900 Series HP 3000 Computer Systems

SORT-MERGE/XL
General Users Guide

HP Part No. 32650-90082
Printed in U.S.A. 1994

Third Edition
E0494

Printing History

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The date on the title page and back cover of the manual changes only when a new edition is published. When an edition is reprinted, all the prior updates to the edition are incorporated. No information is incorporated into a reprinting unless it appears as a prior update.

First Edition	November 1987	A.01.00
Second Edition	July 1988	A.01.00
Third Edition	April 1994	C.50.00

There are many other manuals applicable to the 900 Series HP 3000. The *MPE/iX Documentation Guide & Glossary Of Terms* (5958-9511) contains a complete list of all MPE/iX manuals.

Preface

SORT-MERGE/XL is a subsystem of the MPE/iX operating system on the 900 Series HP 3000. It allows you to sort data in files, based on one or more data items. You can also merge two or more sorted files into a single, new merged file.

This guide is written for general and experienced users. It introduces how to use SORT-MERGE/XL in both interactive and batch job modes of operation. It includes a reference section on the commands used to specify what will be sorted or merged. If you are interested in information on using this subsystem programmatically, refer to the *SORT-MERGE/XL Programmer's Guide* (32650-90080).

This guide is part of the General User's Series of manuals. See the documentation map at the front of this guide for a description of how it relates to the other manuals in the series.

Organization of this Manual

To help you find the information you need, a brief description of each chapter and appendix in the guide follows:

Chapter 1	**Introduction to SORT-MERGE/XL** is an overview of how and what you can sort or merge, and an explanation of the basic procedure.
Chapter 2	**Getting Started With SORT-MERGE/XL** is an introduction to commands, key data items, collating sequences, translation tables, and the types of files used by the subsystem.
Chapter 3	**Using SORT/XL Interactively** is a discussion on how to perform various sort functions in an interactive session.
Chapter 4	**Using MERGE/XL Interactively** is a discussion on how to perform various merge functions in an interactive session.
Chapter 5	**Using SORT-MERGE/XL in Batch Mode** is a discussion on how to build a job file, begin its operation, schedule it for processing, and terminate it, if necessary.
Chapter 6	**SORT-MERGE/XL Commands** is a reference section for SORT-MERGE/XL commands, including options, parameters, operation, and in most cases, examples.
Appendix A	**Error Messages** is a listing of all SORT-MERGE/XL subsystem error messages.
Appendix B	**ASCII/EBCDIC Tables** contains ASCII/EBCDIC tables showing codes values in character, decimal, octal, and hexadecimal formats.
Appendix C	**Native Language Collating** is a listing of native languages for which collating is available on the 900 Series HP 3000.
Glossary	A listing of terms and definitions used in this manual.

How to Use this Manual

If you are new to the SORT-MERGE/XL subsystem you should read Chapters 1 and 2 first. If you are an experienced user of SORT-MERGE/XL, turn to Chapters 2, 3, and 4 for task-oriented discussions on performing various functions with the subsystem. If you require specific information on SORT-MERGE/XL commands, turn to the reference section in Chapter 6.

In addition to this guide, you might also find the following sources of information useful:

General User's Reference Manual (32650-90002)

HP 3000 Guide for the New User (32033-90009)

Migration Process Guide (30367-90007)

MPE V to MPE/iX: Getting Started (30367-90002)

SORT-MERGE/XL Programmer's Guide (32650-90080)

Conventions

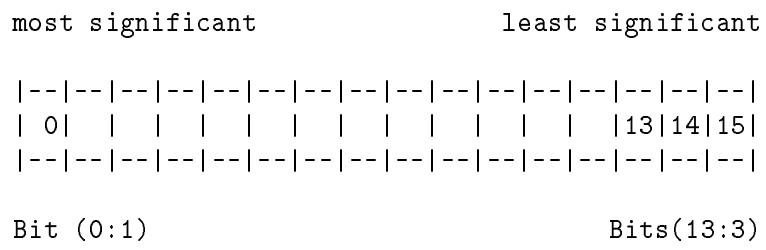
NOTATION	DESCRIPTION
`UPPERCASE`	Within syntax statements, characters in uppercase must be entered in exactly the order shown, though you can enter them in either uppercase or lowercase. For example: `SHOWJOB` Valid entries: `showjob` `ShowJob` `SHOWJOB` Invalid entries: `shojwob` `ShoJob` `SHOW_JOB`
italics	Within syntax statements, a word in italics represents a formal parameter or argument that you must replace with an actual value. In the following example, you must replace *filename* with the name of the file you want to release: `RELEASE` *filename*
punctuation	Within syntax statements, punctuation characters (other than brackets, braces, vertical parallel lines, and ellipses) must be entered exactly as shown.
`{ }`	Within syntax statements, braces enclose required elements. When several elements within braces are stacked, you must select one. In the following example, you must select `ON` or `OFF`: `SETMSG` { `ON` / `OFF` }
`[ ]`	Within syntax statements, brackets enclose optional elements. In the following example, brackets around `,TEMP` indicate that the parameter and its delimiter are optional: `PURGE {`*filename*`} [,TEMP]` When several elements with brackets are stacked, you can select any one of the elements or none. In the following example, you can select *devicename* or *deviceclass* or neither: `SHOWDEV` [*devicename* / *deviceclass*]

NOTATION	DESCRIPTION

`[ ... ]`	Within syntax statements, a horizontal ellipsis enclosed in brackets indicates that you can repeatedly select elements that appear within the immediately preceding pair of brackets or braces. In the following example, you can select *itemname* and its delimiter zero or more times. Each instance of *itemname* must be preceded by a comma:
	`[,`*itemname*`][ ... ]`
	If a punctuation character precedes the ellipsis, you must use that character as a delimiter to separate repeated elements. However, if you select only one element, the delimiter is not required. In the following example, the comma cannot precede the first instance of *itemname*:
	`[`*itemname*`][, ... ]`

`| ... |`	Within syntax statements, a horizontal ellipsis enclosed in parallel vertical lines indicates that you can select more than one element that appears within the immediately preceding pair of brackets or braces. However, each element can be selected only one time. In the following example, you must select `,A` or `,B` or `,A,B` or `,B,A` : {`,A` / `,B`}`| ... |` If a punctuation character precedes the ellipsis, you must use that character as a delimiter to separate repeated elements. However, if you select only one element, the delimiter is not required. In the following example, you must select `A` or `B` or `AB` or `BA`. The first element cannot be preceded by a comma: {`A` / `B`}`|, ... |`
...	Within examples, horizontal or vertical ellipses indicate where portions of the example are omitted.
␣	Within syntax statements, the space symbol ␣ shows a required blank. In the following example, you must separate *modifier* and *variable* with a blank: `SET[(`*modifier*`)]␣(`*variable*`);`
`shading`	Within an example of interactive dialog, `shaded` characters indicate user input or responses to prompts. In the following example, `GATO` is the user's response to the `NEW NAME` prompt: `NEW NAME? GATO`

NOTATION	DESCRIPTION
KEY	The symbol KEY indicates a key on the terminal's keyboard. For example, CTRL indicates the Control key.
CTRL *char*	CTRL *char* indicates a control character. For example, CTRL Y means you have to simultaneously press the Control key and the Y key on the keyboard.
base prefixes	The prefixes **%**, **#**, and **$** specify the numerical base of the value that follows: **%***num* specifies an octal number. **#***num* specifies a decimal number. **$***num* specifies a hexadecimal number. When no base is specified, decimal is assumed.
Bit (*bit:length*)	When a parameter contains more than one piece of data within its bit field, the different data fields are described in the format Bit (*bit:length*), where *bit* is the first bit in the field and *length* is the number of consecutive bits in the field. For example, `Bits (13:3)` indicates bits 13, 14, and 15:



Contents

Figures

Tables

1

Introduction To SORT-MERGE/XL

SORT-MERGE/XL is a subsystem of the MPE XL operating system for the 900 Series HP 3000. The SORT/XL portion of the subsystem allows you to sort files, based on one or more data items, into a specified alphabetical, numerical, or alphanumerical order. The MERGE/XL portion of the subsystem allows you to combine data from two or more sorted files into a single, new, merged file.

SORT-MERGE/XL operates as a standalone utility (either interactively or in batch mode), or within a program (programmatically). This manual provides information on how to use SORT-MERGE/XL as a standalone utility. It also serves as a reference manual for SORT-MERGE/XL commands. For information on how to use SORT-MERGE/XL programmatically, refer to the *SORT-MERGE/XL Programmer's Guide* (32650-90080).

What Can You Sort?

You can use SORT-MERGE/XL to manipulate data for various business applications. Some types of data you might choose to sort or merge could include:

- Employee or customer names.
- Dates, telephone numbers, or zip codes.
- Inventory or payroll numbers.
- Part numbers or model numbers.
- Check numbers or amounts.
- Existing data to be merged with new data.

How Can You Sort?

You can use SORT-MERGE/XL to sort or merge data in various ways. Some sequences you might choose as the basis for sorting or merging data could be:

- Alphabetically in either an ascending or descending order.
- Numerically in either an ascending or descending order.
- Alphabetically or numerically based on a single key data item.
- Alphabetically or numerically based on more than one key data item.
- Define a unique collating sequence for your application.
- Merge two or more sorted files into a new merged file.

The SORT/XL Process

The procedure for using the SORT/XL subsystem is explained below. By invoking just a few commands you can convert the types of data mentioned above, from random listings, into productive and useful data.

The following example shows accessing SORT/XL, identifying the file to be sorted, identifying the file where the sorted data is to be stored, identifying the item(s) within the file to be sorted, and initiating the sort operation.

```
:SORT
>INPUT filename
>OUTPUT filename
>KEY 1,10
>END
```

Taking this example line by line:

```
:SORT
```

Specifying `SORT` at the MPE XL colon prompt (`:`) takes you into the SORT/XL subsystem and displays the subsystem chevron prompt (`>`). The ability to run a program, such as `SORT.PUB.SYS`, without explicitly using the MPE XL `:RUN` command is called an Implied :RUN. You can use the `:RUN` command (`:RUN SORT.PUB.SYS`) or simply enter `:SORT` to access the subsystem.

```
>INPUT filename
```

Specifies invoking the SORT/XL `>INPUT` command and identifies the file you want sorted.

```
>OUTPUT filename
```

Specifies invoking the SORT/XL `>OUTPUT` command and identifies the name of the file where the sorted data is to be stored. The file identified can be either a new or an existing file.

```
>KEY 1, 10
```

Specifies invoking the `>KEY` command and identifies the location of the data you want sorted. For example, `1` identifies the location of the data (the first character position of each line in the file) and `10` identifies the length of the data (in characters).

```
>END
```

Specifies invoking the `>END` command. The `>END` command indicates to the subsystem that all commands have been entered and the sort specified should be performed. After the sort operation is completed the data is stored in the specified file, the subsystem is exited, and you are returned to the MPE XL colon prompt (:).

The sorted data is accessed through the text processing system you used to create the files containing data to be sorted. The EDIT/V text editing subsystem is supplied with the 900 Series HP 3000. Check with your System Manager to determine what editors are available on your system.

There are other SORT/XL commands you can use to manipulate data. Refer to Chapter 6 for additional information on commands.

The MERGE/XL Process

The procedure for using the MERGE/XL subsystem is explained below. By invoking just a few commands, you can combine data from two or more sorted files into a single merged file.

The following example shows accessing MERGE/XL, identifying the files to be merged, identifying the file where the merged data is to be stored, the item(s) within the files to be merged, and initiating the merge operation.

```
:MERGE
>INPUT filename,filename...filename
>OUTPUT filename
>KEY 1, 10
>END
```

Taking this example line by line:

```
:MERGE
```

Specifying `MERGE` at the MPE XL colon prompt (:) takes you into the MERGE/XL subsystem and displays the subsystem chevron prompt (`>`). The ability to run a program, such as `MERGE.PUB.SYS`, without explicitly using the MPE XL `:RUN` command is called an Implied :RUN. You can use the `:RUN` command (`:RUN MERGE.PUB.SYS`) or simply enter `:MERGE` to access the subsystem.

```
>INPUT filename,filename...filename
```

Specifies invoking the MERGE/XL `>INPUT` command and identifies the files you want merged. Files identified with the MERGE/XL `>INPUT` command must be sorted files.

```
>OUTPUT filename
```

Specifies invoking the MERGE/XL `>OUTPUT` command and identifies the name of the file where the merged data is to be stored. The file identified can be either a new or an existing file.

```
>KEY 1, 10
```

Specifies invoking the `>KEY` command and identifies the location of the data you want merged. For example, `1` identifies the location of the data (the first character position of each line in the file) and `10` identifies the length of the data (in characters). The data in all files you want merged must be aligned in identical formats.

```
>END
```

Specifies invoking the `>END` command. The `>END` command indicates to the subsystem that all commands have been entered and the merge operation specified should be performed. After the merge operation is completed the data is stored in the specified file, the subsystem is exited, and you are returned to the MPE XL colon prompt (:).

The merged data is accessed through the text processing system you used to create the files containing data to be sorted and merged. The EDIT/V text editing subsystem is supplied with the 900 Series HP 3000. Check with your System Manager to determine what editors are available on your system.

There are other commands you can use to merge data. Refer to Chapter 6 for additional information on commands.

2

Getting Started With SORT-MERGE/XL

This chapter introduces SORT-MERGE/XL commands, key data items, collating sequences, translation tables, and file types.

SORT-MERGE/XL commands are key words that direct the subsystem to perform a specific operation.

A key data item is the data contained within a specific location in a file. Key data items are sometimes referred to as data fields.

Collating sequences define the order in which characters and items are sorted or merged. You can alter the standard collating sequence to customize data for a particular application.

Translation tables can be generated by the SORT-MERGE/XL subsystem to show ASCII or EBCDIC characters and their ordinal values.

Several types of files are used by SORT-MERGE/XL during its operation. Some are defined by the user and some are system generated.

SORT- MERGE/XL Commands

The SORT-MERGE/XL commands available to execute sort or merge operations on files are introduced below. Table 2-1 lists each SORT-MERGE/XL command, gives its abbreviation, and defines its function. With the exception of `>INPUT` and `>OUTPUT`, all commands are identical in format for both SORT/XL and MERGE/XL.

Chapter 6 provides a complete description of all commands and their available options.

Table 2-1. SORT-MERGE/XL Commands

Command	Abbreviation	Function
>ALTSEQ	>A	The >ALTSEQ command defines a collating sequence by modifying the standard ASCII (or EBCDIC) collating sequence.
>DATA	—	The >DATA command specifies the type of input data (ASCII or EBCDIC) and the basic collating sequence to be used in the sort or merge operation.
>END	>E	The >END command indicates that all specifications of SORT/XL or MERGE/XL commands are concluded and the operation specified should be performed.
>EXIT	>EX	The >EXIT command terminates the SORT/XL or MERGE/XL program. Once issued it prevents any sort or merge operation from being performed.
>INPUT (MERGE/XL)	>I	In MERGE/XL, the >INPUT command specifies the sorted input files to be merged.
>INPUT (SORT/XL)	>I	In SORT/XL, the >INPUT command specifies the input file(s) to be sorted.
>KEY	>K	The >KEY command specifies the location in the record of the key data items to be used as the basis for the sort or merge operation.
>LANGUAGE	>L	The >LANGUAGE command defines the configured native language collating sequence to be used.

Table 2-1. SORT-MERGE/XL Commands (Cont.)

Command	Abbreviation	Function
>OUTPUT (MERGE/XL)	>O	In MERGE/XL, the >OUTPUT command defines and creates the output file which will contain the merged records.
>OUTPUT (SORT/XL)>O	In SORT/XL, the >OUTPUT command defines and creates the output file which will contain the sorted records.	
>RESET	—	The >RESET command corrects errors made while issuing the >KEY command.
>SHOW	>SH	The >SHOW command displays the collating sequence or the translation table.
>VERIFY	>V	The >VERIFY command displays the various options specified for a particular sort or merge operation.
>:MPE Command	—	The colon command (:) is entered prior to issuing MPE XL system commands from within either SORT/XL or MERGE/XL.
>:EOD	—	The >:EOD command terminates the list of input records to SORT/XL when the terminal ($STDIN) is used as the input file.

Key Data Items

In SORT-MERGE/XL a key data item is a group of alphabetic, numeric, or alphanumeric characters. The key data item is used by SORT-MERGE/XL as a reference to find and arrange the data in a specified order. You specify a key data item by identifying its position (column) in the record and its length (number of succeeding columns) with the >KEY command.

A record is a continuous collection of related data that is treated as one unit. A record can consist of more than one line of data in a file. It is continued to subsequent lines by entering an ampersand (&) as the last nonblank character on a line.

To define a key data item with the >KEY command, enter:

```
>KEY 40, 12
```

This specifies a key data item that begins in the 40th character position (column) of the record and is 12 characters (columns) long.

You can sort or merge data based on one or more key data items.

Sorting Files By A Single Key Data Item

SORT-MERGE/XL can sort or merge files based on a single key data item within a record. Figure 2-1 shows three records of data in a file. Each record is one line in length. Each record contains a person's last name, first name, occupation, and social security number. The last name starts in column 1, the first name in column 10, the occupation in column 25, and the social security number in column 40.

Note

The examples in this chapter contain two extra lines of numbers (for example, `123456 ... 9`). These two extra lines are included to show that the data is aligned in the columns established as key data items. These two extra lines will not appear in your file.

```
         1         2         3         4
1234567890123456789012345678901234567890123456789

WELBY    MARCUS          PHYSICIAN       242244444
JONES    SMOKEY          TRUCKER         333887777
SOUSE    EGBERT          DETECTIVE       123234454
```

Figure 2-1. Key Data Item Positions

To define the last name as the single key data item to be sorted alphabetically, enter:

```
>KEY 1, 9
```

The result of a sort done on the records shown in Figure 2-1, based on the command `>KEY 1, 9`, is shown in Figure 2-2 Note that the last names are now arranged in alphabetical order.

```
         1         2         3         4
1234567890123456789012345678901234567890123456789

JONES    SMOKEY          TRUCKER         333887777
SOUSE    EGBERT          DETECTIVE       123234454
WELBY    MARCUS          PHYSICIAN       242244444
```

Figure 2-2. Alphabetical Sort By Last Name

To define the social security number as the single key data item to be sorted numerically, enter:

```
>KEY 40, 9
```

The result of a sort done on the records shown in Figure 2-1, based on the command `>KEY 40, 9`, is shown in Figure 2-3. Note that the social security numbers are now arranged in ascending numerical order.

```
          1         2         3         4
12345678901234567890123456789012345678901234567890

SOUSE     EGBERT          DETECTIVE       123234454
WELBY     MARCUS          PHYSICIAN       242244444
JONES     SMOKEY          TRUCKER         333887777
```

Figure 2-3. Numerical Sort By Social Security Number

Note

All entries in a file to be sorted for a key data item must start in exactly the same column. In Figure 2-1, all last names start in column 1 and the number of characters must not extend into the next data item field. Therefore, if Boris Tscherbakhanovski were added to the list of last names in Figure 2-1, his last name would have to be shortened to Tscherbak so it would not extend into the next key data item field containing first names. If you want to merge this file with other files, the key data items in all files must be located in exactly the same position and have the same data format.

Sorting Files By Multiple Key Data Items

SORT-MERGE/XL allows you to specify more than one key within a record for sort or merge purposes. For example, the data in Figure 2-1, can be arranged according to four different key data items (last name, first name, occupation, or social security number). The command to specify the last name as the single key data item to be sorted is:

```
>KEY 1, 9
```

This specifies that the key data item to be sorted begins in character position (column) 1 and is 9 characters (columns) long. The result of a sort done using the command `>KEY 1, 9` is shown in Figure 2-2.

The command to specify a multiple key data item sort with the last name as the major key data item, and the first name as the second key data item, is:

```
>KEY 1, 9; 10, 14
```

This specifies that the major key data item to be sorted is located in character position (column) 1 and is 9 characters long; the second key data item begins at character position 10 and is 14 characters long.

