to Secondary Data File	A field from the primary file to be used as the key to the secondary file. Any valid field name found in primary file.

Byte Number	Field Name	Permissible Values
66 - 67	Occurrences	Numeric value, legal range 1 - 99 or blank. If the occurrence of the field chosen from the primary file is greater than 1, then this value indicates the array subscript of the field to be used as the key from the primary file. Otherwise, this value is blank.
68 - 69	Not used	Always blank.
70 - 87	Not used	

C.4 CONTROL FILE RECORD

The format of the Control Files record is as follows:

Byte Number	Field Name	Permissible Values
1 - 13	Key to Control Files Record	Must be "ƀƀFILESCTLƀƀƀ."
14 - 21	Primary Control File Name	1- to 8-character file name. Must begin with an alphabetic character and cannot contain any embedded blanks.
22 - 29	Primary Control File Library	1- to 8-character library name. Must begin with an alphabetic character and cannot contain any embedded blanks.
30 - 35	Primary Control File Volume	1- to 6-character volume name. Must be an existing volume name.
36 - 43	Secondary Control File Name	1- to 8-character file name. Must begin with an alphabetic character and cannot contain any embedded blanks.
44 - 51	Secondary Control File Library	1- to 8-character library name. Must begin with an alphabetic character and cannot contain any embedded blanks.

Byte Number	Field Name	Permissible Values
52 - 57	Secondary Control File Volume	1- to 6-character volume name. Must be an existing volume name.
58 - 87	Not used	

C.5 SORT RECORDS

The format of Sort Record 1 is as follows:

Byte Number	Field Name	Permissible Values
1 - 13	Key to Sort Record 1	Must be "␣␣SORT1␣␣␣␣␣␣."
14 - 21	Field Name 1	1- to 8-character field name. Must begin with an alphabetic character and cannot contain any embedded blanks.
22 - 23	Occurrences 1	Numeric Value, legal range 1 - 99 or blank.
24	Order Code 1	A - Ascending sort order D - Descending sort order

These last three fields are repeated five more times as needed with their byte positions as shown below.

Byte Number	Field Name	Permissible Values
25 - 32	Field Name 2	
33 - 34	Occurrences 2	
35	Order Code 2	
36 - 43	Field Name 3	
44 - 45	Occurrences 3	
46	Order Code 3	
47 - 54	Field Name 4	

Byte Number	Field Name	Permissible Values
55 - 56	Occurrences 4	
57	Order Code 4	
58 - 65	Field Name 5	
66 - 67	Occurrences 5	
68	Order Code 5	
69 - 76	Field Name 6	
77 - 78	Occurrences 6	
79	Order Code 6	
80 - 87	Not used	

The format of Sort Record 2 is as follows:

Byte Number	Field Name	Permissible Values
1 - 13	Key to Sort Record 2	Must be "␣␣SORT2␣␣␣␣␣."
14 - 23	Field Name 7	1- to 8-character field name. Must begin with an alphabetic character and cannot contain any embedded blanks.
24 - 25	Occurrences 7	Numeric value, legal range 1 - 99 or blank.
26	Order Code 7	A - Ascending sort order D - Descending sort order

These field are an extension of the record, Sort Record 1. These last three fields are repeated once, with byte positions shown below.

Byte Number	Field Name	Permissible Values
27 - 34	Field Name 8	
35 - 36	Occurrences 8	
37	Order Code 8	
38 - 87	Not used	

C.6 CONTROL FIELDS RECORD

The format of the Control Fields record is as follows:

Byte Number	Field Name	Permissible Values
1 - 13	Key to Control Fields Record	Must be "ƀCNTRLƀFLDSƀƀ."
14 - 21	Field Name 1	1- to 8-character field name. Must begin with an alphabetic character and cannot contain any embedded blanks.
22 - 23	Occurrences 1	Numeric value, legal range 1 - 99 or blank.
24 - 25	Control Break Action Code 1	YE - Page eject follows control break. 0 - 99 - Numeric values: specifies number of lines to be skipped following control break.
26	Field Origin Code 1	1 - Primary file 2 - Secondary file 3 - New field
27	Control Field Repeat Code 1	1 - Control Field name repeated on every data line. 0 - Control Field name printed only on first data line of Control Break section.

These last five fields are repeated four more times with byte positions shown below.

Byte Number	Field Name	Permissible Values
28 - 25	Field Name 2	
36 - 37	Occurrences 2	
38 - 39	Control Break Action Code 2	

Byte Number	Field Name	Permissible Values
40	Field Origin Code 2	
41	Control Field Repeat Code 2	
42 - 49	Field Name 3	
50 - 51	Occurrences 3	
52 - 53	Control Break Action Code 3	
54	Field Origin Code 3	
55	Control Field Repeat Code 3	
56 - 63	Field Name 4	
64 - 65	Occurrences 4	
66 - 67	Control Break Action Code 4	
68	Field Origin Code 4	
69	Control Field Repeat Code 4	
70 - 77	Field Name 5	
78 - 79	Occurrences 5	
80 - 81	Control Break Action Code 5	
82	Field Origin Code 5	
83	Control Field Repeat Code 5	
84 - 87	Not used	

C.7 TITLE DESCRIPTOR RECORDS

The format of the Title Descriptor records is as follows:

Byte Number	Field Name	Permissible Values
1 - 13	Key to Title Records	The three types of title have different key values: Report titles: "Ъ H1-TITLEЪЪЪЪ" "Ъ H2-TITLEЪЪЪЪ" "Ъ H3-TITLEЪЪЪЪ" Page titles: "Ъ H4-PAGEHЪЪЪЪ" "Ъ H5-PAGEHЪЪЪЪ" "Ъ H6-PAGEHЪЪЪЪ" Control break descriptors: "ЪHC1CNTRLЪЪЪЪ" "ЪHC2CNTRLЪЪЪЪ" "ЪHC3CNTRLЪЪЪЪ" "ЪHC4CNTRLЪЪЪЪ" "ЪHC5CNTRLЪЪЪЪ"
14 - 73		For report titles and page titles: 60-character alphanumeric strings. For control break descriptors: 40-character alphanumeric strings, followed by a 3-character numeric value for the Control Break Descriptor column position, followed by an unused space.

Byte Number	Field Name	Permissible Values	
		Bytes	**Use**
		14 - 53	40-character string
		54 - 56	3-character string
		57 - 87	Not used
74 - 87	Not used		

Note: *There may be up to 11 title records with all of the possibilities listed above (only the first is required). The key names given above correspond to the three types of title noted above and have different title values in byte positions 14 - 73 as shown.*

C.8 DATA LIMITS RECORDS

For a given report, it is possible to have up to 10 sets of data limits (designated by the values C - L in the Record Type Indicator field, byte position 1). Up to four records may exist for each Record Type Indicator (i.e., each field), depending on the number of fields/constants included in the data limit specifications. The first three records contain three fields and/or constants, and the last record contains only one. The format of the Data Limits records is as follows:

Byte Number	Field Name	Permissible Values
1	Record Type Indicator	Must be in the range C - L. The data limits sets begin with the letter C, and continue in alphabetic order up to L.
2 - 9	Field Name	1- to 8-character field name. Must begin with an alphabetic character and cannot contain embedded blanks.
10 - 11	Occurrences	Numeric value, legal range 1 - 99, or blank.
12	Field Origin Code	Indicates origin of field for which data limits are specified. 1 - Primary file 2 - Secondary file 3 - New field
13	Record Number	Numeric value, legal range 0 - 3. Identifies each of the four records for a given Record Type Indicator value (i.e., data field).
14 - 15	Operand Code 1	Operational connector between Field/Constants. Six options: GT - Greater than LT - Less than EQ - Equal to GE - Greater than or equal to LE - Less than or equal to NE - Not equal to

The next 20 bytes may contain two different types of information, depending upon the value in the field "Literal Indicator" (in byte position 35). If the Literal Indicator value is X, then byte positions 16 - 34 contain the information for a field that is to be used in the Data Limits record. If the Literal Indicator value is not X, then the information in bytes 16 - 35 is assumed to be literal input to the Data Limits record. These two types appear as follows:

Byte Number	Field Name	Permissible Values
	Field Information	
16 - 23	Field Name 1	Name of first field to be used in the Data Limits record. Valid existing file name.
24 - 27	Occurrences 1	Numeric value, legal range 01 - 99, in parentheses (e.g., "(01)").
28	Field Origin Code 1	1 - Primary file 2 - Secondary file 3 - New field
29 - 34	Not used	
35	Literal Indicator 1	Must be "X" to indicate that the above is field information.
	Literal Information	
14 - 33	Literal Input	Byte position 35 must *not* be "X" to indicate literal information. Character: Input delimited by double quotation marks. Numeric: Maximum 15 digits (includes sign, if used, plus decimal point).