If there were two identical last names, the sort program would look to the second key data item to break the tie. Multiple `>KEY` commands may be entered one to a line, or all on one line, each separated by a semicolon:

```
>KEY 1, 9
>KEY 10, 14
>KEY 25, 15
```

Or:

```
>KEY 1, 9; 10, 14; 25, 15
```

If you define multiple key data items with the `>KEY` command, the priority of the sort operation is:

- SORT-MERGE/XL treats the first key data item you enter (in this example the last name) as the major key and sorts that item first.
- If there are two or more items of equal value in the major key (two identical last names), the key data items are ordered according to the second data item identified with the `>KEY` command.
- In the case of ties on the second data item, the third data item entered with the `>KEY` command is used, and so on.
- If two or more records are equal in all key fields, the original order of the records in the input file(s) is used. (This is not possible in this example since each person has a unique social security number.)

For additional information on sorting by single or multiple key data items, refer to Chapter 3.

Merging Files By Key Data Items

MERGE/XL allows you to combine two or more sorted files into a single, new file based on one or more key data items. Figure 2-4 shows a file called `SORTED1`. `SORTED1` contains the three records shown as sorted in Figure 2-2.

```
          1         2         3         4
12345678901234567890123456789012345678901234567890

JONES    SMOKEY          TRUCKER         333887777
SOUSE    EGBERT          DETECTIVE       123234454
WELBY    MARCUS          PHYSICIAN       242244444
```

Figure 2-4. Sorted File SORTED1

You can merge this data with one or more additional sorted files such as `SORTED2`, shown below in Figure 2-5.

```
          1         2         3         4
12345678901234567890123456789012345678901234567890

JONES     AL              POLITICIAN      768098989
SMITH     REGGIE          OUTFIELDER      436897302
TRUMAN    HARRY           POLITICIAN      895634409
```

Figure 2-5. Sorted File SORTED2

If you merge the two files (`SORTED1` and `SORTED2`) based on the command `>KEY 1, 9; 10, 14`, the resulting new file (`MERGED1`) would contain the information shown in Figure 2-6.

```
          1         2         3         4
12345678901234567890123456789012345678901234567890

JONES     AL              POLITICIAN      768098989
JONES     SMOKEY          TRUCKER         333887777
SMITH     REGGIE          OUTFIELDER      436897302
SOUSE     EGBERT          DETECTIVE       123234454
TRUMAN    HARRY           POLITICIAN      895634409
WELBY     MARCUS          PHYSICIAN       242244444
```

Figure 2-6. Merged File MERGED1

For additional information on merging files, refer to Chapter 4.

Collating Sequences

The collating sequence defines the order in which characters are listed and records are sorted and merged. SORT-MERGE/XL allows you to specify the collating sequence as either ASCII, EBCDIC, a native language sequence, or a user-defined sequence. You can specify these sequences to be ordered in either an ascending or descending order.

The `>DATA` command allows you to specify either an ASCII or EBCDIC collating sequence. The `>LANGUAGE` command allows you to specify the collating sequence for various non-English languages if they are configured on your system. The `>ALTSEQ` command allows you to alter the ASCII character sequence to create a customized sequence to suit your application.

A common reason for altering the standard ASCII sequence is to have each upper case letter followed by its corresponding lower case letter, rather than listing all upper case letters first, followed by all lower case letters.

You may also want to use this feature to alter the sequence of special characters. For example, an accountant might wish to have `$` appear directly after `D` (so that `$ INVENTORY` would appear immediately after `Dollar INVENTORY`, rather than with the special characters.) Refer to "Defining Your Own Collating Sequence" in Chapter 3 for an example on altering the sequence so that `$` follows `D` but comes before `E`.

The standard types of collating sequences available to you, as well as information on user-defined collating sequences, are discussed below.

ASCII/EBCDIC

ASCII and EBCDIC are the standard collating sequences used by SORT-MERGE/XL and the data processing industry. The `>ALTSEQ` command allows you to modify these sequences to suit your particular application. Refer to Chapter 6 for more information on the `>ALTSEQ` command and collating sequences.

Native Language Collating Sequences

The `>LANGUAGE` command allows you to use the collating sequences for native languages other than English if they are configured on your system. The use of native language collating sequences is described in the *Native Language Programmer's Guide* (32650-90022). Refer to Chapter 6 for additional information on the `>LANGUAGE` command and Appendix C for a list of native language collating sequences.

User-Defined Sequences

The `>ALTSEQ` command allows you to alter the standard ASCII or EBCDIC collating sequence to suit your application. Refer to Chapter 6 for additional information on the `>ALTSEQ` command.

Ascending/ Descending Order

SORT-MERGE/XL allows you to arrange records in either an ascending or descending order. Unless you specify a descending order (for example 9, 7, 1 or Z, Y, X), SORT-MERGE/XL automatically orders the data in the ascending order (for example 1, 7, 9 or X, Y, Z). To specify a descending order use the `DESC` parameter of the `>KEY` command. Refer to Chapter 6 for additional information about the `>KEY` command.

Translation Tables

SORT-MERGE/XL arranges records corresponding to the sequence shown in the system translation table. The translation table follows the standard 128-character ASCII sequence, where each character is represented internally by a numeric value from 0 to 127. For example, the numeric value for F is 70 (decimal). The table lists the characters in ascending order by numeric value. If the collating sequence is altered with the `>ALTSEQ` command, those changes are reflected in the translation table. Refer to the discussion about the `>SHOW` command in Chapter 6 for additional information on translation tables.

SORT-MERGE/XL Files

SORT-MERGE/XL uses several different types of files during its operation. These file types are defined below:

Display File

The display file receives the output from the `>SHOW` command. This output can be in the form of either the translation table or the collating sequence. The formal designator of the display file is `DISPLOUT`, which defaults to `$STDLIST`. `$STDLIST` is the terminal for a session and the printer for a job.

Input File

The input file contains the data you want to sort or merge. This file can reside on any storage device such as magnetic tape or disc. For SORT-MERGE/XL, the formal designator of the input file is `INPUT`. `INPUT` is then equated to the actual file designator you specify with the `>INPUT` command. `$NULL` is not a valid input file.

Output File

The output file receives the results of the sorted or merged records. The formal designator for the output file is `OUTPUT` which is equated to the actual file designator you specify with the `>OUTPUT` command.

List File

The list file is used by SORT-MERGE/XL to display error messages and prompts during interactive sessions. It does not contain sorted or merged records. The formal designator for the list file is `LIST`, which defaults to `$STDLIST`. `$STDLIST` is the terminal for a session and the printer for a job.

Scratch File

The scratch file is used as a work area by SORT/XL. It is not used by MERGE/XL. The formal designator for the file is `SORTSCR` in Compatibility Mode. In Native Mode, there is an unnamed scratch file. Refer to the discussion of the `>INPUT` command for SORT/XL to estimate the size of the scratch file. All extents for the file are allocated at once.

Text File SORT-MERGE/XL reads commands directly from this file. The formal designator for the file is `TEXT`, which defaults to `$STDIN`.

Prompt File Used by SORT-MERGE/XL to prompt the user for input when the text file is the session terminal but the list file is not. `PROMPT` is the formal designator which defaults to `$STDLIST`. The prompt file is the session output device.

3

Using SORT/XL Interactively

This chapter introduces using SORT/XL in an interactive session. The examples use a variety of SORT/XL commands and options to provide an overview of how SORT/XL operates. Refer to Chapter 6 for information on all SORT-MERGE/XL commands, including their syntax, parameters, options, and examples of their operation.

Throughout this chapter two files (`EMPLOYEE` and `COMPANY`) are used to illustrate how SORT/XL operates. They are patterned on typical information that might be used by the Personnel Department of your company. The data is listed by the employee's last name, first name, job title, and employee number. The file `EMPLOYEE` contains unsorted data on existing employees and is designated as the `>INPUT` file in all examples. The file `COMPANY` contains the sorted data in various orders and is designated as the `>OUTPUT` file in all examples. A third file (`NEWHIRES`) contains unsorted data on new employees and is designated as the `>INPUT` file in the example showing how to sort and merge multiple files in a single operation.

Determining the File Format

When creating a file for sorting data, first determine the information to include and its format. For example, the input file `EMPLOYEE` contains four key data items in each record (employee's last name, first name, job title, and employee number). The format for the first two lines of the file `EMPLOYEE` is shown in Figure 3-1.

```
          1         2         3         4
12345678901234567890123456789012345678901234567

1    FISHER     TOM        SHIPPING CLERK     7309
2    TAYLOR     HEATHER    SECRETARY          7272
    \---------/\---------/\----------------/\--/
         \/         \/           \/         \/
         Last       First          Job      Employee
         Name       Name         Title      Number
```

Figure 3-1. File Format For Sorting

The file format shown in Figure 3-1 allots 11 characters (columns) for last names beginning in position 1; 11 characters for first names,

beginning in position 12; 19 characters for jobs titles, beginning in position 23; and 4 characters for employee numbers, beginning in position 42. Use the starting position location for each key data item as tab settings when creating the file. A single line record can contain up to 80 characters.

For clarity in reading the report when it is printed, allow enough characters for the longest piece of information in each key data item, and some blank spaces between them.

When creating multiple files containing similar information to be sorted and then merged, ensure the key data item starting positions and data length are identical in all files.

Creating an Editor File

The SORT/XL subsystem sorts information contained in records within a file. The example files used in this manual were created using EDIT/V, which is supplied as a subsystem of MPE XL on the 900 Series HP 3000. SORT-MERGE/XL can manipulate files created with other editors such as Text and Document Processing/V (TDP/V). Check with your System Manager to determine which editors are available on your system.

To access EDIT/V, at the MPE XL colon prompt (:), enter:

```
:EDITOR
```

The EDIT/V banner appears, followed by the subsystem slash prompt (/):

```
HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987,  8:10 AM
© HEWLETT-PACKARD CO. 1985
/
```

The following example shows how to create a file named **EMPLOYEE** using the EDIT/V **SET** command tab function. The tab locations you establish can then be used to designate the location of key data items with the **>KEY** command when sorting files. Tabs automatically align the data in the file for you.

After you access the EDIT/V subsystem, establish the tab character and the tabs for file to be created. In this example the exclamation point (**!**) is used as the tab character and the tabs are set at 12, 23, and 42:

```
/SET TABCHAR="!", TABS=(12, 23, 42)
```

To verify that the tab character and tabs are set correctly, enter:

```
/VERIFY TABCHAR, TABS
```

The system displays the message:

```
TAB CHARACTER = "!"
TABS = ( 12, 23, 42)
```

After establishing the tab character and tabs, create a new file using EDIT/V. To do so, enter an `A` (for `ADD`) at the slash prompt (`/`) and press the Return key. In response, a `1` followed by a blinking cursor appears on the terminal screen:

```
/A
  1        _ (blinking cursor)
```

The `1` represents the first line in your file and indicates the editor is ready for you to enter data. As each line becomes full, or when you press the Return key, a new line number appears. The blinking cursor indicates where you begin entering data. Enter the data in the following format:

```
/A
  1     FISHER!TOM!SHIPPING CLERK!7309
  2     TAYLOR!HEATHER!SECRETARY!7272
  3     ANDERSON!MARY!ACCOUNTANT!6345
  4     LANGE!ROBERT!ENGINEER!3235
  5     ANDERSON!CHARLES!SALES REP!3456
  6     ANDERSON!CHARLES!PRESIDENT!0247
  7     ZIMMER!ANDREW!ENGINEER!5739
  8     SMITH!HOWARD!DESIGNER!6794
  9     CARLSON!ROBERTA!TREASURER!3586
 10     JOHNSON!FRANCES!RECEPTIONIST!7943
 11     //
```

Tell the system you are finished adding data by entering two slashes (`//`) as the first two characters on a new line. The system responds by displaying three dots and then the subsystem slash prompt:

```
...
/
```

At the slash prompt enter `LIST ALL` to display the data aligned according to the tabs you set.

Note

The examples in this chapter contain two extra lines of data containing numbers (for example, `123456 ... 5`). These two lines are included to show that the data is aligned in the columns established as tabs which are then used for specifying key data items with the `>KEY` command. These two extra lines will not appear in your file.

```
/ LIST ALL
                  1         2         3         4
        12345678901234567890123456789012345678901234567

  1     FISHER     TOM        SHIPPING CLERK     7309
  2     TAYLOR     HEATHER    SECRETARY          7272
  3     ANDERSON   MARY       ACCOUNTANT         6345
  4     LANGE      ROBERT     ENGINEER           3235
```

```
 5      ANDERSON    CHARLES    SALES REP          3456
 6      ANDERSON    CHARLES    PRESIDENT          0247
 7      ZIMMER      ANDREW     ENGINEER           5739
 8      SMITH       HOWARD     DESIGNER           6794
 9      CARLSON     ROBERTA    TREASURER          3586
10      JOHNSON     FRANCES    RECEPTIONIST       7943
```

The data is now aligned with the last names beginning in position (column) 1 of the record, first names in position 12, job titles in position 23, and employee numbers in position 42.

Keep (save) the file and give it the unique name `EMPLOYEE` by entering `KEEP EMPLOYEE, UNNUMBERED`. To sort a file using SORT/XL it is necessary to keep the file in an `UNNUMBERED` state. `UNNUMBERED` does not refer to the line numbers that appear on the screen. These will continue to be displayed for your convenience in editing your files.

```
/KEEP EMPLOYEE, UNNUMBERED
```

To ensure the file has been successfully created, exit EDIT/V by entering `E` (for `END`) at the slash prompt (`/`). Then at the MPE XL colon prompt (`:`), enter `LISTF`:

```
/E

END OF SUBSYSTEM
:LISTF

FILENAME

EMPLOYEE
```

The file `EMPLOYEE` has been created. You can now add, modify, or delete information in the file with EDIT/V, or use the file with the SORT/XL subsystem to arrange the information in different orders. The remainder of this chapter illustrates how to manipulate this data in ways useful to a Personnel Department.

If you need additional information on creating, modifying, and keeping (saving) files refer to the *EDIT/3000 Reference Manual* (03000-90012).

Initiating a SORT/XL Interactive Session

After you create the editor file containing the data to be sorted, begin an interactive session using the SORT/XL subsystem. To do so, at the MPE XL colon prompt (:), enter:

```
:SORT
```

This accesses the SORT/XL subsystem and makes the capabilities of the program SORT.PUB.SYS available to you. The ability to run a program, such as SORT.PUB.SYS, without explicitly using the MPE XL :RUN command is called an Implied :RUN. You can use the :RUN command (:RUN SORT.PUB.SYS) or simply enter :SORT to access the subsystem.

The SORT/XL header appears,followed by the subsystem chevron prompt (>):

```
HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  8:18 AM
© HEWLETT-PACKARD CO. 1986

>
```

You are now in the SORT/XL subsystem program and can enter SORT/XL commands at the chevron prompt (>).

Exiting SORT/XL

To terminate access to the SORT/XL subsystem without performing a sort operation, use the >EXIT command. The >EXIT command prevents any sort operation from being performed and returns you to the MPE XL colon prompt (:).

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  8:19 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 1, 11
>EXIT
:
```

Single Key Alphabetical Sorting

A basic sorting operation can arrange, or order, data in an ascending alphabetical sequence, using a single key data item. The following example shows the commands you use to direct SORT/XL to order the last names in the file EMPLOYEE into a standard alphabetical order. Following the SORT/XL commands, you will see the steps for entering the EDIT/V subsystem to display the results of the sort process. As mentioned above, the output file containing the sorted data is named COMPANY.

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  8:20 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 1, 11
>END

   <<The SORT Statistics Appear Here>>

:EDITOR

HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987,  8:21 AM
© HEWLETT-PACKARD CO. 1985
/TEXT COMPANY
FILE UNNUMBERED
/ LIST ALL
                  1         2         3         4
         12345678901234567890123456789012345678901234567890

  1      ANDERSON   MARY       ACCOUNTANT         6345
  2      ANDERSON   CHARLES    SALES REP          3456
  3      ANDERSON   CHARLES    PRESIDENT          0247
  4      CARLSON    ROBERTA    TREASURER          3586
  5      FISHER     TOM        SHIPPING CLERK     7309
  6      JOHNSON    FRANCES    RECEPTIONIST       7943
  7      LANGE      ROBERT     ENGINEER           3235
  8      SMITH      HOWARD     DESIGNER           6794
  9      TAYLOR     HEATHER    SECRETARY          7272
 10      ZIMMER     ANDREW     ENGINEER           5739
/E

END OF SUBSYSTEM
:
```

The SORT/XL program has listed the employee's last names alphabetically. There are three ANDERSON entries. Notice they are not alphabetized according to their first names. If there is a tie in a single key sort, as in this case, he names are listed in the order in which they appeared in the input file (see where these three names were listed in the file EMPLOYEE shown earlier in this chapter). Refer to the section on "Multiple Key Alphabetical Sort" below for information on breaking ties while running the SORT/XL subsystem.

SORT/XL Statistics Report

SORT/XL generates a statistical report summarizing the sort operation. This statistical report is generated and displayed each time you enter the >END command. Values for a sort operation on the file EMPLOYEE might be:

```
                    STATISTICS

NUMBER OF RECORDS               =         10
NUMBER OF INTERMEDIATE PASSES   =          0
SPACE AVAILABLE (IN WORDS)      =     11,090
NUMBER OF COMPARES              =         34
NUMBER OF SCRATCHFILE IO'S      =          8
CPU TIME (MINUTES)              =        .00
SCRATCH FILE SIZE (#SECTORS)    =         11
ELAPSED TIME (MINUTES)          =        .01
:
```

The statistics generated by this report are used mostly by System Managers and Programmers. Although this information may not apply to you, it is mentioned here since it appears on your terminal screen each time you enter the >END command to start a sort operation. Refer to the *SORT-MERGE/XL Programmer's Guide* (32650-90080) for additional information on SORT/XL statistics.

Multiple Key Alphabetical Sorting

You can designate multiple key data items to break sorting ties. This prevents the situation described in the "Single Key Alphabetical Sorting" section above, where there were three ANDERSON entries in the file. Issue the following commands to designate multiple keys for the file EMPLOYEE. Only the three lines of the file listing the ANDERSON entries are shown:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  8:28 AM
© HEWLETT-PACKARD CO. 1986
```

```
>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 1, 11
>KEY 12, 11
>END
PURGE OLD OUTPUT FILE COMPANY.GROUP.ACCOUNT ? Y

   <<The SORT Statistics Appear Here>>

:EDITOR

HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987,  8:29 AM
© HEWLETT-PACKARD CO. 1985
/TEXT COMPANY
FILE UNNUMBERED
/LIST 1/3
              1         2         3         4
     12345678901234567890123456789012345678901234

1    ANDERSON   CHARLES    SALES REP          3456
2    ANDERSON   CHARLES    PRESIDENT          0247
3    ANDERSON   MARY       ACCOUNTANT         6345
```

The two entries for CHARLES ANDERSON now appear before MARY ANDERSON in the list. However, for the sort to be completely alphabetized the job title also needs to be considered. To accomplish this you would designate three key data items with the >KEY command.

To designate three key data items for last name, first name, and job title, enter the following sequence of commands:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  8:30 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 1, 11; 12, 11; 23, 19
>END
PURGE OLD OUTPUT FILE COMPANY.GROUP.ACCOUNT ? Y

    <<The SORT Statistics Appear Here>>

 :EDITOR

 HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987,  8:31 AM
 © HEWLETT-PACKARD CO. 1985
```

```
/TEXT COMPANY
FILE UNNUMBERED
/LIST 1/3
                1         2         3         4
     12345678901234567890123456789012345678901234567

1    ANDERSON   CHARLES    PRESIDENT          0247
2    ANDERSON   CHARLES    SALES REP          3456
3    ANDERSON   MARY       ACCOUNTANT         6345
```

The three ANDERSON entries are now correctly alphabetized by last name, first name, and job title. Notice in the last two examples that it is acceptable to enter multiple key data items with the >KEY command either one to a line or all on one line.