The rest of the record is as follows:

36	Logical Connector 1	Logically connects operands. Two options: A - And O - Or

These last seven fields (bytes 16 - 36) are repeated twice with their byte positions as shown.

Byte Number	Field Name	Permissible Values
37 - 38	Operand Code 2	
39 - 46	Field Name 2	
47 - 50	Occurrences 2	
51	Field Origin Code 2	
52 - 57	Not used	
58	Literal Indicator 2	
59	Logical Connector 2	
60 - 61	Operand Code 3	
62 - 69	Field Name 3	
70 - 73	Occurrences 3	
74	Field Origin Code 3	
75 - 80	Not used	
81	Literal Indicator 3	
82	Logical Connector 3	

The end of the record is as follows:

Byte Number	Field Name	Permissible Values
83	Data Limits Set Connector	Logically connects set a of data limits with its subsequent set. This field occurs only for the first of the four possible records for a given "Record Type Indicator" value (e.g., when the "Record Number" Value is "0"). There are two options: A - And O - Or
84 - 87	Not used	

C.9 NEW FIELD RECORDS

The format of the New Field records is as follows:

Byte Number	Field Name	Permissible Values
1	Record Type Indicator	Must be "R".
2 - 9	New Field Name	1- to 8-character field name. Must begin with an alphabetic character and cannot contain embedded blanks.
10 - 11	Record Number	Up to four records may exist in the definition of a New Field, the first three of which contain three fields and/or literals, and the last of which contains only one. These records are identified by their Record Number as follows: 00 - First record 01 - Second record 02 - Third record 03 - Fourth record
12 - 13	New Field Number	Identifies the sequence in which the New Fields were created. Packed format numeric value, legal range 0 - 9.

The following 20 bytes may contain two different types of information, depending on the value in the field "Literal Indicator" (in byte position 33). If the Literal Indicator value is X, then byte positions 14 - 32 contain the information for a field that is to be used in calculating the New Field. If the Literal Indicator value is not X, then the information in bytes 14 - 33 is assumed to be literal input to the New Field. These two modes appear as follows:

Field Information

14 - 21	Field Name 1	Name of first field to be used in creating New Field. Valid existing field name (may be a previously defined New Field).

Byte Number	Field Name	Permissible Values
22 - 25	Occurrences 1	Numeric value, legal range 1 - 99, in parentheses (e.g., "(01)"), or blank.
26	Field Origin Code 1	1 - Primary file 2 - Secondary file 3 - New field
27 - 32	Not used	
33	Literal Indicator 1	Must be "X" to indicate above field information.

Literal Information

14 - 33	Literal Input	Byte position 33 must *not* be "X" to indicate literal information. There are two types of literal input: Character: Input delimited by double quotation marks. Numeric: Maximum 15 digits (includes sign, if used), plus decimal point.

The rest of the record is as follows:

Byte Number	Field Name	Permissible Values
34	Operation 1	Indicates operation to be performed between already specified value (in bytes 14 - 33) and the next value to be specified. *Options:* & - Concatenates character information +,-,/,* - For numeric information Blank - Indicates end of creation sequence

These last six fields (bytes 14 - 34) are repeated either once or twice, as needed, with their byte positions as shown below.

Byte Number	Field Name	Permissible Values
35 - 42	Field Name 2	
43 - 46	Occurrences 2	
47	Field Origin Code 2	
48 - 53	Not used	
54	Literal Indicator 2	
55	Operation 2	
56 - 63	Field Name 3	
64 - 67	Occurrences 3	
68	Field Origin Code 3	
69 - 74	Not used	
75	Literal Indicator 3	
76	Operation 3	

The end of the record (i.e., information in bytes 77 - 81) is used only for the first of the four possible records (e.g., when the Record Number Value is 00).

Byte Number	Field Name	Permissible Values
77	New Field Type	C = Character P = Numeric
78 - 79	Internal New	Packed numeric value. Computed Field Length internally based upon specified New Field Length. *Two types:* Numeric: Always eight bytes (for maximum calculation precision). Character: 1 - 132 byte range (same as specified length)

Byte Number	Field Name	Permissible Values
80	Decimal Positions	If New Field Type = P, numeric value, legal range 0 - 9. If New Field Type = C, blank.
81 - 87	Not used	

C.10 FIELD DESCRIPTOR RECORD

The format of the Field Descriptor record is as follows:

Byte Number	Field Name	Permissible Values
1	Record Type Indicator	Indicates on which print line the field is to appear. Z indicates line 1, Y indicates line 2, and X indicates line 3.
2 - 3	Field Sequence	Numeric values 1 - 40, 98 and 99
4 - 11	Field Name	1- to 8-character file name. Must begin with an alphabetic character and cannot contain any embedded blanks.
12 - 13	Occurrences	Any subscript number. Numeric values 1 - 99, or blank if not part of an array.
14 - 38	Column Heading	Alphanumeric
39 - 63	Column Subheading	Alphanumeric
64 - 65	Spaces Before	Packed format; numeric values fields 0 - 99.
66 - 67	External Size	Packed format; numeric values. Defaults to same value as CONTROL External Field Length.

Note: *Bytes 68 - 84 are used for numeric fields only.*

Byte Number	Field Name	Permissible Values
68 - 69	Zero Suppress Code	Blank - No zero suppression *n - Asterisk protection where n = 0 - 9 Zn - Zero suppression where n = 0 - 9
70 - 71	Sign Control Code	Blank - No signs used 9 - Trailing minus sign CR - CR on minus (trailing) DB - DB on minus (trailing)
72	Decimal Carry	Numeric values 0 - 9 or blank.
73 - 78	Special Insertions 1, 2, and 3	Allows insertion of three special characters: nx, where n = 1 - 9 and x = ".", ",", "/", "*", "-". and blank.
79	Dollar Sign Code	0 - No dollar signs 1 - Floating dollar signs 2 - Fixed dollar signs
80	Comma Code	0 - No commas 1 - Commas inserted
81	Total Code	X - Totals printed at control breaks at end of report b - Totals not printed
82	Maximum Code	X - Maximum values printed blank b - Maximums not printed
83	Minimum Code	X - Minimum values printed blank b - Minimums not printed
84	Average Code	X - Average values printed blank b - Averages not printed
85	File Type or New Field Code	1 - Primary file 2 - Secondary file 3 - New field
86 - 87	Not used	ASCII Os

APPENDIX D
RETURN CODES

All VS programs and some Procedure Language statements generate return codes during execution. These return codes indicate the reasons for a program failure, or warn of conditions that could cause program failure. A successful program execution generates a return code of 0, which is transparent to the user. The return codes are listed below according to program.

Program Name	Return Code	Meaning
All compilers	1 - 4	Warning
	5 - 8	Severe error. Program will not execute correctly.
	9 - 16	Fatal error. Program will not execute.
EZFORMAT	01 - 04	Warning. Program will probably run.
	05 - 07	Severe error. Program will not execute correctly.
	8 - 16	Fatal error. Program will not execute.

INDEX

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

Z

WANG

Customer Comment Form

Publication Number 715-1715

Title VS FILE MANAGEMENT AND APPLICATION DEVELOPMENT UTILITIES

Help Us Help You . . .

We've worked hard to make this document useful, readable, and technically accurate. Did we succeed? Only you can tell us! Your comments and suggestions will help us improve our technical communications. Please take a few minutes to let us know how you feel.

How did you receive this publication?

- ☐ Support or Sales Rep
- ☐ Wang Supplies Division
- ☐ From another user
- ☐ Enclosed with equipment
- ☐ Don't know
- ☐ Other _____

How did you use this Publication?

- ☐ Introduction to the subject
- ☐ Classroom text (student)
- ☐ Classroom text (teacher)
- ☐ Self-study text
- ☐ Aid to advanced knowledge
- ☐ Guide to operating instructions
- ☐ As a reference manual
- ☐ Other _____

Please rate the quality of this publication in each of the following areas.

	EXCELLENT	GOOD	FAIR	POOR	VERY POOR
Technical Accuracy — Does the system work the way the manual says it does?	☐	☐	☐	☐	☐
Readability — Is the manual easy to read and understand?	☐	☐	☐	☐	☐
Clarity — Are the instructions easy to follow?	☐	☐	☐	☐	☐
Examples — Were they helpful, realistic? Were there enough of them?	☐	☐	☐	☐	☐
Organization — Was it logical? Was it easy to find what you needed to know?	☐	☐	☐	☐	☐
Illustrations — Were they clear and useful?	☐	☐	☐	☐	☐
Physical Attractiveness — What did you think of the printing, binding, etc?	☐	☐	☐	☐	☐

Were there any terms or concepts that were not defined properly? ☐ Y ☐ N If so, what were they? _____

After reading this document do you feel that you will be able to operate the equipment/software? ☐ Yes ☐ No ☐ Yes, with practice

What errors or faults did you find in the manual? (Please include page numbers) _____

Do you have any other comments or suggestions? _____

Name _____ **Street** _____

Title _____ **City** _____

Dept/Mail Stop _____ **State/Country** _____

Company _____ **Zip Code** _____ **Telephone** _____

Thank you for your help.