In the above example, after you entered the >END command, the system displayed the message:

```
PURGE OLD OUTPUT FILE COMPANY.GROUP.ACCOUNT ? Y
```

This message tells you that a file named COMPANY already exists in your group and account, and asks if you want the old version purged. If you reply YES, the old version of COMPANY is purged and a new version containing the information from this sort is created. If you reply NO you are prompted for a new file name. You then enter a new, unique file name and you have two files; the original file named COMPANY and the newly created file.

Using >VERIFY to Check Options

The >VERIFY command allows you to check the options you specified for the sort operation. Enter the >VERIFY command after the >KEY command, as follows:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  8:32 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 1, 11; 12, 11; 23, 19
>VERIFY
```

SORT/XL responds to the >VERIFY command with the following display:

```
INPUT ENTITY = EMPLOYEE
RECORD LENGTH  = SAME AS THAT OF THE INPUT FILE
OUTPUT ENTITY = COMPANY
KEY POSITION    LENGTH  TYPE  ASC/DESC
       1         11     BYTE     ASC  (MAJOR KEY)
      12         11     BYTE     ASC
      23         19     BYTE     ASC
```

This display tells you that the input file is EMPLOYEE; the output file is COMPANY; and the sort is based on three designated keys. The first key (identified as the major key) starts in character position 1 and is 11 characters long. In case of ties on the first key, entries in COMPANY are sorted according to the second key. The second key starts in character position 12 and is 11 characters long. The third key starts in character position 23 and is 19 characters long. It also shows that the default values for TYPE (BYTE) and ASC/DESC (ASC for ascending) are used. Refer to the >VERIFY command in Chapter 6 for additional information.

Sorting Numerically

The last column of data in the file EMPLOYEE lists employee numbers. These were assigned chronologically to each new employee. To obtain a list of all employees in the order of their hiring you would proceed as shown in the following example:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  8:35 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 42, 4
>END
PURGE OLDOUTPUT FILE COMPANY.GROUP.ACCOUNT ? Y

   <<The SORT Statistics Appear Here>>

:EDITOR

HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987,  8:40 AM
© HEWLETT-PACKARD CO. 1985
/TEXT COMPANY
 FILE UNNUMBERED
/LIST ALL
                1         2         3         4
       12345678901234567890123456789012345678901234

  1    ANDERSON   CHARLES    PRESIDENT          0247
  2    LANGE      ROBERT     ENGINEER           3235
  3    ANDERSON   CHARLES    SALES REP          3456
  4    CARLSON    ROBERTA    TREASURER          3586
  5    ZIMMER     ANDREW     ENGINEER           5739
  6    ANDERSON   MARY       ACCOUNTANT         6345
  7    SMITH      HOWARD     DESIGNER           6794
  8    TAYLOR     HEATHER    SECRETARY          7272
  9    FISHER     TOM        SHIPPING CLERK     7309
 10    JOHNSON    FRANCES    RECEPTIONIST       7943
```

To determine the newest employee, and obtain a list in descending order to the one with the most seniority, use the DESC (for descending) parameter of the >KEY command:

```
>KEY 42, 4, DESC
```

The file COMPANY would now list Receptionist, FRANCES JOHNSON as the first record in the file and President, CHARLES ANDERSON, as the last record in the file.

Sorting and Merging Multiple Files

SORT-MERGE/XL allows you to sort and merge multiple files in a single operation. This is done by placing the names of the files to be sorted and then merged within parentheses when entering the >INPUT command. Below, the files EMPLOYEE (shown in the previous example) and NEWHIRES (shown below) are sorted by job title and then merged into the file COMPANY in a single operation. The file NEWHIRES contains the following four records:

```
                1         2         3         4
       12345678901234567890123456789012345678901234567

   1   CARLSON    PETER      BUYER               8043
   2   ADAMS      JERROLD    INSPECTOR           8044
   3   MATHEWS    EDDY       PLANNER             8045
   4   CLARK      STEVE      ASSEMBLER           8046
```

To sort, and then merge, these two files in a single operation, enter the following commands:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  9:11 AM
© HEWLETT-PACKARD CO. 1986

>INPUT (EMPLOYEE, NEWHIRES)
>OUTPUT COMPANY
>KEY 23, 19
>END
>PURGE OLDOUTPUT FILE COMPANY.GROUP.ACCOUNT? Y

   <<The SORT Statistics Appear Here>>

 :EDITOR

 HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987,  9:12 AM
 © HEWLETT-PACKARD CO. 1985
 /TEXT COMPANY
 FILE UNNUMBERED
 /LIST ALL

                1         2         3         4
       12345678901234567890123456789012345678901234567

   1   ANDERSON   MARY       ACCOUNTANT          6345
   2   CLARK      STEVE      ASSEMBLER           8046
   3   CARLSON    PETER      BUYER               8043
   4   SMITH      HOWARD     DESIGNER            6794
   5   LANGE      ROBERT     ENGINEER            3235
   6   ZIMMER     ANDREW     ENGINEER            5739
```

```
 7   ADAMS      JERROLD    INSPECTOR          8044
 8   MATHEWS    EDDY       PLANNER            8045
 9   ANDERSON   CHARLES    PRESIDENT          0247
10   JOHNSON    FRANCES    RECEPTIONIST       7943
11   ANDERSON   CHARLES    SALES REP          3456
12   TAYLOR     HEATHER    SECRETARY          7272
13   FISHER     TOM        SHIPPING CLERK     7309
14   CARLSON    ROBERTA    TREASURER          3586
```

The two files (EMPLOYEE and NEWHIRES) are sorted and merged in the same SORT/XL operation and the output file COMPANY is created.

Saving Selected Key Data Only

It is possible to create an output file containing only those key data items you need rather than the entire file. To do this you use the KEY parameter of the >OUTPUT command.

For example, if you want a listing of all employees, showing their last names only, enter the following sequence of commands:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  9:13 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY, KEY
>KEY 1, 11
>END

   <<The SORT Statistics Appear Here>>

 END OF PROGRAM
 :EDITOR

 HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987,  9:14 AM
 © HEWLETT-PACKARD CO. 1985
 /TEXT COMPANY
 FILE UNNUMBERED
 /LIST ALL

        12345678901

   1     ANDERSON
   2     ANDERSON
   3     ANDERSON
   4     CARLSON
```

```
  5     FISHER
  6     JOHNSON
  7     LANGE
  8     SMITH
  9     TAYLOR
 10     ZIMMER
```

To receive a hard copy (printed report) of the results of the sort operation shown in the examples above, request a copy by entering `LIST ALL, OFFLINE` from within the EDIT/V subsystem. To receive a printed copy follow the procedure below:

```
:EDITOR

HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987,  9:15 AM
© HEWLETT-PACKARD CO. 1985
/TEXT COMPANY
FILE UNNUMBERED
/LIST ALL, OFFLINE
```

A message appears on your terminal screen indicating the printing has begun:

```
***OFF LINE LISTING BEGUN***
```

Wait a few minutes to allow the job to be processed; then get your printout from the system printer.

Using the MPE XL :PRINT Command

In the examples in this chapter, you were directed back to the editor to text the output file to view the results of the sort operation. The MPE XL :PRINT command allows you to view the results of the sort operation without calling EDIT/V. The command also allows you to print the results of the sort on the system printer.

For example, to view the results of a single key sort, as shown earlier in this chapter, you would proceed, as follows:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  9:16 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 1, 11
>END

   <<The SORT Statistics Appear Here>>

:PRINT COMPANY
ANDERSON    MARY       ACCOUNTANT         6345
ANDERSON    CHARLES    SALES REP          3456
ANDERSON    CHARLES    PRESIDENT          0247
CARLSON     ROBERTA    TREASURER          3586
FISHER      TOM        SHIPPING CLERK     7309
JOHNSON     FRANCES    RECEPTIONIST       7943
LANGE       ROBERT     ENGINEER           3235
SMITH       HOWARD     DESIGNER           6794
TAYLOR      HEATHER    SECRETARY          7272
ZIMMER      ANDREW     ENGINEER           5739
```

To have a copy of this report printed on the line printer use the MPE XL :FILE command to establish the following equation:

```
:FILE T;DEV=LP
:PRINT COMPANY, *T
```

This equation establishes T as the file equated with the line printer. That is then backreferenced with the :PRINT command to send the file COMPANY to the line printer.

Using >SHOW to Display the Collating Sequence

Use the >SHOW command to display the designated collating sequence on your terminal screen or have it printed on the system printer.

To display the standard ASCII sequence on your terminal screen, enter the following commands in an interactive session:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  9:17 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>SHOW SEQUENCE
```

```
nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
 sp   !   "   #   $   &   %   ’   (   )   *   +   ,   -   .   /
  0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
  @   A   B   C   D   E   F   G   H   I   J   K   L   M   N   O
  P   Q   R   S   T   U   V   W   X   Y   Z   [   \   ]   ^   _
  ‘   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
  p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

To receive a copy of this collating sequence from the printer, use the OFFLINE parameter of the >SHOW command, as follows:

```
>DATA IS ASCII, SEQUENCE IS ASCII
>SHOW SEQUENCE, OFFLINE
```

For additional information on collating sequences, refer to the >SHOW command in Chapter 6.

Using >SHOW to Display the Translation Table

Use the >SHOW command to display the translation table on your terminal screen or have it printed on the system printer.

To display the standard ASCII translation table on the terminal screen, use the TABLE parameter of the >SHOW command. The translation table follows the standard 128-character ASCII sequence. It shows the ordinal value for each character. For example, the numeric value for F is 70 (decimal). To generate the ASCII translation table, enter the following commands:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  9:20 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>SHOW TABLE

TABLE OF ORDINAL VALUE ASSIGNED TO EACH CHARACTER.
    !    0 !    1 !    2 !    3 !    4 !    5 !    6 !    7 !    8 !    9
----+------+------+------+------+------+------+------+------+------+------
  0 !    0 !    1 !    2 !    3 !    4 !    5 !    6 !    7 !    8 !    9 !
  1 !   10 !   11 !   12 !   13 !   14 !   15 !   16 !   17 !   18 !   19 !
  2 !   20 !   21 !   22 !   23 !   24 !   25 !   26 !   27 !   28 !   29 !
  3 !   30 !   31 !sp=32 !!= 33 !"= 34 !#= 35 !$= 36 !%= 36 !&= 38 !'= 39 !
  4 !(= 40 !)= 41 !*= 42 !+= 43 !,= 44 !-= 45 !.= 46 !/= 47 !0= 48 !1= 49 !
  5 !2= 50 !3= 51 !4= 52 !5= 53 !6= 54 !7= 55 !8= 56 !9= 57 !:= 58 !;= 59 !
  6 !<= 60 !== 61 !>= 62 !?= 63 !@= 64 !A= 65 !B= 66 !C= 67 !D= 68 !E= 69 !
  7 !F= 70 !G= 71 !H= 72 !I= 73 !J= 74 !K= 75 !L= 76 !M= 77 !N= 78 !O= 79 !
  8 !P= 80 !Q= 81 !R= 82 !S= 83 !T= 84 !U= 85 !V= 86 !W= 87 !X= 88 !Y= 89 !
  9 !Z= 90 ![= 91 !\= 92 !]= 93 !^= 94 !_= 95 !`= 96 !a= 97 !b= 98 !c= 99 !
 10 !d=100 !e=101 !f=102 !g=103 !h=104 !i=105 !j=106 !k=107 !l=108 !m=109 !
 11 !n=110 !o=111 !p=112 !q=113 !r=114 !s=115 !t=116 !u=117 !v=118 !w=119 !
 12 !x=120 !y=121 !z=122 !{=123 !|=124 !}=125 !~=126 ! =127 !  128 !  129 !
 13 !  130 !  131 !  132 !  133 !  134 !  135 !  136 !  137 !  138 !  139 !
 14 !  140 !  141 !  142 !  143 !  144 !  145 !  146 !  147 !  148 !  149 !
 15 !  150 !  151 !  152 !  153 !  154 !  155 !  156 !  157 !  158 !  159 !
 16 !  160 !  161 !  162 !  163 !  164 !  165 !  166 !  167 !  168 !  169 !
 17 !  170 !  171 !  172 !  173 !  174 !  175 !  176 !  177 !  178 !  179 !
 18 !  180 !  181 !  182 !  183 !  184 !  185 !  186 !  187 !  188 !  189 !
 19 !  190 !  191 !  192 !  193 !  194 !  195 !  196 !  197 !  198 !  199 !
 20 !  200 !  201 !  202 !  203 !  204 !  205 !  206 !  207 !  208 !  209 !
 21 !  210 !  211 !  212 !  213 !  214 !  215 !  216 !  217 !  218 !  219 !
 22 !  220 !  221 !  222 !  223 !  224 !  225 !  226 !  227 !  228 !  229 !
 23 !  230 !  231 !  232 !  233 !  234 !  235 !  236 !  237 !  238 !  239 !
 24 !  240 !  241 !  242 !  243 !  244 !  245 !  246 !  247 !  248 !  249 !
 25 !  250 !  251 !  252 !  253 !  254 !  255 !
```

WHEN PASSED TO SORTINIT, THE TABLE ABOVE IS PRECEDED BY TWO BYTES. THESE FIRST TWO BYTES CONTAIN A FLAG BYTE OF %000 AND A LENGTH BYTE OF %377 RESPECTIVELY.

This translation table display is followed by a display of the contents of the `ALTSEQ` array in decimal format and octal word format. This information is not shown here as it is intended for programmatic use. Refer to the *SORT-MERGE/XL Programmer's Guide* (32650-90080) for additional information.

To receive a copy of the standard ASCII translation table from the printer, use the `OFFLINE` parameter of the `>SHOW` command, as follows:

```
>DATA IS ASCII, SEQUENCE IS ASCII
>SHOW TABLE, OFFLINE
```

For additional information on translation tables refer to the `>SHOW` command in Chapter 6.

Defining Your Own Collating Sequence

You can define a customized collating sequence unique to your application. For example, in the section "Collating Sequences" in Chapter 2, an example was used where an accountant wanted the special character `$` to follow the `D` in the collating sequence. This makes it possible to have `$ INVENTORY` follow `DOLLAR INVENTORY` but come before other entries, such as `EXPENSES`.

To alter the standard sequence so `$ INVENTORY` follows `DOLLAR INVENTORY`, but comes before `EXPENSES`, enter the following commands:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  9:30 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>ALTSEQ MERGE "D" = "$"
>SHOW SEQUENCE

     nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
     del dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
      sp   !   "   #   %   &   '   (   )   *   +   ,   -   .   /   0
       1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?   @
       A   B   C   D   $   E   F   G   H   I   J   K   L   M   N   O
       P   Q   R   S   T   U   V   W   X   Y   Z   [   \   ]   ^   _
       `   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
       p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

The following example shows the results of the sort with and without altering the collating sequence for the entries $ INVENTORY, DOLLAR INVENTORY, and EXPENSES:

SORT WITH ALTERED SEQUENCE	SORT WITH STANDARD SEQUENCE
DOLLAR INVENTORY	DOLLAR INVENTORY
$ INVENTORY	EXPENSES
EXPENSES	$ INVENTORY

A commonly used alteration to the standard ASCII collating sequence is merging upper case and lower case alphabetic characters. In the standard collating sequence, all upper case characters precede all lower case characters. The standard ASCII collating sequence is shown in the section, "Using >SHOW to Display the Collating Sequence". To order alternating upper case and lower case characters, enter the following commands:

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987,  9:45 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>ALTSEQ MERGE "A-Z" with "a-z"
>SHOW SEQUENCE

     nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
     del dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
      sp   !   "   #   $   %   &   '   (   )   *   +   ,   -   .   /
       0   1   2   3   4   5   6   7   8   9   :   ;   <   +   >   ?
       @   A   a   B   b   C   c   D   d   E   e   F   f   G   g   H
       h   I   i   J   j   K   k   L   l   M   m   N   n   O   o   P
       p   Q   q   R   r   S   s   T   t   U   u   V   v   W   w   X
       x   Y   y   Z   z   [   \   ]   ^   _   `   {   |   }   ~ del
```

For additional information on altering collating sequences, refer to the >ALTSEQ command in Chapter 6.

Using the Terminal as the Output File

It is possible to enter the SORT/XL subsystem, issue a series of commands, and have the results of the sort operation immediately displayed on your terminal screen. This process eliminates the need of going into EDIT/V, calling up the file, and listing its contents to see the results of a sort operation. To use the terminal as the output device use the `$STDLIST` parameter of the SORT/XL `>OUTPUT` command.

The following example shows how to sort the file `EMPLOYEE` based on a single key. The results are displayed on the terminal screen when you enter the `>END` command.

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987, 10:00 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT $STDLIST
>KEY 1, 11
>END

         1         2         3         4
12345678901234567890123456789012345678901234

ANDERSON   MARY       ACCOUNTANT         6345
ANDERSON   CHARLES    SALES REP          3456
ANDERSON   CHARLES    PRESIDENT          0247
CARLSON    ROBERTA    TREASURER          3586
FISHER     TOM        SHIPPING CLERK     7309
JOHNSON    FRANCES    RECEPTIONIST       7943
LANGE      ROBERT     ENGINEER           3235
SMITH      HOWARD     DESIGNER           6794
TAYLOR     HEATHER    SECRETARY          7272
ZIMMER     ANDREW     ENGINEER           5739

   <<The SORT Statistics Appear Here>>

:
```

When you designate the terminal as the output file (`$STDLIST`), the system does not generate a system file or keep any permanent record of the sort results.

Using the Terminal as the Input File and the Output File

It is possible to enter data to be sorted with the SORT/XL subsystem without creating an input file and have the sort results displayed on the terminal screen.

To input data and have it immediately displayed on the terminal screen, use the * (for $STDIN) parameter of the SORT/XL >INPUT command and the $STDLIST parameter of the SORT/XL >OUTPUT command.

In the following example, when you enter the >END command, the system prompts you with a question mark (?). List the data you want sorted and enter the :EOD command to terminate the input records. The input data is sorted and the results displayed on your terminal screen.

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987, 10:20 AM
© HEWLETT-PACKARD CO. 1986

>INPUT *
>OUTPUT $STDLIST
>KEY 1, 4
>END
?GLOBE
?APPLE
?BANANA
?1234
?3456
?2345
?:eod
1234
2345
3456
APPLE
BANANA
GLOBE

   <<The SORT Statistics Appear Here>>

:
```

When you designate the terminal as the input file (* for $STDIN) and the output file ($STDLIST), the system does not generate a system file or keep any permanent record of the sort results.

Using File Equations in SORT/XL

An alternative method for designating the input and output files is to establish file equations at the MPE XL colon prompt (:) before you access the SORT/XL subsystem. Then when you access SORT/XL, you need only enter the >KEY and >END commands, as follows:

```
:FILE INPUT=EMPLOYEE
:FILE OUTPUT=COMPANY
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987, 11:15 AM
© HEWLETT-PACKARD CO. 1986

>KEY 1, 11; 12, 11
>END

   <<The SORT Statistics Appear Here>>

:EDITOR

HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987, 11:18 AM
© HEWLETT-PACKARD CO. 1985
/TEXT COMPANY
FILE UNNUMBERED
/LIST ALL

                1         2         3         4
       12345678901234567890123456789012345678901234567

    1  ANDERSON  CHARLES   PRESIDENT          0247
    2  ANDERSON  CHARLES   SALES REP          3456
    3  ANDERSON  MARY      ACCOUNTANT         6345
    4  CARLSON   ROBERTA   TREASURER          3586
    5  FISHER    TOM       SHIPPING CLERK     7309
    6  JOHNSON   FRANCES   RECEPTIONIST       7943
    7  LANGE     ROBERT    ENGINEER           3235
    8  SMITH     HOWARD    DESIGNER           6794
    9  TAYLOR    HEATHER   SECRETARY          7272
   10  ZIMMER    ANDREW    ENGINEER           5739
```

4

Using MERGE/XL Interactively

This chapter introduces using MERGE/XL in an interactive session. The examples show a variety of MERGE/XL commands and options to provide an overview of how MERGE/XL works. Refer to Chapter 6 for information on all SORT-MERGE/XL commands, including their syntax, parameters, options, and examples of their operation.

Throughout this chapter three files (`EMPLOYEE`, `NEWHIRES`, and `COMPANY`) are used to illustrate how MERGE/XL operates. They are patterned on typical information that might be used by the Personnel Department of your company. The data is listed by the employee's last name, first name, job title, and employee number. The file `EMPLOYEE` contains previously sorted data for existing employees, and is designated as an `>INPUT` file in all examples. The file `NEWHIRES` is an unsorted file containing a list of newly hired employees. After it is sorted, it is also designated as an `>INPUT` file in all examples. The file `COMPANY` contains the merged data from `EMPLOYEE` and `NEWHIRES` in various orders and is designated as the `>OUTPUT` file in all examples.

Determining File Format

When creating a new file to be merged with an existing file, both files must have the identical format. For example, when creating the file `NEWHIRES` to be merged with the existing file `EMPLOYEE`, the beginning position of each key data item must be identical in both files. The file `EMPLOYEE` contains four key data items in each record (employee's last name, first name, job title, and employee number). The format for the first two lines of the file `EMPLOYEE` is shown in Figure 4-1.

```
          1         2         3         4
    12345678901234567890123456789012345678901234567
1   FISHER     TOM        SHIPPING CLERK     7309
2   TAYLOR     HEATHER    SECRETARY          7272
    \---------/\---------/\-----------------/\--/
        \/         \/          \/          \/
        Last       First         Job       Employee
        Name       Name        Title       Number
```

Figure 4-1. File Format For Merging

Since you want the file format for NEWHIRES to be identical to the file format of EMPLOYEE, you would allow 11 characters (columns) for last names, beginning in position 1; 11 characters for first names, beginning in position 12; 19 characters for job titles, beginning in position 23; and 4 characters for employee numbers beginning in position 42. Use the starting position location for each key data item as tab settings when creating the file. A single line record can contain up to 80 characters.

If any of the key data items in the file NEWHIRES is longer than the number of characters established for the key data items in the file EMPLOYEE you may need to alter the format in both files. All characters exceeding established character limits are truncated and do not appear in the file COMPANY.

Creating an Editor File

The MERGE/XL subsystem merges information contained in records from two or more files. The example files used in this manual were created using EDIT/V, which is supplied as a subsystem of MPE XL on the 900 Series HP 3000. SORT-MERGE/XL can also manipulate files created with other editors such as Text and Document Processing/V (TDP/V). Check with your System Manager to determine which editors are available on your system.

To access EDIT/V, at the MPE XL colon prompt (:), enter:

```
:EDITOR
```

The EDIT/V banner appears, followed by the subsystem slash prompt (/):

```
HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987, 11:20 AM
© HEWLETT-PACKARD CO. 1985
/
```

The following example shows how to create a file named NEWHIRES using the EDIT/V SET command tab function. The tab locations you establish can then be used to designate the location of key data items with the >KEY command when merging files. Tabs automatically align the data in the file for you.

After you access the EDIT/V subsystem, establish the tab character and the tabs for the file to be created. To merge the new file NEWHIRES with the existing file EMPLOYEE, the key data items must be located in the same positions (columns) in both files. Therefore, set the tabs for the file NEWHIRES the same as those established for the file EMPLOYEE (12, 23, 42) in Chapter 3. The exclamation point (!) is used as the tab character in this example.

```
/SET TABCHAR="!", TABS=(12, 23, 42)
```

To verify that the tab character and tabs are set correctly, enter:

```
/VERIFY TABCHAR, TABS
```

The system displays the message:

```
TAB CHARACTER = "!"
TABS = ( 12, 23, 42)
```

After establishing the tab character and tabs, create a new file using EDIT/V. To do so, enter an A (for ADD) at the slash prompt (/) and press the Return key. In response, a 1 followed by a blinking cursor appears on the terminal screen:

```
/A
  1        _ (blinking cursor)
```

The 1 represents the first line in your file and indicates the editor is ready for you to enter data. As each line becomes full, or when you press the Return key, a new line number appears. The blinking cursor indicates where you enter data.

For the purposes of this example, assume four new employees were hired. To create a file containing data on these new employees use the following format:

```
/A
  1     CARLSON!PETER!BUYER!8043
  2     ADAMS!JERROLD!INSPECTOR!8044
  3     MATHEWS!EDDY!PLANNER!8045
  4     CLARK!STEVE!ASSEMBLER!8043
  5     //
```

Tell the system you are finished adding data by entering two slashes (“~~//”) as the first two characters on a new line. The system responds by displaying three dots and then the subsystem slash prompt:

```
...
/
```

At the slash prompt enter `/LIST ALL` to display the data aligned according to the tabs you set.

Note

The examples in this chapter contain two extra lines of data containing numbers (for example, `123456 ... 5`). These two lines are included to show that the data is aligned in the columns established as tabs which are then used to specify key data items with the `>KEY` command. These two extra lines will not appear in your file.

```
/LIST ALL
                1         2         3         4
       12345678901234567890123456789012345678901234

1      CARLSON    PETER      BUYER               8043
2      ADAMS      JERROLD    INSPECTOR           8044
3      MATHEWS    EDDY       PLANNER             8045
4      CLARK      STEVE      ASSEMBLER           8046
```

The data is now aligned with the last names appearing in position (column) 1 of the record, first names in position 12, job titles in position 23, and employee numbers in position 42. This alignment corresponds to the location of key data items in the file `EMPLOYEE`.

Keep (save) the file and give it the unique name `NEWHIRES` by entering `KEEP NEWHIRES, UNNUMBERED`. To merge files using MERGE/XL it is necessary to keep the files in an `UNNUMBERED` state. `UNNUMBERED` does not refer to the line numbers that appear on the screen. These continue to be displayed for your convenience in editing your files. To keep the file, enter:

```
/KEEP NEWHIRES, UNNUMBERED
```

To ensure the file has been successfully created, exit EDIT/V by entering E (for END) at the slash prompt (/). Then at the MPE XL colon prompt (:), enter LISTF:

```
/E

END OF SUBSYSTEM
:LISTF

FILENAME

EMPLOYEE          NEWHIRES
```

The file NEWHIRES has been created and is listed along with the existing file EMPLOYEE. You can now add, modify, or delete information in the file with EDIT/V, or sort the data so it can be merged with other files containing similar information.

If you need additional information on creating, modifying, and keeping (saving) files, refer to the *EDIT/3000 Reference Manual* (03000-90012).

Sorting the File

Before the newly created file NEWHIRES can be merged with any other files it must first be sorted. Sort the file NEWHIRES using the same >KEY command used in the file EMPLOYEE (>KEY 1, 11; 12, 11; 23, 19):

```
:SORT

HP32214A.01.00  SORT/3000 WED, JUN  3, 1987, 11:35 AM
© HEWLETT-PACKARD CO. 1986

>INPUT NEWHIRES
>OUTPUT NEWHIRES
>KEY 1, 11; 12, 11; 23, 19
>END
PURGE OLD OUTPUT FILE NEWHIRES.GROUP.ACCOUNT ? Y

   <<The SORT Statistics Appear Here>>

:EDITOR

HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987, 11:40 AM
© HEWLETT-PACKARD CO. 1985
/TEXT NEWHIRES
FILE UNNUMBERED
```

```
/LIST ALL
                    1         2         3         4
          12345678901234567890123456789012345678901234

1         ADAMS      JERROLD    INSPECTOR          8044
2         CARLSON    PETER      BUYER              8043
3         CLARK      STEVE      ASSEMBLER          8046
4         MATHEWS    EDDY       PLANNER            8045
```

This chapter contains examples of merging this file with the file `EMPLOYEE` in ways useful to a Personnel Department.

In the above example, after you entered the `>END` command, the system displayed the message:

```
PURGE OLD OUTPUT FILE NEWHIRES.GROUP.ACCOUNT ? Y
```

This message tells you that a file named `NEWHIRES` already exists in your group and account, and asks if you want the old version purged. If you reply `YES`, the old version of `NEWHIRES` is purged and a new version containing the information from this sort is created. If you reply `NO`, you are prompted for a new file name. You then enter a new, unique file name; and you have two files, the original file named `NEWHIRES` and the newly created file.

Initiating an Interactive MERGE/XL Session

After you create and sort the editor file containing the data to be merged with other files, begin an interactive session using the MERGE/XL subsystem. At the MPE XL colon prompt (:), enter:

```
:MERGE
```

This accesses the MERGE/XL subsystem and makes the capabilities of the program `MERGE.PUB.SYS` available to you. The ability to run a program, such as `MERGE.PUB.SYS`, without explicitly using the MPE XL `:RUN` command is called an Implied :RUN. You can use the `:RUN` command (`:RUN MERGE.PUB.SYS`) or simply enter `:MERGE` to access the subsystem.

The MERGE/XL header appears, followed by the subsystem chevron prompt (`>`):

```
HP32214A.01.00  MERGE/3000 WED, JUN  3, 1987, 11:45 AM
© HEWLETT-PACKARD CO. 1986

>
```

You are now in the MERGE/XL subsystem and can enter MERGE/XL commands at the chevron prompt (`>`).

Exiting MERGE/XL

To terminate access to the MERGE/XL subsystem without performing a merge operation, use the `>EXIT` command. The `>EXIT` command prevents any merge operation from being performed and returns you to the MPE XL colon prompt (:).

```
:MERGE

HP32214A.01.00  MERGE/3000 WED, JUN  3, 1987, 11:50 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE, NEWHIRES
>OUTPUT COMPANY
>KEY 1, 11
>EXIT
:
```

Merging Files Using a Single Key

To combine the two files EMPLOYEE and NEWHIRES using a single key data item to create a new merged file named COMPANY, enter the following commands:

```
:MERGE

HP32214A.01.00  MERGE/3000 WED, JUN  3, 1987, 11:55 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE, NEWHIRES
>OUTPUT COMPANY
>KEY 1, 11
>END
```

The two input files (EMPLOYEE and NEWHIRES) and the resulting output file (COMPANY) are shown below. These files are merged according to last name, as indicated by the command >KEY 1, 11. Since no other specification was made, the merge is done alphabetically using the default ascending alphabetical order. Notice that the entries for ADAMS, CARLSON, CLARK, and MATHEWS are merged into a single list with the other employees in the file COMPANY.

The existing file EMPLOYEE contains the following data:

```
                1         2         3         4
       12345678901234567890123456789012345678901234567

  1    ANDERSON  CHARLES   PRESIDENT          0247
  2    ANDERSON  CHARLES   SALES REP          3456
  3    ANDERSON  MARY      ACCOUNTANT         6345
  4    CARLSON   ROBERTA   TREASURER          3586
  5    FISHER    TOM       SHIPPING CLERK     7309
  6    JOHNSON   FRANCES   RECEPTIONIST       7943
  7    LANGE     ROBERT    ENGINEER           3235
  8    SMITH     HOWARD    DESIGNER           6794
  9    TAYLOR    HEATHER   SECRETARY          7272
 10    ZIMMER    ANDREW    ENGINEER           5739
```

The newly created and sorted file NEWHIRES contains the following data:

```
                1         2         3         4
       12345678901234567890123456789012345678901234567

1      ADAMS      JERROLD    INSPECTOR          8044
2      CARLSON    PETER      BUYER              8043
3      CLARK      STEVE      ASSEMBLER          8046
4      MATHEWS    EDDY       PLANNER            8045
```

The file COMPANY, created as a result of merging the files NEWHIRES and EMPLOYEE, contains the following data:

```
                1         2         3         4
       12345678901234567890123456789012345678901234567

1      ADAMS      JERROLD    INSPECTOR          8044
2      ANDERSON   CHARLES    PRESIDENT          0247
3      ANDERSON   CHARLES    SALES REP          3456
4      ANDERSON   MARY       ACCOUNTANT         6345
5      CARLSON    ROBERTA    TREASURER          3586
6      CARLSON    PETER      BUYER              8043
7      CLARK      STEVE      ASSEMBLER          8046
8      FISHER     TOM        SHIPPING CLERK     7309
9      JOHNSON    FRANCES    RECEPTIONIST       7943
10     LANGE      ROBERT     ENGINEER           3235
11     MATHEWS    EDDY       PLANNER            8045
12     SMITH      HOWARD     DESIGNER           6794
13     TAYLOR     HEATHER    SECRETARY          7272
14     ZIMMER     ANDREW     ENGINEER           5739
```

Note that the two CARLSON entries are not listed alphabetically according to their first names. In case of a tie during a single key merge, the names are listed in the order in which the system receives them. Since ROBERTA CARLSON appeared in the file EMPLOYEE, which was the first file designated with the >INPUT command, that entry is listed first in the merged file. Doing a multiple key merge, as shown below, would arrange these entries in the proper order.

Merging Files Using Multiple Keys

You can combine files based on more than one key. The files must contain the same data format, and the key data items must start at the same character position (column) and be the same length.

To combine EMPLOYEE and NEWHIRES into a new file named COMPANY by last name, first name, and job title, enter the following sequence of commands:

```
:MERGE

HP32214A.01.00  MERGE/3000 WED, JUN  3, 1987, 12:15 PM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE, NEWHIRES
>OUTPUT COMPANY
>KEY 1, 11; 12, 11; 23, 19
>END
```

This would arrange the two CARLSON entries in their proper alphabetical order with regard to their first names. Only the two CARLSON entries are shown in the following example:

```
:EDITOR

HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987, 12:30 PM
© HEWLETT-PACKARD CO. 1985
/TEXT NEWHIRES
FILE UNNUMBERED
/LIST 5/6
                1         2         3         4
       12345678901234567890123456789012345678901234567890

5      CARLSON    PETER      BUYER              8043
6      CARLSON    ROBERTA    TREASURER          3586
```

MERGE/XL Statistics Report

MERGE/XL generates a statistical report summarizing the merge operation. This statistical report is generated and displayed each time you enter the >END command. Statistical values for a merge operation on the files EMPLOYEE and NEWHIRES might be:

```
                    STATISTICS

NUMBER OF INPUT FILES =              2
NUMBER OF RECORDS =                 18
SPACE AVAILABLE (IN WORDS) =    26,872
NUMBER OF COMPARES =                15
CPU TIME (MINUTES)                 .00
RECORD SIZE (IN BYTES) =            72
:
```

The statistics generated by this report are used mostly by System Managers and Programmers. Although this information may not apply to you, it is mentioned here since it appears on your terminal screen each time you enter an >END command to start a merge operation. Refer to the *SORT-MERGE/XL Programmer's Guide* (32650-90080) for additional information on MERGE/XL statistics.

Using Verify to Check MERGE/XL Options

Use the >VERIFY command to check the options specified for the merge operation. Enter the >VERIFY command after the >KEY command, as follows:

```
:MERGE

HP32214A.01.00  MERGE/3000 WED, JUN  3, 1987, 12:15 PM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE, NEWHIRES
>OUTPUT COMPANY
>KEY 1, 11; 12, 11; 23, 19
>VERIFY
```

MERGE/XL responds to the >VERIFY command with the following display:

```
INPUT FILES = EMPLOYEE, NEWHIRES
OUTPUT ENTITY = COMPANY
KEY POSITION     LENGTH    TYPE    ASC/DESC
       1           11      BYTE      ASC   (MAJOR KEY)
      12           11      BYTE      ASC
      23           19      BYTE      ASC
```

This display tells you that the input files are EMPLOYEE and NEWHIRES, the output file is COMPANY, and the merge is based on three designated keys. The first key (identified as the major key) starts in position (column) 1 and is 11 characters (columns) long. In the case of ties on the first key, entries in the input files are merged by the second key. The second key starts in character position 12 and is 11 characters long. The third key starts in character position 23 and is 19 characters long. The display also shows that the default values for TYPE (BYTE) and the order under ASC/DESC (ASC for ascending) are used. Refer to the >VERIFY command in Chapter 6 for additional information.

Getting a Printout of MERGE/XL Results

To receive a hard copy (printed report) of the results of the merge operation shown in the examples above, request a copy by entering LIST ALL, OFFLINE from within the EDIT/V subsystem. To receive a printed copy, enter the following commands:

```
:EDITOR

HP32201A.07.17 EDIT/3000 WED, JUN  3, 1987, 12:45 PM
© HEWLETT-PACKARD CO. 1985
/TEXT COMPANY
FILE UNNUMBERED
/LIST ALL, OFFLINE
```

A message appears on your terminal screen indicating the printing has begun:

```
***OFF LINE LISTING BEGUN***
```

Wait a few minutes to allow the job to be processed; then get your printout from the system printer.

Using the MPE XL :PRINT Command

In the examples in this chapter, you were directed back to the editor to text the output file to view the results of the merge operation. The MPE XL :PRINT command allows you to view the results of the merge operation without calling the EDIT/V subsystem. This command also allows you to print the results of the merge on the system printer.

For example, to view the results of a single key merge, as shown earlier in this chapter, you would proceed, as follows:

```
:MERGE

HP32214A.01.00  MERGE/3000 WED, JUN  3, 1987, 12:31 PM
© HEWLETT-PACKARD CO. 1985

>INPUT EMPLOYEE, NEWHIRES
>OUTPUT COMPANY
>KEY 1, 11
>END

   <<The MERGE Statistics Appear Here>>
:PRINT COMPANY

        ANDERSON   CHARLES    PRESIDENT          0247
        ANDERSON   CHARLES    SALES REP          3456
        ANDERSON   MARY       ACCOUNTANT         6345
        CARLSON    ROBERTA    TREASURER          3586
        FISHER     TOM        SHIPPING CLERK     7309
        JOHNSON    FRANCES    RECEPTIONIST       7943
        LANGE      ROBERT     ENGINEER           3235
        SMITH      HOWARD     DESIGNER           6794
        TAYLOR     HEATHER    SECRETARY          7272
        ZIMMER     ANDREW     ENGINEER           5739
```

To have a copy of this report printed on the line printer use the MPE XL :FILE command to establish the following equation:

```
:FILE T;DEV=LP
:PRINT COMPANY, *T
```

This equation establishes T as the file equated with the line printer. This is then backreferenced with the :PRINT command to send the file COMPANY to the line printer.

5

Using SORT-MERGE/XL in Batch Mode

This chapter discusses using SORT-MERGE/XL in batch mode. It shows how to build a job file, initiate it, schedule it, and terminate it if necessary.

You can create a job file containing SORT-MERGE/XL commands and run it in batch mode. A batch job, which can contain one or more commands, is executed independently of your interactive session. This frees your terminal for other processing. This technique is known as streaming a job and is initiated with the MPE XL `:STREAM` command. The section below, "Initiating a Batch Job" discusses streaming a job file.

Building a Job File

A job file is created using an editor text processor such as EDIT/V. The first line of the job file must begin with the `:JOB` command. This is followed by the logon to the appropriate *user.account,group*. The logon must include all necessary passwords. A batch job is terminated with the `:EOJ` command. For additional information on the `:JOB` and `:EOJ` commands refer to the *MPE XL Commands Reference Manual* (32650-90003).

When creating a batch job, use a substitute character for the MPE XL colon prompt. MPE XL expects the exclamation point (`!`) as the substitute prompt, but you may choose another special character for this purpose. This substitute prompt must appear in column one of each record, followed by the remainder of the command.

It is not necessary to indicate subsystem prompts or use a substitute character for them. Notice in the example below that the SORT/XL subsystem chevron (`>`) prompt does not appear before the SORT/XL `INPUT`, `OUTPUT`, `KEY`, or `END` commands.

The following job file logs into a *group* of a *user.account*, enters the SORT/XL subsystem, initiates a multiple key sort on the input file `EMPLOYEE`, creates the output file `COMPANY`, and accesses EDIT/V to list the contents of the file. The job is then printed on the system printer. This job file was created using EDIT/V, but other editors may be used. Check with your System Manager to determine which editors are available on your system.