Printed in U.S.A. 14-3140A 2-88

WANG

Cut along dotted line

Fold

NO POSTAGE
NECESSARY IF
MAILED IN THE
UNITED STATES

WANG

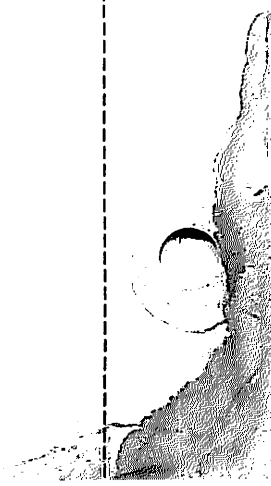
BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 16 LOWELL, MA

POSTAGE WILL BE PAID BY ADDRESSEE

Wang Laboratories, Inc.
Technical Publications Dept.
M/S 012-260
One Industrial Avenue
Lowell, Massachusetts 01851-9971

Fold



WANG

To Order by Phone, Call:
1-(800)225-0234
Telex 172108

Order Form for Wang Manuals and Documentation

① Customer Number (If Known)

② Bill To: | Ship To:

③ Customer Contact: ( ) ( ) Phone Name | ④ Date | Purchase Order Number

⑤ Taxable Yes ☐ No ☐ | ⑥ Tax Exempt Number | ⑦ Credit This Order to A Wang Salesperson Please Complete | Salesperson's Name | Employee No. | RDB No.

⑧ Document Number	Description	Quantity	⑨ Unit Price	Total Price

⑩ Authorized Signature | Date

Sub Total	
Less Any Applicable Discount	
Sub Total	
LocalStateTax	
Total Amount	

☐ Check this box if you would like a free copy of
WangDirect Software & Literature Catalog
(711-0888A)

Ordering Instructions

1. If you have purchased supplies from Wang before, and know your Customer Number, please write it here.
2. Provide appropriate Billing Address and Shipping Address.
3. Please provide a phone number and name, should it be necessary for WANG to contact you about your order.
4. Your purchase order number and date.
5. Show whether order is taxable or not.
6. If tax exempt, please provide your exemption number.
7. If you wish credit for this order to be given to a WANG salesperson, please complete.
8. Show part numbers, description and quantity for each product ordered.
9. *Pricing extensions and totaling can be completed at your option; Wang will refigure these prices and add freight on your invoice.*
10. Signature of authorized buyer and date.

Wang Terms and Conditions

1. **TAXES** — Prices are exclusive of all sales, use, and like taxes.
2. **DELIVERY** — Delivery will be F.O.B. Wang's plant. Customer will be billed for freight charges; and unless customer specifies otherwise, all shipments will go best way surface as determined by Wang. Wang shall not assume any liability in connection with the shipment nor shall the carrier be construed to be an agent of Wang. If the customer requests that Wang arrange for insurance the customer will be billed for the insurance charges.
3. **PAYMENT** — Terms are net 30 days from date of invoice. Unless otherwise stated by customer, partial shipments will generate partial invoices.
4. **PRICES** — The prices shown are subject to change without notice. Individual document prices may be found in the WangDirect Software & Literature Catalog (711-0888A)
5. **LIMITATION OF LIABILITY** — In no event shall Wang be liable for loss of data or for special, incidental or consequential damages in connection with or arising out of the use of or information contained in any manuals or documentation furnished hereunder.

Printed in U.S.A. 14-3141A 2-88

WANG

Cut along dotted line

Fold

NO POSTAGE
NECESSARY IF
MAILED IN THE
UNITED STATES

WANG

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 16 LOWELL, MA

POSTAGE WILL BE PAID BY ADDRESSEE

WangDirect
Wang Laboratories, Inc.
M/S 017-110
800 Chelmsford Street
Lowell, Massachusetts 01851-9972

Fold

WANG

ONE INDUSTRIAL AVENUE, LOWELL, MA 01851
TEL. (508) 459-5000, TELEX 172108

Printed in U.S.A. 715-1715 11-88