```
:RUN EDITOR.PUB.SYS
```

```
HP 32201A.07.17 EDIT/3000 WED, JUN  3, 1987  1:00 PM
© HEWLETT-PACKARD CO. 1985
/A
  1  !JOB JOBNAME,USER/PASSWORD.ACCOUNT/PASSWORD,GROUP
  2  !SORT
  3  INPUT EMPLOYEE
  4  OUTPUT COMPANY
  5  KEY 1, 11; 12, 11
  6  END
  7  !EDITOR
  8  TEXT COMPANY
  9  LIST ALL
 10  EXIT
 11  !EOJ
 12  //
/KEEP RUNSORT
```

You have created a job file named RUNSORT. It may be used to generate reports on the data in the input file EMPLOYEE. It can also be used to generate a sort on any other input file by modifying the lines in the job file containing the INPUT and TEXT commands (see lines 3 and 8 in the example above).

Every time you run this job, you need to modify the line containing the name of the output file. If a unique name is not supplied, the system issues the message:

```
OUTPUT FILE CLOSED WITH FILENAME OUTPUT1
```

This is followed by the SORT/XL statistics, and then another message:

```
PROGRAM TERMINATED IN AN ERROR STATE.   (CIERR 976)
REMAINDER OF JOB FLUSHED.
CPU SEC. = 3. ELAPSED MIN. = 1. WED, JUN  3, 1987,  1:05 PM
```

If the system finds an existing file name for the output file, it supplies the file name OUTPUT1, OUTPUT2, ... OUTPUT*n* by successive ascending numbers and then aborts the job in an error state.

If this occurs, either purge the existing file COMPANY, or provide a unique name for the file designated by OUTPUT. Be sure the file designated by >OUTPUT and the file accessed by the editor are the same.

Initiating a Batch Job

To send the job file RUNSORT to the computer for processing, exit the editor subsystem. Then at the MPE XL colon prompt (:), enter:

```
:STREAM RUNSORT
```

Once this command has been received by the system, it issues a job number identifying your job, as follows:

```
#J555
```

The following section explains how to use this number to check on the status of your job.

Checking on the Status of your Job

To check on the status of your job, use the system assigned job number and enter the :SHOWJOB command, as follows:

```
:SHOWJOB #J555
```

If the job has not completed processing, the following message appears:

```
JOBNUM  STATE  IPRI  JIN   JLIST  INTRODUCED  JOB NAME

#J555   EXEC          10S  LP     WED 10:46A  RUNSORT,USER.ACCOUNT

JOBFENCE= 5;  JLIMIT= 10; SLIMIT= 20
```

If your job has completed processing, the following message appears:

```
NO SUCH JOB(S)
JOBFENCE= 5; JLIMIT= 10; SLIMIT=20
```

If your job has completed processing, go to the system printer and get the printout of your report. For additional information on the :SHOWJOB command, refer to the *MPE XL Commands Reference Manual* (32650-90003).

Scheduling a Batch Job

The :STREAM command can be used to schedule the job for execution at any given time (tonight, tomorrow, next week, or next month).

To schedule a job for execution at a particular time, use the MPE XL :STREAM command. For example, you have completed the job file and would like to have a copy of your report waiting for you by tomorrow morning. To run the job during the night, when the system might not be fully utilized, issue the following command:

```
:STREAM RUNSORT;AT=22:00
```

This command schedules the job RUNSORT to execute at 10:00 PM of the day the command was issued. The system responds by assigning a job number, as follows:

```
#Jnnn
```

Enter the :SHOWJOB command to ensure that the job is scheduled:

```
:SHOWJOB #Jnnn
```

The systems responds with the message:

```
JOBNUM  STATE  IPRI  JIN   JLIST  SCHEDULED-INTRO  JOB NAME

#Jnnn   SCHED     8   10S   LP     6/3/87 22:00     RUNSORT,USER.ACCT

1 SCHEDULED JOB(S)
```

For additional information on using the MPE XL :STREAM command to schedule jobs refer to the *MPE XL Commands Reference Manual* (32650-90003).

Note

Scheduled jobs may, or may not, survive a system shutdown and subsequent startup, depending on system events. For this reason, it is best to schedule jobs no more than a few days in advance.

Terminating a Batch Job

If you send a job to the system and then decide you don't want it to execute, you can abort it with the :ABORTJOB command. To check on the status of the job, enter a :SHOWJOB command and look at the left column on the screen. If the job number is not listed, the job has completed executing and you can pick up the printout from the printer. If the number is listed, enter the following command at the MPE XL prompt:

```
:ABORTJOB #Jnnn
```

Refer to the *MPE XL Commands Reference Manual* (32650-90003) for additional information on the :ABORTJOB command.

6

SORT-MERGE/XL Commands

This chapter describes the SORT-MERGE/XL commands used to execute the sort or merge operations you want to perform on files. In previous chapters you were exposed to the basic aspects of some commands, such as `>INPUT`, `>OUTPUT`, `>KEY`, and `>END`.

The SORT-MERGE/XL commands in this chapter are listed in alphabetical order. All SORT-MERGE/XL commands are identical for both subsystems, with the exception of the `>INPUT` and `>OUTPUT` commands.

The SORT/XL and MERGE/XL subsystems can be used during an interactive session or in a batch job. In an interactive session, commands are entered at the subsystem chevron (`>`) prompt. When large amounts of input and output are involved, it may be more convenient to run the program as a batch job by streaming it from a terminal, and scheduling it to run at a time when the system is not being fully utilized.

When the length of a command exceeds one line, enter an ampersand (`&`) as the last nonblank character on the line to continue the command to a second line or subsequent lines. In an interactive session, both SORT/XL and MERGE/XL prompt you for the rest of the command with the double-chevron (`>>`) prompt as shown in the following example:

```
:SORT
HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  8:00 AM
© HEWLETT-PACKARD CO. 1986

>INPUT SORT1, SORT2, SORT3, SORT4, ... SORTn&
>>
```

The SORT-MERGE/XL commands described in this chapter are listed below:

```
>ALTSEQ                  >OUTPUT (MERGE/XL)
>DATA                    >OUTPUT (SORT/XL)
>END                     >RESET
>EXIT                    >SHOW
>INPUT (MERGE/XL)        >VERIFY
>INPUT (SORT/XL)         >:(MPE Command)
>KEY                     >:EOD
>LANGUAGE
```

For information on possible error messages you may encounter while using SORT-MERGE/XL commands, refer to Appendix A.

ALTSEQ

The >ALTSEQ command defines a collating sequence other than the standard ASCII or EBCDIC format. The >ALTSEQ command must be preceded by a >DATA command. It is effective only if the keys are of *type* BYTE and if the input data is ASCII. (Refer to Appendix B for information on ASCII and EBCDIC character set values.)

SYNTAX

```
>A[LTSEQ] modspec1[, modspec2]...[, modspecN]
```

PARAMETERS

modspec

A set of parameters you use to define your own collating sequence. You can use more than one group of these parameters in one or more successive >ALTSEQ commands until the desired collating sequence is defined.

The *modspec* parameter set has the following form:

```
                { = }
[EACH] leftspec {   } rightspec
                {WITH}

                    or

                  {WITH}
MERGE    leftspec {   } rightspec
                  { = }
```

To specify *leftspec* and *rightspec* use the following form:

```
{string       }
{num byte     }
{range string }
```

EACH

The EACH parameter indicates that the collating sequence is to be modified by assigning each character of *leftspec* the ordinal value obtained by taking the ASCII code decimal value of the corresponding character in *rightspec*. If *leftspec* is longer than *rightspec*, *rightspec* is concatenated to itself enough times to make it equal in length to *leftspec*.

MERGE

The MERGE parameter indicates that the collating

merging *leftspec* and *rightspec*. Characters are selected alternatively from *leftspec* and *rightspec*.

Note

If neither EACH nor MERGE is specified, the collating sequence is modified as if EACH was specified, but *rightspec* is padded with blanks if it is shorter than *leftspec*.

`=`	When used in the *modspec* parameter, the equal sign (`=`) functions as a separator between *leftspec* and *rightspec*.
`WITH`	The `WITH` parameter can be used interchangeably with the equal sign (`=`) and is generally used when `MERGE` is specified.
string	A *string* is a single character or a group of ASCII or EBCDIC characters specified by enclosing them in quotation marks, for example, "`J`" or "`JAS`".
num byte	A numerical specification used in the following form: `[%[(`*bb*`)]]` *nnn* The *bb* is a base of any decimal number between 2 and 16 inclusive. You specify *%(bb)* to indicate a base other than 8 or 10. The *%* indicates base 8 when no (*bb*) is specified. If both *%* and (*bb*) are omitted, the *nnn* parameter is assumed to be a decimal number (that is, base 10). The *nnn* represents a number (integer) with a value between 0 and decimal 255, inclusive. Each *%n* is a digit between 0 and 9, inclusive, or one of the letters A, B, C, D, E, or F. The letters A through F are used to represent the digits 10 through 15, when a base greater than 10 is used. Each digit *n* or *nnn* must be less than the base *bb*. For example, 12 represents the decimal value 12; %12 represents the octal value 12, which is equivalent to the decimal value 10; and %(16)12 represents the hexadecimal value 12, which is equivalent to the decimal value 18.
range string	Specifies two characters separated by a minus sign (`-`) and enclosed in quotation marks, or two numeric byte specifications separated by a minus sign. For example, "A-Z" or %101-%132 (which is the octal specification for the character range "A-Z").

Note

Whenever a minus sign (-) is the second character in a group of three characters, the group is treated as a range. In all other cases, the minus sign is treated the same as any other character. For example, "A-D" represents the four characters A, B, C, and D while "AD-" represents the three characters A, D, and -.

DISCUSSION

Each modification of the collating sequence changes the ordinal values in the translation table assigned to the characters specified by *leftspec*. Refer to the `>SHOW` command for a discussion of the translation table. If *rightspec* is longer than *leftspec*, the extra characters are ignored. If *leftspec* is longer than *rightspec* and neither `EACH` nor `MERGE` has been specified, *rightspec* is padded with blanks to make it equal in length to *leftspec*. For example, the command, `>ALTSEQ "SAW"="TG"` gives `S`, `A`, and `W` the ordinal values `T`, `G`, and space. (See the discussion below for explanations of *modspec* with `EACH` and `MERGE`.) These assignments of new ordinal values are only for collating purposes. That is, the identity of the character is not lost; data is unchanged and appears in its original form in the output.

You must issue a `>DATA` command, specifying data type and a collating sequence type before you can use the `>ALTSEQ` command in any SORT/XL or MERGE/XL operation. The system displays the error message `THE DATA COMMAND MUST BE ISSUED BEFORE THE ALTSEQ COMMAND CAN BE ISSUED`, if the `>ALTSEQ` command is not preceded by a `>DATA` command.

Note

The operation of SORT/XL (or MERGE/XL) is slower when you define a collating sequence with the `>ALTSEQ` command than when a standard ASCII or EBCDIC collating sequence is used.

Using modspec With EACH

If `EACH` is specified, the modifications of the collating sequence are the same as explained above, except if *leftspec* is longer than *rightspec*, *rightspec* is concatenated to itself a sufficient number of times to make it equal in length to *leftspec*. For example, the command, `>ALTSEQ EACH "ADW"="FG"`, give `A`, `D`, and `W` the ordinal values obtained by taking the ASCII code decimal values of `F`, `G`, and `F`. Assuming the basic collating sequence has been specified as ASCII, this means A=70 appears in the sixth row, fifth column of the translation table, D=71 in the sixth row, eighth column, and W=70 in the eighth row, seventh column. Note that 70 and 71 are the ASCII code decimal values of the characters F and G, respectively. For additional information refer to the "EXAMPLES" section below.

Using modspec With MERGE

When `MERGE` is specified in the *modspec* parameter, the values in the translation table assigned to the characters specified by *leftspec* and *rightspec*, and the characters in between are modified. Characters are selected alternatively from *leftspec* and *rightspec* and the translation table is modified so the characters collate in this order. The first character is always selected from *leftspec*. If *leftspec* precedes *rightspec* in the collating sequence, the sequence is modified so the characters between the two ranges collate after the merger of the ranges. If *rightspec* precedes *leftspec*, the characters between the two ranges collate before the first character of the first range. When either range is exhausted, the characters from the other range are simply appended until that range is also exhausted. Note that the

strings specified by *leftspec* and *rightspec* must be strictly increasing and contiguous whenever MERGE is specified.

If you wish to do an alphabetic sorting in which each upper case letter collates ahead of the corresponding lower case letter, use the command >ALTSEQ MERGE "A-Z" WITH "a-z". The following six special characters follow the lower case z since the first range precedes the second range:

[\ ] ^ _ and ‘

If the *modspec* is MERGE "a-z" WITH "A-Z", the same six characters precede the lower case a. For additional information, refer to the "EXAMPLES" section below.

Consider this form of *modspec* as a shorthand for the *modspec* specifying EACH. For example, the command, >ALTSEQ MERGE "A-Z" WITH "a-z", is equivalent to the longer command >ALTSEQ "AaBb ... Zz"= "AB ... Zab ... z", where ... represents all the necessary characters.

EXAMPLES

The following examples show how to use various parameters with the >ALTSEQ command, as well as the resulting collating sequences.

Standard ASCII Collating Sequence

To display the standard collating sequence enter the DATA IS ASCII, SEQUENCE IS ASCII and >SHOW SEQUENCE commands, as shown below. Refer to this display, for comparative purposes, to see what occurs to the collating sequence when you use >ALTSEQ for various functions in the following examples.

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  8:10 AM
©: HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>SHOW SEQUENCE


     nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
     dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
      sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
       0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
       @   A   B   C   D   E   F   G   H   I   J   K   L   M   N   O
       P   Q   R   S   T   U   V   W   X   Y   Z   [   \   ]   ^   _
       ‘   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
       p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

Using the EACH Parameter

The following example shows how to use the >ALTSEQ command with the EACH parameter followed by a *string* specification:

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  8:10 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>ALTSEQ EACH "LMN"="ST"
>SHOW SEQUENCE
```

```
nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
 sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
  0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
  @   A   B   C   D   E   F   G   H   I   J   K   0   P   Q   R
 L=  N=   S  M=   T   U   V   W   X   Y   Z   [   \   ]   ^   -
  ‘   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
  p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

The result of the *modspec* in the above example where EACH "LMN"="ST" is shown below:

```
        Original List                Sorted Result

        TOKEN                        COST
        MOP                          COME
        COST                         SING
        COME                         NOSE
        TABLE                        LONESOME
        MISS                         SOLE
        SING                         TABLE
        NOSE                         MISS
        LONESOME                     TOKEN
        SOLE                         MOP
```

During the sort operation, L and N are equated to S, and M is equated to T.

Using >ALTSEQ Without the EACH Parameter

The following example shows how to use the >ALTSEQ command without including the EACH parameter. You may use abbreviated forms for >ALTSEQ (>A), >SHOW SEQUENCE (>SH S), and SEQUENCE IS ASCII (SEQ A), if you wish.

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  8:15 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>ALTSEQ "ABC" = "X"
>SHOW SEQUENCE


     nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
     dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
      sp=  B=  C   !   "   #   $   %   &   ’   (   )   *   +   ,   -
       .   /   0   1   2   3   4   5   6   7   8   9   :   ;   <   =
       >   ?   @   D   E   F   G   H   I   J   K   L   M   N   O   P
       Q   R   S   T   U   V   W   A=  X   Y   Z   [   \   ]   ^   _
       ‘   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
       p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

The >ALTSEQ command pads X with two blank characters making it equal to ABC in length. Note the character sp (space) is equated to B and C and the character A is equated to X. The table position identified by each character of the left string is replaced by the corresponding character of the right string until the string ABC is exhausted.

Numeric Byte Specification

The following example shows how to use the >ALTSEQ command for a numeric byte specification:

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  8:20 PM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>ALTSEQ 65=%141
>SHOW SEQUENCE
```

```
nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
 sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
  0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
  @   B   C   D   E   F   G   H   I   J   K   L   M   N   O   P
  Q   R   S   T   U   V   W   X   Y   Z   [   \   ]   ^   -   ‘
 A=   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
  p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

In this example, the upper case A (represented by the decimal value 65) is assigned the same ordinal value as the lower case a (represented by the octal value %141) in the final collating sequence.

Using a Range String Specification

The following example shows how to use the >ALTSEQ command for a *range string* specification:

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  8:25 AM
© HEWLETT-PACKARD CO. 1986

>ALTSEQ %101-%132="a-z"
>SHOW SEQUENCE
```

```
nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
 sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
  0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
  @   [   \   ]   ^   _   ‘   A=  a   B=  b   C=  c   D=  d   E=
  e   F=  f   G=  g   H=  h   I=  i   J=  j   K=  k   L=  l   M=
  m   N=  n   O=  o   P=  p   Q=  q   R=  r   S=  s   T=  t   U=
  u   V=  v   W=  w   X=  x   Y=  y   Z=  z   {   |   }   ~ del
```

The left range in the above example is specified by two numeric byte specifications separated by a minus sign. Note that the same range can be represented by "A-Z" (characters), %101-"Z" (octal representation), or 65-90 (decimal representation).

Collating Upper Case Before Lower Case

The following example shows how to use the >ALTSEQ command for collating upper case, then lower case characters. This is a commonly used alternative to the standard collating sequence.

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  8:30 AM
© HEWLETT-PACKARD CO. 1986

>ALTSEQ MERGE "A-Z" WITH "a-z"
>SHOW SEQUENCE
```

```
nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
 sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
  @   A   a   B   b   C   c   D   d   E   e   F   f   G   g   H
  0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
  h   I   i   J   j   K   k   L   l   M   m   N   n   O   o   P
  p   Q   q   R   r   S   s   T   t   U   u   V   v   W   w   X
  x   Y   y   Z   z   [   \   ]   ^   _   ‘   {   |   }   ~ del
```

The six characters [, \,], ^, _," and ‘ follow the lower case z. The result of MERGE "A-Z" WITH "a-z" is as follows:

Original List	Sorted List Without MERGE	Sorted List Using MERGE
CAN	AXE	AXE
shovel	BROOM	BROOM
MAN	CAN	boy
BROOM	DOG	CAN
TABLE	MAN	DOG
AXE	TABLE	drawer
drawer	boy	MAN
boy	drawer	shovel
DOG	shovel	TABLE

Collating Lower case Before Upper Case

The following shows how to use the >ALTSEQ command to collate lower case alphabetic characters, and have each followed by its corresponding upper case character:

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  8:35 AM
© HEWLETT-PACKARD CO. 1986

>ALTSEQ MERGE "a-z" = "A-Z"
>SHOW SEQUENCE
```

```
nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
 sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
  0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
  @   [   \   ]   ^   _   ‘   a   A   b   B   c   C   d   D   e
  E   f   F   g   G   h   H   i   I   j   J   k   K   l   L   m
  M   n   N   o   O   p   P   q   Q   r   R   s   S   t   T   u
  U   v   V   w   W   x   X   y   Y   z   Z   {   |   }   ~ del
```

The six characters [, \,], ^, _, and ‘’ precede the lower case a.

The result of MERGE "a-z" = "A-Z" is as follows:

Original List	Sorted List Without MERGE	Sorted List Using MERGE
CAN	AXE	AXE
shovel	BROOM	boy
MAN	CAN	BROOM
BROOM	DOG	CAN
TABLE	MAN	drawer
AXE	TABLE	DOG
drawer	boy	MAN
boy	drawer	shovel
DOG	shovel	TABLE

Merging Unequal Strings

The following example shows how to use the >ALTSEQ command to merge unequal strings:

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  8:40 AM
© HEWLETT-PACKARD CO. 1986

>ALTSEQ MERGE "ABCD" WITH "ab"
>SHOW SEQUENCE
```

```
nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
 sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
  0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
  @   A   a   B   b   C   D   E   F   G   H   I   J   K   L   M
  N   O   P   Q   R   S   T   U   V   W   X   Y   Z   [   \   ]
  ^   _   ‘   c   d   e   f   g   h   i   j   k   l   m   n   o
  p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

The collating sequence appears AaBbCDE ... Z. The merging of the strings continues until the right string ab is exhausted.

ADDITIONAL DISCUSSION

Refer to the >DATA and >SHOW commands in this chapter.

DATA

The >DATA command specifies the type of the input data (either ASCII or EBCDIC) and the basic collating sequence to be used in the particular SORT/XL (or MERGE/XL) operation. The collating sequence may be altered, if desired, by using the >ALTSEQ command.

SYNTAX

```
>DATA [IS] {A[SCII] } [ , ] SEQ[UENCE] [IS] {A[SCII  }
           {E[BCDIC]}                       {E[BCDIC]}
```

DISCUSSION

The >DATA command must precede the first >ALTSEQ command found in any SORT/XL or MERGE/XL operation since it is the command that initiates the translation table. If the system encounters an >ALTSEQ command before a >DATA command, the message, THE DATA COMMAND MUST BE ISSUED BEFORE THE ALTSEQ OR SHOW COMMANDS, is displayed. If the >DATA command is entered again, following an >ALTSEQ command, the translation table (and the collating sequence) are reset to their original status.

When you specify a particular sequence, it is for collating purposes only. A user-defined sequence can be designated only if the input data is ASCII. The input data is unchanged and appears in the output in its original form. In the example below, the >DATA command nullifies the effect of the >ALTSEQ command issued previously during a SORT/XL operation.

EXAMPLES

The following example shows what occurs if you do not enter the >DATA command before an >ALTSEQ command. It also shows how the >DATA command nullifies the effect of the >ALTSEQ command issued previously during a SORT/XL operation.

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  9:25 AM
© HEWLETT-PACKARD CO. 1986

>ALTSEQ MERGE "A-T" WITH "V-Y"
THE DATA COMMAND MUST BE ISSUED BEFORE THE ALTSEQ OR SHOW COMMANDS
>DATA IS ASCII, SEQUENCE IS ASCII
>ALTSEQ MERGE "A-T" WITH "V-Y"
>SHOW SEQUENCE


     nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
     dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
      sp   !   "   #   $   %   &   '   (   )   *   +   ,   -   .   /
       0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
       @   A   V   B   W   C   X   D   Y   E   F   G   H   I   J   K
       L   M   N   O   P   Q   R   S   T   U   Z   [   \   ]   ^   _
       `   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
       p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del

>DATA IS ASCII, SEQUENCE IS ASCII
>SHOW SEQUENCE

     nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
     dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
      sp   !   "   #   $   %   &   '   (   )   *   +   ,   -   .   /
       0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
       @   A   B   C   D   E   F   G   H   I   J   K   L   M   N   O
       P   Q   R   S   T   U   V   W   X   Y   Z   [   \   ]   ^   _
       `   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
       p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

ADDITIONAL DISCUSSION Refer to the >ALTSEQ command in this chapter.

END

The >END command specifies the conclusion of SORT-MERGE/XL parameters. It also starts the sort or merge operation specified.

SYNTAX

```
>E[ND]
```

DISCUSSION

The >END command indicates all commands have been specified and the SORT/XL or MERGE/XL program should begin operation.

If the terminal ($STDIN) is specified in the >INPUT command of the SORT/XL program, you enter and receive sort data from the terminal. A work file is not created. The character ? is displayed following the >END command, and the input records are typed in from the terminal.

After the >END command is issued, in an interactive session or batch job, the sort or merge operation is started. This is followed by a statistical report on the operation that is displayed on your terminal screen. This report is followed by the message END OF PROGRAM and the MPE XL colon prompt (:).

EXAMPLES

The following example shows how to use the >END command in an interactive session:

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  9:30 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 1,10
>END

                    STATISTICS
NUMBER OF RECORDS =                    6
NUMBER OF INTERMEDIATE PASSES =        0
SPACE AVAILABLE (IN WORDS) =      27,047
NUMBER OF COMPARES =                  13
NUMBER OF SCRATCHFILES IO'S =          6
CPU TIME (MINUTES) =                 .00
ELAPSED TIME (MINUTES) =             .49
RECORD SIZE (IN BYTES) =              80
SCRATCH FILE SIZE (# SECTORS) =    3,502

END OF PROGRAM
:
```

The following example shows what occurs when the >END command is entered after the terminal has been designated as the input device and output device. The terminal is designated as the input device by specifying * (for $STDIN) in the >INPUT command. It is designated as the output device by specifying $STDLIST in the >OUTPUT command.

```
>INPUT *
>OUTPUT $STDLIST
>KEY 1, 4
>END
?user input
?user input
?user input
?:EOD
sorted data
sorted data
sorted data
```

ADDITIONAL DISCUSSION

None.

EXIT

The >EXIT command terminates the operation of SORT/XL or MERGE/XL and exits the subsystem.

SYNTAX

```
>EX[IT]
```

DISCUSSION

The >EXIT command terminates access to the SORT-MERGE/XL subsystem. Once this command is entered, no further program execution is performed.

EXAMPLE

The following example shows how to use the >EXIT command to terminate operation of the SORT/XL subsystem. The sort is not performed and the designated output file, COMPANY is not created.

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  9:35 AM
©: HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 1, 9
>EXIT

END OF PROGRAM
:EDITOR
HP32201A.07.17 EDIT/3000 THU, JUN  4, 1987,  9:40 AM
©: HEWLETT-PACKARD CO. 1985
/TEXT COMPANY, UNNUMBERED

+-F-I-L-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y+
!  ERROR NUMBER: 52    RESIDUE: 0                !
!  BLOCK NUMBER: 0            NUMREC: 0          !
+---------------------------------------------------+
*23*FAILURE TO OPEN TEXT FILE   (52)
```

ADDITIONAL DISCUSSION

None.

INPUT (MERGE/XL)

Within the MERGE/XL subsystem, the >INPUT command specifies the sorted files to be merged. Refer to the SORT/XL >INPUT command for information on how to use the command within that subsystem.

SYNTAX

>I[NPUT] {*fname1,fname2*}[,*fname3*]...[,*fnameN*]

PARAMETERS

fname — The *fname* parameter specifies the actual file designator. $NULL is not a valid input file. The order in which the input files are specified is important in that records with equal keys are positioned according to the order of the files in which they appear.

Notice that parentheses are not used with >INPUT command in MERGE/XL.

DISCUSSION

Unlike the SORT/XL >INPUT command, the input files are not enclosed in parentheses. The order in which the files are specified is important only in that the records with equal keys are ordered according to the order of the files in which they appear. If more than one >INPUT command is entered, only the last command is effective. See the discussion under the SORT/XL >INPUT command for information about file equations.

EXAMPLE

The following example shows how to merge three previously sorted files, SORTED1, SORTED2, and SORTED3, into a single, new, output file named MERGE1:

```
:MERGE

HP32214A.01.00  MERGE/3000 THU, JUN  4, 1987,  9:45 AM
© HEWLETT-PACKARD CO. 1986

>INPUT SORTED1, SORTED2, SORTED3
>OUTPUT MERGE1
>KEY 1, 11
>END
```

ADDITIONAL DISCUSSION

Refer to the SORT/XL >INPUT command in this chapter.

INPUT (SORT/XL)

Within the SORT/XL subsystem, the `>INPUT` command specifies the input file(s) to be sorted. Refer to the MERGE/XL `>INPUT` command for information on how to use the command within that subsystem.

SYNTAX

```
                {$STDIN [ X ]                }
>I[NPUT]        {*                           }[,#records][,rec size]
                {fname                  }
                {(fname1, fname2,...fnameN )}
```

PARAMETERS

`$STDIN [ X ]`
Specifies that the input records are to be read from `$STDIN[ X ]`. In interactive mode a question mark (`?`) prompt is displayed.

`*`
Entering an asterisk (`*`) in an interactive session specifies that the input records are read from the text file `TEXT`. In this case, the input records are to follow the `>END` command, and the list of records is to be terminated with `:EOD`.

Recall that `TEXT` is the formal file designator of the file containing the SORT/XL commands and that it defaults to `$STDIN`. Therefore, if no file equation has been entered against the file `TEXT`, the input records are entered from the terminal in interactive mode and are included in the stream file in batch mode. (The prompt "`?`" is displayed for each record in interactive mode.) If a file equation has been issued, then the records should be part of the file equated to `TEXT` (again following the `>END` command).

fname
Specifies the actual file designator. `$NULL` is not a valid input file.

#records
The *#records* parameter should be specified only if one or more of the input files is not on disc. It is a positive integer specifying the upper limit of the number of records sorted. If multiple input files are specified, it is the total number of records from all input files. When all input files are on disc, the current end-of-file (EOF) values are used and *#records* is ignored. If all input files are not on disc and *#records* is not specified, a default value of 10,000 is assumed by SORT/XL. This parameter cannot be used to extract a subset of the input file.

rec size	A positive integer specifying the maximum allowable number of bytes in a record. This parameter may be used to set the record size of the output file, but is used mainly for files containing variable-length records. When sorting such files, this parameter should be set to the size of the largest record present in the input. If *rec size* is not specified when sorting variable-length record files, SORT/XL will use the block size as the maximum record size. This could result in more space than necessary being used for the scratch file, as well as causing some degradation of performance.

MPE XL SORT has two sets of scratch files. If the sort takes place in compatibility mode, there is one scratch file for which the size is computed as in MPE V/E SORT. If the sort takes place in native mode, there are two (mapped) scratch files. Both computations are described below.

Compatibility Mode Scratch File Size

If you want to extimate the scratch file record size (SFRS) and the scratch file size (SFS), use the following equations:

```
SFRS+((rec size + 7)/2) + 4
```

where *rec size* is the input record size in bytes. (You must add the length of the keys to the *rec size* if the keys are of the *type*, BYTE, and ALTSEQ is used.) SFRS is in words.

```
SFRS+((SFRS*#records )/128) + 1
```

SFS is in sectors.

Compatibility Mode Scratch Filename

You can issue a file equation for the scratch file only to specify a particular logical device which must be a disc. For example:

```
FILE SORTSCR; DEV=2
```

Native Mode Scratch File Size

Native mode scratch files contain two types of records: Work Records and End-of-Subfile Records. The following algorithm calculates the size of one native mode scratch file.

```
Work Record Length (#Work Records + #End-of-Subfile Records)
                    = #Bytes in 1 scratch file
Where:
                  #Work Records = #Input Records

               #End-of-Subfile Records = #Input Records / 50

and Work Record Length = Input Record Length + Expansion Bytes
```

The value of Expansion Bytes depends on the number and type of keys that the user specifies. Expansion Bytes is expressed as:

```
#key 5's + #Key 9's + 2(#key 4's + #key 6's + 3key 7's +#key 8's +#key 12's) + 3(#key 13's) +3
```

The space used for a sort occuring in native mode will be two times the value returned by this formula.

Note

This formula illustrates a worst case situation.

Native Mode Scratch Filenames

You cab issue file equations for the native mode scratch files only to specify a particular device which must be a disc. For example:

```
FILE HPSORTS1; DEV=2

FILE HPSORTS2; DEV=2
```

DISCUSSION

When specifying more than one input file to SORT/XL, the list of files must be enclosed in parentheses. This differs from the use of the `>INPUT` command for MERGE/XL, where parentheses cannot be used. If more than one `>INPUT` command is entered, only the last command is effective. Thus, all the files to be sorted must be specified in a single `>INPUT` command. This command can be entered any time before the `>END` command. In the absence of the `>INPUT` command, any disc file with the formal designator `>INPUT` is considered the input file. Also, file equations may be issued before entering or during either subsystem. Thus, if the `>INPUT` command refers to the same file as specified in a file equation, the file's characteristics are determined by the file equation. The user issues the `>RESET` command before entering SORT/XL or MERGE/XL if the default values for the parameters of the file are desired. The same holds for the `>OUTPUT` command for SORT/XL and the `>INPUT` and `>OUTPUT` command for MERGE/XL.

EXAMPLE

In the following example, the file `EMPLOYEE` is to be sorted with a maximum of 30 characters from each record:

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987,  9:50 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE, 30
```

ADDITIONAL DISCUSSION

See the `>INPUT` command for MERGE/XL in this chapter.

KEY

The >KEY command specifies the location of the key data items in a file's records which are to be sorted or merged.

SYNTAX

>K[EY] *keyspec1* [; *keyspec2*]...[; *keyspecN*]

PARAMETERS

keyspec
: A group of parameters used to specify a key data item to be sorted or merged. The syntax of the *keyspec* parameters follows:

  position, length [,*type*][,DESC]

position
: A positive number (integer) specifying the position of the first character of the key data item within the record. (The first position of a record line is numbered one.)

length
: A positive number (integer) indicating the length of the data item key field in bytes.

type
: Defines the type of data contained in the data item key field. The type of data can be one of the following:

B[YTE]
: A direct byte comparison is used. It is the default value for the *type* parameter and should be used for ASCII or EBCDIC data. The specification of an alternate collating sequence via the >DATA and >ALTSEQ commands affects the collating of this key type only.

C[HARACTER]
: The collating sequence for the native language defined in the >LANGUAGE command is used. If no >LANGUAGE command has been issued, SORT/XL and MERGE/XL use the default data language of the system (usually ASCII). Refer to the *Native Language Programmer's Guide* (32650-90022) for additional information on the default data language.

I[NTEGER]
: The key data item field contains a two's complement number of the specified length in bytes. Any value may be specified for *length*. The *length* parameter defaults to two bytes.

R[EAL]
: The key data item field contains a floating-point number in standard HP 3000 format. Any value may be entered for *length*. The *length* parameter defaults to four bytes.

`L[ONG]`	`LONG` is the same as `REAL` except that *length* defaults to eight bytes.
`F[POINT]`	The key data item field contains a floating-point number in IEEE standard format. The *length* parameter defaults to four bytes. Any value may be entered for *length*.

Note

NANs (Not A Number) will collate at the beginning or end for IEEE floating-point numbers. The method of reporting (or ignoring) these entities has not been determined.

F8[POINT]	Same as FPOINT except that *length* defaults to eight bytes.
F16[POINT]	Same as FPOINT except that *length* defaults to sixteen bytes.
T[WO-BYTE]	Key data item field contains 16-bit data. The *length* specified for this key type must be an even number of bytes.
P[ACKED]	Key data item field contains a packed decimal number. In this format, each character except the last contains two digits. Each digit occupies four bits. The rightmost character contains the least significant digit of the number in its four leftmost bits, and the sign of the number in its four rightmost bits. The sign is considered minus if it has the value 1101 (binary) and plus otherwise.
PACKED*	Same as PACKED except there are an even number of digits and a sign. The leftmost four bits are not treated as part of the field.
DI[SPLAY-TRAILING-SIGN]	Key field contains a numeric display quantity. Numeric display items are represented by ASCII-coded decimal digits (0 through 9) except for the rightmost digit, which carries the sign of the data item. The sign is determined according to the table shown in Figure 6-1. (Sign is optional.) For example, 123 is represented by 12C.
DISPLAY-L[EADING-SIGN]	In this case, the leftmost digit carries the sign of the data item. For example, -123 is represented by J23. Refer to the table shown in Figure 6-1. (Sign is optional.)
DISPLAY-TRAILING-SIGN-S[EPARATE]	The sign is contained in the character position to the right of the rightmost digit. For example, 123 is represented by 123+. (Sign can be blank.)
DISPLAY-LEADING-SIGN-S[EPARATE]	The sign is contained in the character position to the left of the leftmost digit. For example, -123 is represented by -123. (Sign can be blank.)
DESC	Indicates the records are to be arranged in descending order. If this parameter is not specified, the records are arranged in the default ascending order.

Display Digit	Positive	Negative	No Sign
0	{ (%173)	} (%175)	0 (%60)
1	A (%101)	J (%112)	1 (%61)
2	B (%102)	K (%113)	2 (%62)
3	C (%103)	L (%114)	3 (%63)
4	D (%104)	M (%115)	4 (%64)

5	E (%105)	N (%116)	5 (%65)
6	F (%106)	O (%117)	6 (%66)
7	G (%107)	P (%120)	7 (%67)
8	H (%110)	Q (%121)	8 (%70)
9	I (%111)	R (%122)	9 (&71)

DISCUSSION

SORT-MERGE/XL sorts keys contain binary, ASCII, or EBCDIC data according to an eight-bit binary sequence (00000000 to 11111111), except for the type `CHARACTER`, which is sorted according to the collating sequence of the native language specified in the `>LANGUAGE` command. Refer to Appendix C for further information on native language collating. Other types of data (integer, real, etc.) are sorted according to the standard arithmetic relational operators. For example, 2 is greater than -5. The keys can contain alphabetic, numeric, or alphanumeric (alphabetic and numeric intermixed) data. They can be contiguous or separated in a record or they can overlap each other, provided the collating sequence is not altered, or a user-defined sequence is not used. An entire record can be considered as a single key.

As explained in Chapter 3, each `>KEY` command can specify one or more key fields which are separated by semicolons. Multiple key fields can also be specified with more than one `>KEY` command. All the key fields do not have to be specified in the same command. The most significant key is called the major key and is declared first in the command. Other keys have decreasing significance according to their relative positions following the major key. They are compared if a comparison or more significant keys results in an equal condition.

Consider a file containing the records of all the students in a high school. Each record can contain information such as name, address, grade level, grades in individual courses, as well as data on other information. You can specify the order in which the records are sorted. If the first record is of the student with the highest grades (A) in English and Math, you specify an ascending order. If the major key is English and the other key is Math, the data in the Math fields are compared only if the data in the English fields are the same. The sorting order is specified in the same commands that specify the keys. An order is declared for each key. This order does not have to be the same for all the keys in a record. For example, in the high school file, you can declare English (major key) with an ascending order and Math with a descending order. Note even if the sorting order is different for each key, only one collating sequence is used for a particular operation.

EXAMPLES

The following examples show using the >KEY command and some of its options:

```
>KEY 10, 5
```

BYTE key of length 5 starting in position 10, sorted in the ascending order.

```
>KEY 20, REAL
```

REAL key of length 4, starting in position 20 and sorted in an ascending order since four is the default for the *length* parameter when the key data type is REAL.

```
>KEY 30, 20, INT, DESC
```

20-byte integer key starting in position 30, and sorted in a descending order.

For information on making corrections to the key specification, refer to the >RESET command in this chapter.

ADDITIONAL DISCUSSION

Refer to the >RESET command in this chapter.

LANGUAGE

The >LANGUAGE command defines the native language whose collating sequence is to be used to sort keys of type CHARACTER.

SYNTAX

```
                              {langnum }
>L[ANGUAGE][IS]               {        }
                              {langname}
```

PARAMETERS

langnum	& This parameter specifies a language identification number. & The language specified & must be configured on the system.
langname	& The *langname* parameter specifies a language by name. & The language specified must be & configured on the system.

DISCUSSION

The >LANGUAGE command causes SORT-MERGE/XL to sort keys of type CHARACTER according to the collating sequence of the language specified by the *langnum* or *langname* parameter. The Native Language Support (NLS) intrinsics and files must first be installed on the system before the >LANGUAGE command can be used. Refer to Appendix C of this manual, the *Native Language Programmer's Guide*, (32650-90022) and the *Intrinsics Reference Manual* (32650-90028) for additional information on the >LANGUAGE command.

The >LANGUAGE command does not affect SORT-MERGE/XL messages, syntax, or prompts.

EXAMPLES

The following examples show using the >LANGUAGE command and its options.

```
>LANGUAGE IS SPANISH
```

Specifies Spanish as the native language. The Spanish language collating sequence is used.

```
>LANG 5
```

Specifies the native language identified as "5" in the system configuration.

```
>L FRENCH
```

Specifies French as the native language whose collating sequence will be used.

ADDITIONAL DISCUSSION

Refer to Appendix C, "Native Language Collating". Refer to the *Native Language Programmer's Guide* (32650-90022) and the *MPE XL Intrinsics Reference Manual* (32650-90028).

OUTPUT (MERGE/XL)

The >OUTPUT command is used to designate and create the output file, which is to receive the merged records. Refer to the SORT/XL >OUTPUT command for information on how to use the command within that subsystem.

SYNTAX

```
                {fname   }
>O[UTPUT]       {        } [,num records][, KEY]
                {$STDLIST}
```

PARAMETERS

fname	& The *fname* parameter represents the actual file designator.
$STDLIST	& Using this parameter specifies that the output is to & be sent to $STDLIST. The & output file is not saved when this parameter is used.

Note

In interactive mode the default is NOCCTL. In batch mode the default is CCTL (first byte stripped). To force NOCCTL in batch mode issue the following file equation:

```
:FILE OUTPUT; DEV=LP; NOCCTL
```

In this case, do not issue an output command to MERGE/XL.

num records	This parameter should be specified only if one or more input files are not on disc. (It is ignored if all input files are disc files.) It is a positive integer specifying the upper limit of the number of records to be merged and is used as the *filesize* parameter during the opening of the output file. If one or more of the files is not on disc and the parameter is not specified, a default value of 10,000 records is used.
KEY	Specifies that the output file is to consist of the key fields only, with the major key field on the left.

DISCUSSION

If more than one >OUTPUT command is issued, only the last one is effective.

If no >OUTPUT command is issued, MERGE/XL creates an output file with the name OUTPUT.

If a file already exists with the same name specified in the >OUTPUT command, during an interactive session, the following message is displayed:

```
PURGE OLD OUTPUT FILE filename ?
```

If the response is YES the old file is purged. If the response is NO or you press the Return key, the following message is displayed:

```
ENTER NEW NAME FOR OUTPUT FILE
```

If this prompt is displayed, enter a new name for the output file.

If this situation occurs in batch mode, the old file is not disturbed. Instead a new permanent file, OUTPUT*nn* (*n* is a non-negative integer) is created and the following message displayed:

```
OUTPUT FILE CLOSED WITH FILE NAME OUTPUT nn
```

If the above message is displayed the Job Control Word (JCW) is set to FATAL and the job aborts in an error state.

EXAMPLES

The following is an example of using the MERGE/XL >OUTPUT command in interactive mode:

```
:MERGE

HP32214A.01.00  MERGE/3000 THU, JUN  4, 1987, 10:00 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE, NEWHIRES
>OUTPUT COMPANY
>KEY 12, 11
>END
PURGE OLD OUTPUT FILE COMPANY.GROUP.ACCOUNT. ? YES


                    STATISTICS

NUMBER OF INPUT FILES =                          2
NUMBER OF RECORDS =                             20
SPACE AVAILABLE (IN WORDS) =                11,164
NUMBER OF COMPARES =                            18
CPU TIME (MINUTES) =                           .00
ELAPSED TIME (MINUTES) =                       .01
```

The two files EMPLOYEE and NEWHIRES are sorted files that are being merged into the new file COMPANY.

ADDITIONAL DISCUSSION

Refer to the >OUTPUT command for SORT/XL in this chapter.

OUTPUT (SORT/XL)

The >OUTPUT command designates and creates the output file which is to receive the sorted records. Refer to the MERGE/XL >OUTPUT command for information on how to use the command within that subsystem.

SYNTAX

```
                {*          }
   >O[UTPUT] { $STDLIST } [, NUM][, KEY]
                { filename }
```

PARAMETERS

*	Using this parameter specifies that the records are to be & sent to the file LIST, which defaults to $STDLIST.
$STDLIST	& Specifies that the sorted records are to be sent to $STDLIST. & The output file is not saved in this case.

Note

Use * cor CCTL first byte is stripped) and use $STDLIST for NOCCTL in interactive mode. In batch mode, both default to CCTL. The user may specify the following file equation to force NOCCTL:

```
FILE LIST; DEV=LP; NOCCTL
```

filename	This parameter identifies the actual file designator.
NUM	Specifies that the output records consist of the original logical record numbers only. The first record in the input file is considered number one.
KEY	Specifies that the output records consist of the key fields concatenated together from left to right with the major key field on the left. If neither NUM nor KEY is specified, the output records are identical to the input records. If NUM is specified, but KEY is not specified, the output records consist of a double integer whose value is the original logical (relative) record number. If KEY is specified and NUM is not specified, the output records consist of the key fields concatenated together from left to right. If both NUM and KEY are specified then each output record consists of the key fields concatenated together followed by the original logical record number.

DISCUSSION

If more than one `OUTPUT` command is issued, only the last one is effective.

If no output command is issued, SORT/XL creates an output file with the name `OUTPUT`.

If a file already exists with the same name as that specified in the `>OUTPUT` command, during an interactive session, the following message is displayed:

```
PURGE OLD OUTPUT FIle filename ?
```

If you respond `YES`, the old file is purged. If you respond `NO` or press the Return key, the following message is displayed:

```
ENTER NEW NAME FOR OUTPUT FILE
```

You should then enter a new name for the output file.

In batch mode, the old file is not disturbed. Instead a new permanent file, `OUTPUT`*nn* (*n* is a non-negative integer) is created and the following message is displayed:

```
OUTPUT FILE CLOSED WITH FILE NAME OUTPUT nn
```

The Job Control Word (JCW) is set to `FATAL`.

EXAMPLE

The following shows specifying the file company as the output file.

```
>OUTPUT COMPANY
```

ADDITIONAL DISCUSSION

Refer to the `>OUTPUT` command for MERGE/XL in this chapter.

RESET

The >RESET command is used to correct errors made in the specification of keys. When entered, it nullifies all existing >KEY commands.

SYNTAX

```
>RESET
```

DISCUSSION

If an error is made while entering specifications within the >KEY command enter >RESET. Then issue a new >KEY command with the correct key specifications.

EXAMPLE

The following example shows the key data item specifications for a sort on data located in character position (column) 12, and is 11 characters long:

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987, 10:20 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 11, 13
>RESET
>KEY 12, 11
>END
```

SHOW

The >SHOW command displays the collating sequence or the translation table.

SYNTAX

```
                {S[EQUENCE][,O[FFLINE]]}
                {T[ABLE][,O[FFLINE]]   }
   >SH[OW]      {                      }
                {NOS[EQUENCE]          }
                {NOT[ABLE]             }
```

PARAMETERS

S[EQUENCE]

The S[EQUENCE] parameter displays the collating sequence. This sequence is determined by the first 128 characters of the ASCII code, unless preceded by an >ALTSEQ command or a >DATA command with the EBCDIC sequence parameter. If the OFFLINE parameter is not issued, the sequence is displayed on the terminal. (If the OFFLINE parameter is issued, the sequence is printed on the line printer.) The display consists of the representation of each character in the relative order in which the collating sequence sorts (or merges) the records. Characters with the same ordinal values are adjoined by equal sign(s). Once specified in the >SHOW command, it is displayed after each subsequent >ALTSEQ command during a particular sort or merge operation until you specify NOSEQUENCE. OFFLINE activates the formal file designator DISPLOUT, with the line printer as the default device type (DEV=LP). Alternatively, you can store the contents of the sequence on a disc (or tape) file by appending DEV=DISC (or TAPE) to the file equation.

T[ABLE]

This parameter displays the translation table. After defining your special collating sequence, you may want to look at the table and the changes that occur in it. The table is helpful if you call SORT/XL (or MERGE/XL) from a program. (Refer to the *SORT-MERGE/XL Programmers Guide* (32650-90080) for additional information.) The translation table is organized according to the ASCII code decimal values of the characters. You should look at the position defined by the ASCII code decimal value to determine the ordinal value of a particular character. The table displays graphic characters each equated to its ordinal value, and the ordinal values of the characters that do not have graphic representation. Like the SEQUENCE option, the translation table is displayed after each >ALTSEQ command. The >SHOW TABLE command displays the table (in decimal) on the terminal.

NOS[EQUENCE]	Suppresses the display of the collating sequence in a particular SORT/XL (or MERGE/XL) operation. However, you can again get the display by specifying SEQUENCE.
NOT[ABLE]	Suppresses the display of the translation table until you enter a >SHOW TABLE command.

EXAMPLES

The following examples show how to display collating sequences and transaction tables.

Displaying the ASCII Collating Sequence

To display the standard ASCII collating sequence to your terminal enter >DATA IS ASCII, SEQUENCE IS ASCII followed by >SHOW SEQUENCE. This command generates the ASCII collating sequence based on the first 128 characters of the ASCII code. If you also enter OFFLINE after >SHOW SEQUENCE, the sequence is printed on the line printer.

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987, 10:25 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>SHOW SEQUENCE


     nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
     dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
      sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
       0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
       @   A   B   C   D   E   F   G   H   I   J   K   L   M   N   O
       P   Q   R   S   T   U   V   W   X   Y   Z   [   \   ]   ^   _
       ‘   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
       p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

Displaying the EBCDIC Collating Sequence

To display the EBCDIC collating sequence, enter the EBCDIC parameter of the >DATA command.

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987, 10:30 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS EBCDIC
>SHOW SEQUENCE


   nul soh stx etx  ht  del  vt  ff  cr  so  si dle dc1 dc2 dc3  bs
   can  em  fs  gs  rs   us  lf etb esc enq ack bel syn eot dc4 nak
   sub  sp   [   .   <    (   +   !   &   ]   $   *   )   ;   ^    -
     /   @   ,   %   _    >   ?   `   :   #   @   '   =   "   a   b
     c   d   e   f   g    h   i   j   k   l   m   n   o   p   q   r
     ~   s   t   u   v    w   x   y   z   {   A   B   C   D   E   F
     G   H   I   }   J    K   L   M   N   O   P   Q   R   \   S   T
     U   V   W   X   Y    Z   0   1   2   3   4   5   6   7   8   9
```

Displaying Recurring Collating Sequences

After you specify SHOW SEQUENCE in the >SHOW command, the collating sequence is displayed after each subsequent >ALTSEQ command until you specify the NOSEQUENCE parameter.

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987, 10:35 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>SHOW SEQUENCE


    nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
    dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
     sp   !   "   #   $   %   &   '   (   )   *   +   ,   -   .   /
      0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
      @   A   B   C   D   E   F   G   H   I   J   K   L   M   N   O
      P   Q   R   S   T   U   V   W   X   Y   Z   [   \   ]   ^   _
      `   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
      p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del



>'A'~~LTSEQ MERGE "A-C" WITH "D-L"   ''
```

```
nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
 sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
  0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
  @   A   D   B   E   C   F   G   H   I   J   K   L   M   N   O
  P   Q   R   S   T   U   V   W   X   Y   Z   [   \   ]   ^   _
  ‘   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
  p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

```
>ALTSEQ "A" = "B"
```

```
nul soh stx etx eot enq ack bel  bs  ht  lf  vt  ff  cr  so  si
dle dc1 dc2 dc3 dc4 nak syn etb can  em sub esc  fs  gs  rs  us
 sp   !   "   #   $   %   &   ’   (   )   *   +   ,   -   .   /
  0   1   2   3   4   5   6   7   8   9   :   ;   <   =   >   ?
  @   A=  D   B   E   C   F   G   H   I   J   K   L   M   N   O
  P   Q   R   S   T   U   V   W   X   Y   Z   [   \   ]   ^   _
  ‘   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o
  p   q   r   s   t   u   v   w   x   y   z   {   |   }   ~ del
```

```
>SHOW NOSEQUENCE
>ALTSEQ MERGE "a-c" WITH "A-C"
```

Using the >SHOW Command TABLE Parameter

Entering the `>SHOW TABLE` command, following the `>DATA` command, generates the translation table either to your terminal or to the printer if you designate `OFFLINE`. The standard ASCII translation table shows each character, in ascending order, and its ordinal (decimal) value.

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987, 10:40 AM
© HEWLETT-PACKARD CO. 1986

>DATA IS ASCII, SEQUENCE IS ASCII
>ALTSEQ "B" = "A"
>SHOW TABLE

:RUN SORT.PUB.SYS

HP32214C.02.05 SORT/3000 SUN,  JUL 19,  1987, 10:55 AM
© HEWLETT-PACKARD CO. 1986

>DATA A SEQ A
>A "B" = "A"
>SHOW TABLE
```

```
TABLE OF ORDINAL VALUE ASSIGNED TO EACH CHARACTER.
     !    0 !    1 !    2 !    3 !    4 !    5 !    6 !    7 !    8 !    9
 ----+------+------+------+------+------+------+------+------+------+------
   0 !    0 !    1 !    2 !    3 !    4 !    5 !    6 !    7 !    8 !    9 !
   1 !   10 !   11 !   12 !   13 !   14 !   15 !   16 !   17 !   18 !   19 !
   2 !   20 !   21 !   22 !   23 !   24 !   25 !   26 !   27 !   28 !   29 !
   3 !   30 !   31 !sp=32 !!= 33 !"= 34 !#= 35 !$= 36 !%= 36 !&= 38 !'= 39 !
   4 !(= 40 !)= 41 !*= 42 !+= 43 !,= 44 !-= 45 !.= 46 !/= 47 !0= 48 !1= 49 !
   5 !2= 50 !3= 51 !4= 52 !5= 53 !6= 54 !7= 55 !8= 56 !9= 57 !:= 58 !;= 59 !
   6 !<= 60 !== 61 !>= 62 !?= 63 !@= 64 !A= 65 !B= 65 !C= 67 !D= 68 !E= 69 !
   7 !F= 70 !G= 71 !H= 72 !I= 73 !J= 74 !K= 75 !L= 76 !M= 77 !N= 78 !O= 79 !
   8 !P= 80 !Q= 81 !R= 82 !S= 83 !T= 84 !U= 85 !V= 86 !W= 87 !X= 88 !Y= 89 !
   9 !Z= 90 ![= 91 !\= 92 !]= 93 !^= 94 !_= 95 !`= 96 !a= 97 !b= 98 !c= 99 !
  10 !d=100 !e=101 !f=102 !g=103 !h=104 !i=105 !j=106 !k=107 !l=108 !m=109 !
  11 !n=110 !o=111 !p=112 !q=113 !r=114 !s=115 !t=116 !u=117 !v=118 !w=119 !
  12 !x=120 !y=121 !z=122 !{=123 !|=124 !}=125 !~=126 ! =127 !  128 !  129 !
  13 !  130 !  131 !  132 !  133 !  134 !  135 !  136 !  137 !  138 !  139 !
  14 !  140 !  141 !  142 !  143 !  144 !  145 !  146 !  147 !  148 !  149 !
  15 !  150 !  151 !  152 !  153 !  154 !  155 !  156 !  157 !  158 !  159 !
  16 !  160 !  161 !  162 !  163 !  164 !  165 !  166 !  167 !  168 !  169 !
  17 !  170 !  171 !  172 !  173 !  174 !  175 !  176 !  177 !  178 !  179 !
  18 !  180 !  181 !  182 !  183 !  184 !  185 !  186 !  187 !  188 !  189 !
  19 !  190 !  191 !  192 !  193 !  194 !  195 !  196 !  197 !  198 !  199 !
  20 !  200 !  201 !  202 !  203 !  204 !  205 !  206 !  207 !  208 !  209 !
  21 !  210 !  211 !  212 !  213 !  214 !  215 !  216 !  217 !  218 !  219 !
  22 !  220 !  221 !  222 !  223 !  224 !  225 !  226 !  227 !  228 !  229 !
  23 !  230 !  231 !  232 !  233 !  234 !  235 !  236 !  237 !  238 !  239 !
  24 !  240 !  241 !  242 !  243 !  244 !  245 !  246 !  247 !  248 !  249 !
  25 !  250 !  251 !  252 !  253 !  254 !  255 !
WHEN PASSED TO SORTINIT, THE TABLE ABOVE IS PRECEDED BY TWO BYTES.
THESE FIRST TWO BYTES CONTAIN A FLAG BYTE OF %000 AND A LENGTH BYTE OF
%377 RESPECTIVELY.
```

Columns are labeled 0, 1, 2, through 9, and rows are labelled 0, 1, 2, through 25. The table is used by first reading down the leftmost column and then across from left to right. If you want to know the current ordinal value of B (whose ASCII code decimal value is 66), read down the table to locate the row labelled 6. Then read across until you reach the column with the heading 6. The value (65) contained in this position (6,6) identifies the location of the character B in the altered collating sequence.

Use the `OFFLINE` parameter to send the contents of the table to the line printer, disc, or tape. In this case, the table is created in three forms. During programmatic usage of SORT/XL or MERGE/XL, this information is edited and inserted into a program and then copied into the `>ALTSEQ` array passed to SORT/XL or MERGE/XL.

ADDITIONAL DISCUSSION

None.

VERIFY

The >VERIFY command displays information on the input and output files, key descriptions, and the various options in effect during a SORT/XL or MERGE/XL operation to the file LIST.

SYNTAX

```
>V[ERIFY]
```

DISCUSSION

The >VERIFY command displays information on the specifications for a particular sort or merge. The information provided includes the name of the input file, the name of the output file, specified key positions including their length and type, whether the sort or merge is to be done in ascending or descending order, and which key is the major key. It also provides the type of input data and type of sequence (ASCII or EBCDIC), if specified.

This command must be entered before the >END command which initializes the sort or merge operation specified.

EXAMPLE

The following example shows how to verify what has been designated as the conditions for a sort operation:

```
:SORT

HP32214A.01.00  SORT/3000 THU, JUN  4, 1987, 10:45 AM
© HEWLETT-PACKARD CO. 1986

>INPUT EMPLOYEE
>OUTPUT COMPANY
>KEY 1, 11; 12, 11
>DATA IS ASCII, SEQUENCE IS EBCDIC
>VERIFY

INPUT ENTITY = EMPLOYEE
OUTPUT ENTITY = COMPANY
KEY POSITION   LENGTH    TYPE    ASC/DESC
    1           11       BYTE       ASC    (MAJOR KEY)
   12           11       BYTE       ASC
INPUT DATA IS ASCII.
SEQUENCE IS IN EBCDIC.
```

ADDITIONAL DISCUSSION

None.

:(MPE Command)

The >: command is entered prior to issuing MPE commands within SORT/XL or MERGE/XL.

SYNTAX

```
>: [MPE command]
```

DISCUSSION

The >: command allows you to enter certain MPE commands without using the Break key. The colon indicates to SORT-MERGE/XL that it should pass the rest of the record to the MPE XL operating system. To continue an MPE command on the next record, the last nonblank character on the current record should be an ampersand (&). The command may be continued after the >: prompt.

Valid MPE commands are those capable of being executed programmatically. (Refer to the *MPE XL Commands Reference Manual* (32650-90003) for a list of valid entries.) Command Interpreter and file system error messages are printed if an error occurs. User Defined Commands, Command Files, and program files cannot be entered from the SORT-MERGE/XL >: command, but are valid during a Break.

EXAMPLE The following example shows using two MPE commands (:BUILD and :LIST) from within the SORT/XL subsystem.

```
:SORT

HP32214C.02.03  SORT/3000 THU, JUN  4, 1987, 11:00 AM
© HEWLETT-PACKARD CO. 1986
>:BUILD EMPLOYEE;REC=-132,10,F,ASCII; &
>:                   DISC=10000,32,32;CCTL
>:LISTF EMPLOYEE,2

THU, JUN  4, 1987,  11:06 AM

ACCOUNT=  SUBSYS       GROUP=  SORT

FILENAME  CODE  ---------LOGICAL RECORD--------  --SPACE--
              SIZE   TYP      EOF    LIMIT R/B  SECTORS #X MX

EMPLOYEE      133B   FAC        0    10000  10     6006 32 32

          >EXIT

          END OF PROGRAM
```

ADDITIONAL DISCUSSION Refer to the *MPE XL Commands Reference Manual* (32650-90003).

:EOD

The :EOD command is not truly a command. It terminates the list of input records to SORT/XL when * (for $STDIN) is the input file.

SYNTAX

```
>:EOD
```

The >:EOD (or :eod) command terminates the list of user input records when the terminal ($STDIN[X] is the input and output device. You input data at the system generated question mark prompt (?), and issue the :EOD command when you are done. The records are sorted and then displayed on the terminal screen.

EXAMPLE

The following example shows how to use your terminal to input and then receive a display of the sorted data:

```
>INPUT *
>OUTPUT $STDLIST
>KEY 1, 4
>END
?user input
?user input
?user input
?:EOD
sorted data
sorted data
sorted data
```

ADDITIONAL DISCUSSION

None.

A

Error Messages

This appendix contains error messages and recovery procedures for SORT-MERGE/XL. There is a listing of SORT/XL error messages, and a separate listing of MERGE/XL error messages. Each error message is numbered for easy referencing.

SORT/XL Error Messages

The SORT/XL program error messages are:

1	IF KEYCOMPARE IS SPECIFIED, KEYS AND NUM KEYS PARAMETERS MUST NOT BE.
2	IF KEYCOMPARE IS NOT SPECIFIED, KEYS AND NUMKEYS MUST BE SUPPLIED.
3	NO RECLEN PARAMETER IS SPECIFIED, OR REC LEN IS <= 0.
4	KEYCOMPARE MAY NOT BE SPECIFIED IF OUTPUTOPTION IS > 1.
5	FREAD ERROR OCCURRED ON SCRATCHFILE.
6	THIS IS AN ILLEGAL OUTPUT OPTION.
7	THE SCRATCH FILE CANNOT BE OPENED.
8	A FAILURE OCCURRED ON FGETINFO (INPUTFILE).
9	NUMKEYS IS ILLEGAL.
10	KEYFIELD IS NOT WITHIN THE SPECIFIED RECORD LENGTH.
11	THE ASCENDING/DESCENDING CODE IS ILLEGAL.
12	THE KEY CODE IS ILLEGAL.
13	THE STACK SPACE IS INSUFFICIENT.
14	THE INPUT RECORD DOES NOT INCLUDE ALL KEY FIELDS.
15	THE INPUT RECORD IS TOO LONG.
16	THERE ARE TOO MANY INPUT RECORDS.

17	FWRITE ERROR OCCURRED ON SCRATCHFILE.
18	FREAD ERROR OCCURRED ON INPUTFILE.
19	FWRITE ERROR OCCURRED ON OUTPUTFILE.
20	FCLOSE ERROR OCCURRED ON SCRATCHFILE.
21	$NULL IS NOT A VALID INPUT FILE.
22	FAILURE OCCURRED ON FGETINFO (OUTPUTFILE).
23	AN ERROR OCCURRED ATTEMPTING TO WRITE EOF ON SCRATCH FILE.
24	AN ERROR OCCURRED ATTEMPTING TO REWIND SCRATCH FILE.
25	ILLEGAL CHARACTERISTICS FOR FOPEN OF SCRATCH FILE.
26	THERE IS INSUFFICIENT STACK SPACE FOR THE SPECIFIED ALLOCATION.
27	A FAILURE OCCURRED ON FFILEINFO (INPUTFILE).
28	A FAILURE OCCURRED ON FFILEINFO (OUTPUTFILE).
29	SORT LANGUAGE IS NOT SUPPORTED.
30	NLINFO ERROR OCCURRED IN OBTAINING LENGTH OF COLLATING SEQUENCE TABLE.
31	NLINFO ERROR OCCURRED IN LOADING COLLATING SEQUENCE TABLE.
32	CHARSEQ PARAMETER IS INVALID.

33	THE TWO-BYTE COLLATING SEQUENCE TABLE IS NOT SPECIFIED.
34	A FAILURE OCCURRED IN FGETINFO (TWO-BYTE COLLATING SEQUENCE TABLE).
35	FREAD ERROR OCCURRED IN TWO-BYTE COLLATING SEQUENCE TABLE.
36	THE FILE IS NOT A VALID TWO-BYTE COLLATING SEQUENCE.
37	TWO-BYTE xxx UNDEF IN COLLATING SEQUENCE TABLE; LARGEST NO. ASSIGNED.
38	THE LENGTH OF TWO-BYTE MUST BE AN EVEN NUMBER OF BYTES.
39	THE FILE TYPE IS NOT A VALID TWO-BYTE COLLATING SEQUENCE TABLE.
40	PRINT INTRINSIC FAILED IN HPSORTTITLE.
41	PRINT INTRINSIC FAILED IN HPSORTSTAT.
190	THERE ARE TOO MANY INPUT FILES.
191	THERE ARE NO INPUT FILES IN THE SUPPLIED PARAMETERS.
193	IF YOU HAVE KEYS, YOU MUST HAVE NUMKEYS.
199	THE RECORD LENGTH EXCEEDS THE MAXIMUM ALLOWED.
200	THE MEMORY ALLOCATION IS NOT ENOUGH TO FIT 3 RECORDS IN A TREE.

201	OPEN OF STORAGE AREA FAILED.
202	SWITCH TO NM FROM CM, BUT NM CANNOT HANDLE THE SORT.
203	THE INPUT FILE NUMBER IS INVALID.
250	PROBE FAILED ON STATUS PARAMETER IN HPSORTINIT.
251	PROBE FAILED ON INPUTFILES PARAMETER IN HPSORTINIT.
252	PROBE FAILED ON OUTPUTFILES PARAMETER IN HPSORTINIT.
253	PROBE FAILED ON KEYSPARM PARAMETER IN HPSORTINIT.
254	PROBE FAILED ON ALTSEQ PARAMETER IN HPSORTINIT.
255	PROBE FAILED ON STATISTICS PARAMETER IN HPSORTINIT.
256	PROBE FAILED ON CHARSEQ PARAMETER IN HPSORTINIT.
257	PROBE FAILED ON STATUS PARAMETER IN HPSORTINPUT.
258	PROBE FAILED ON BUFF PARAMETER IN HPSORTINPUT.
259	PROBE FAILED ON LEN PARAMETER IN HPSORTINPUT.
260	PROBE FAILED ON STATUS PARAMETER IN HPSORTOUTPUT.
261	PROBE FAILED ON BUFF PARAMETER IN HPSORTOUTPUT.
262	PROBE FAILED ON LEN PARAMETER IN HPSORTOUTPUT.
263	PROBE FAILED ON STATUS PARAMETER IN HPSORTEND.

264	PROBE FAILED ON STATISTICS PARAMETER IN HPSORTEND.
265	PROBE FAILED ON STATUS PARAMETER IN HPSORTERRORMESS.
266	PROBE FAILED ON MESSAGE PARAMETER IN HPSORTERRORMESS.
267	PROBE FAILED ON LEN PARAMETER IN HPSORTERRORMESS.
268	PROBE FAILED ON STATUS PARAMETER IN HPSORTSTAT.
269	PROBE FAILED ON STATISTICS PARAMETER IN HPSORTSTAT.
270	PROBE FAILED ON STATUS PARAMETER IN HPSORTTITLE.
990	PREVIOUS NATIVE MODE ERROR OCCURRED.
992	SWITCH_TO_CM FAILED ON SORTTITLE.
993	SWITCH_TO_CM FAILED ON SORTERRORMESS.
994	SWITCH_TO_CM FAILED ON SORTEND2.
995	SWITCH_TO_CM FAILED ON SORTEND1.
996	SWITCH_TO_CM FAILED ON SORTOUTPUT.
997	SWITCH_TO_CM FAILED ON SORTINPUT.
998	SWITCH_TO_CM FAILED ON SORTGETHIDP.
999	SWITCH_TO_CM FAILED ON SORTINIT.
1000	HPSORTERRORMESS FAILED ON THE CALL TO HPERRMSG INTRINSIC.

MERGE/XL Error Messages

The MERGE/XL program error messages are:

3	NO INPUTFILE PARAMETER IS SPECIFIED.
4	NEITHER OUTPUTFILE NOR POSTPROCESSOR PARAMETER IS SPECIFIED.
5	IF KEYCOMPARE IS SPECIFIED, KEYS AND NUMKEYS MUST NOT BE.
6	IF KEYCOMPARE IS NOT SPECIFIED, KEYS AND NUMKEYS MUST BE.
7	NUMKEYS IS ILLEGAL.
8	KEYFIELD IS NOT WITHIN THE RECORD LENGTH OF EACH FILE.
9	THE ASCENDING/DESCENDING CODE IS ILLEGAL.
10	THE KEY CODE IS ILLEGAL.
11	FGETINFO ON INPUTFILE FAILED.
12	FREAD ERROR OCCURRED ON INPUT FILE.
13	FWRITE ERROR OCCURRED ON OUTPUT FILE.
14	THE INPUT RECORD DOES NOT INCLUDE ALL OF THE KEY FIELDS.
15	IF KEYCOMPARE PARAMETER IS SPECIFIED, KEYSONLY PARAMETER MAY NOT BE.
16	THE STACK SPACE IS INSUFFICIENT.

17	THE STACK SPACE IS INSUFFICIENT FOR THE SPECIFIED ALLOCATION.
18	FAILURE OCCURRED ON FGETINFO (OUTPUTFILE).
19	$NULL IS NOT A VALID INPUT FILE.
21	SORT LANGUAGE IS NOT SUPPORTED.
22	NLINFO ERROR OCCURS IN OBTAINING LENGTH OF COLLATING SEQUENCE TABLE.
23	NLINFO ERROR OCCURS IN LOADING COLLATING SEQUENCE.
24	CHARSEQ PARAMETER IS INVALID.
25	TWO-BYTE COLLATING SEQUENCE TABLE IS NOT SPECIFIED.
26	FAILURE OCCURRED ON FGETINFO (TWO-BYTE COLLATING SEQUENCE TABLE).
27	FREAD ERROR OCCURRED ON TWO-BYTE COLLATING SEQUENCE TABLE.
28	THE FILE IS NOT A VALID TWO-BYTE COLLATING SEQUENCE TABLE.
29	TWO-BYTE xxxx IS UNDEFINED IN COLLATING SEQUENCE TABLE; LARGEST NO. ASSIGNED.
30	THE LENGTH OF THE TWO-BYTE KEY MUST BE AN EVEN NUMBER.
31	THE FILE TYPE IS NOT A VALID TWO-BYTE COLLATING SEQUENCE TABLE.
40	PRINT INTRINSIC FAILED IN HPSORTTITLE.

41	PRINT INTRINSIC FAILED IN HPSORTSTAT.
109	NUMKEYS IS ILLEGAL.
250	PROBE FAILED ON STATUS PARAMETER IN HPMERGEINIT.
251	PROBE FAILED ON INPUTFILES PARAMETER IN HPMERGEINIT.
252	PROBE FAILED ON OUTPUTFILES PARAMETER IN HPMERGEINIT.
253	PROBE FAILED ON KEYS PARAMETER IN HPMERGEINIT.
254	PROBE FAILED ON ALTSEQ PARAMETER IN HPMERGEINIT.
255	PROBE FAILED ON STATISTICS PARAMETER IN HPMERGEINIT.
256	PROBE FAILED ON CHARSEQ PARAMETER IN HPMERGEINIT.
257	PROBE FAILED ON STATUS PARAMETER IN HPMERGEINPUT.
258	PROBE FAILED ON BUFF PARAMETER IN HPMERGEOUTPUT.
259	PROBE FAILED ON LEN PARAMETER IN HPMERGEINPUT.
260	PROBE FAILED ON STATUS PARAMETER IN HPMERGEOUTPUT.
261	PROBE FAILED ON BUFF PARAMETER IN HPMERGEOUTPUT.
262	PROBE FAILED ON LEN PARAMETER IN HPMERGEOUTPUT.
263	PROBE FAILED ON STATUS PARAMETER IN HPMERGEEND.
264	PROBE FAILED ON STATISTICS PARAMETER IN HPMERGEEND.
265	PROBE FAILED ON STATUS PARAMETER IN HPMERGEERRORMESS.

266	PROBE FAILED ON MESSAGE PARAMETER IN HPMERGEERRORMESS.
267	PROBE FAILED ON LEN PARAMETER IN HPMERGEERRORMESS.
268	PROBE FAILED ON STATUS PARAMETER IN HPMERGESTAT.
269	PROBE FAILED ON STATISTICS PARAMETER IN HPMERGESTAT.
270	PROBE FAILED ON STATUS PARAMETER IN HPMERGETITLE.
993	SWITCH_TO_CM FAILED ON MERGETITLE.
994	SWITCH_TO-CM FAILED ON MERGEERRORMESS.
995	SWITCH_TO_CM FAILED ON MERGEEND2.
996	SWITCH_TO_CM FAILED ON MERGEEND1.
997	SWITCH_TO_CM FAILED ON MERGEOUTPUT.
998	SWITCH_TO-CM FAILED ON MERGEGETHIDP.
999	SWITCH_TO_CM FAILED ON MERGEINIT.
1000	HPMERGEERRORMESS FAILED ON THE CALL TO HPERRMSG INTRINSIC.

Recovery Procedures

Errors that occur during a batch mode job are not recoverable. An error message is generated and the program terminates abnormally.

During an interactive session syntax errors are recoverable. An error message is displayed and you are requested to enter the command correctly.

B

ASCII/EBCDIC Character Sets

The ASCII/EBCDIC table shown below is arranged according to character code values. Each character is represented by its decimal, octal, and hexadecimal equivalents.

To determine the ASCII code value of the character $, scan down the ASCII graphic column until you locate $. Then read to its right to find the values 36 (decimal), 044 (octal), or 24 (hexadecimal). This is the code value used by devices such as terminals, printers, or the CPU to represent the character $. To find the character with the EBCDIC code value 5B (hexadecimal) locate 5B in the hexadecimal character value column and move left to the EBCDIC graphic column containing $.

Abbreviations appearing in the table (for example NUL, SOH, STX, and ETX) are explained following the table.

Table B-1. ASCII/EBCDIC Character Sets

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal	Octal	Hexadecimal
	NUL			
NUL	SOH	0	000	00
SOH	STX	1	001	01
STX	ETX	2	002	02
ETX		3	003	03
	PF			
EOT	HT	4	004	04
ENQ	LC	5	005	05
ACK	DEL	6	006	06
BEL		7	007	07
BS		8	010	08
HT	SMM	9	011	09
LF	VT	10	012	1A
VT		11	013	0B
	FF			
FF	CR	12	014	0C
CR	SO	13	015	0D
SO	SI	14	014	0E
SI		15	015	0F

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal	Octal	Hexadecimal
	DLE			
DLE	DC1	16	020	10
DC1	DC2	17	021	11
DC2	TM	18	022	12
DC3		19	023	13
	RES			
DC4	NL	20	024	14
NAK	BS	21	025	15
SYN	IL	22	026	16
ETB		23	027	17
	CAN			
CAN	EM	24	030	18
EM	CC	25	031	19
SUB	CU1	26	032	1A
ESC		27	033	1B
	IFS			
FS	IGS	28	034	1C
GS	IRS	29	035	1D
RS	IUS	30	036	1E
US		31	037	1F
	DS			
SP	SOS	32	040	20
!	FS	33	041	21
"		34	042	22
#		35	043	23
	BYP			
$	LF	36	044	24
%	ETB	37	045	25
&	ESC	38	046	26
'		39	047	27
(		40	050	28
)	SM	41	051	29
*	CU2	42	052	2A
+		43	053	2B
	ENQ			
,	ACK	44	054	2C
-	BEL	45	055	2D
.		46	056	2E
/		47	057	2F

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal	Octal	Hexadecimal
0	SYN			
1		48	060	30
2		49	061	31
3	PN	50	062	32
	RS	51	063	33
4	UC			
5	EOT	52	064	34
6		53	065	35
7		54	066	36
		55	067	37
8				
9	CU3	56	070	38
:		57	071	39
;	DC4	58	072	3A
	NAK	59	073	3B
<				
=	SUB	60	074	3C
>		61	075	3D
?		62	076	3E
		63	077	3F
@	SP			
A		64	100	40
B		65	101	41
C		66	102	42
		67	103	43
D				
E		68	104	44
F		69	105	45
G		70	106	46
		71	107	47
H				
I		72	110	48
J		73	111	49
K	.	74	112	4A
		75	113	4B
L	<			
M	(	76	114	4C
N	+	77	115	4D
O	|	78	116	4E
		79	117	4F

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal	Octal	Hexadecimal
	&			
P		80	120	50
Q		81	121	51
R		82	122	52
S		83	123	53
T		84	124	54
U		85	125	55
V		86	126	56
W		87	127	57
X		88	130	58
Y	!	89	131	59
Z	$	90	132	5A
[		91	133	5B
	*			
	)	92	134	5C
]	;	93	135	5D
^^		94	136	5E
_		95	137	5F
‘	_			
a	/	96	140	60
b		97	141	61
c		98	142	62
d		99	143	63
e		100	144	64
f		101	145	65
g		102	146	66
h		103	147	67
i		104	150	68
j		105	151	69
k	,	106	152	6A
l	%	107	153	6B
m	-	108	154	6C
n	>	109	155	6D
o	?	110	156	6E
		111	157	6F

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal	Octal	Hexadecimal
	‘			
p	:	112	160	70
q	#	113	161	71
r		114	162	72
s	@	115	163	73
	’			
t	=	116	164	74
u	"	117	165	75
v		118	166	76
w		119	167	77
x		120	170	78
y		121	171	79
z		122	172	7A
{		123	173	7B
|		124	174	7C
}		125	175	7D
~		126	176	7E
DEL		127	177	7F
	a			
	b	128	200	80
	c	129	201	81
		130	202	82
	d	131	203	83
	e			
	f	132	204	84
	g	133	205	85
		134	206	86
	h	135	207	87
	i			
		136	210	88
		137	211	89
		138	212	8A
		139	213	8B
		140	214	8C
		141	215	8D
		142	216	8E
		143	217	8F

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal Octal Hexadecimal		
	j			
	k	144	220	90
	l	145	221	91
		146	222	92
	m	147	223	93
	n			
	o	148	224	94
	p	149	225	95
		150	226	96
	q	151	227	97
	r			
		152	230	98
		153	231	99
		154	232	9A
		155	233	9B
		156	234	9C
		157	235	9D
		158	236	9E
		159	237	9F
	~			
	s	160	240	A0
	t	161	241	A1
		162	242	A2
	u	163	243	A3
	v			
	w	164	244	A4
	x	165	245	A5
		166	246	A6
	y	167	247	A7
	z			
		168	250	A8
		169	251	A9
		170	252	AA
		171	253	AB
		172	254	AC
		173	255	AD
		174	256	AE
		175	257	AF

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal Octal Hexadecimal		
	j			
	k	144	220	90
	l	145	221	91
		146	222	92
	m	147	223	93
	n			
	o	148	224	94
	p	149	225	95
		150	226	96
	q	151	227	97
	r			
		152	230	98
		153	231	99
		154	232	9A
		155	233	9B
		156	234	9C
		157	235	9D
		158	236	9E
		159	237	9F
	~			
	s	160	240	A0
	t	161	241	A1
		162	242	A2
	u	163	243	A3
	v			
	w	164	244	A4
	x	165	245	A5
		166	246	A6
	y	167	247	A7
	z			
		168	250	A8
		169	251	A9
		170	252	AA
		171	253	AB
		172	254	AC
		173	255	AD
		174	256	AE
		175	257	AF

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal Octal Hexadecimal		
		176	260	B0
		177	261	B1
		178	262	B2
		179	263	B3
		180	264	B4
		181	265	B5
		182	266	B6
		183	267	B7
		184	270	B8
		185	271	B9
		186	272	BA
		187	273	BB
		188	274	BC
		189	275	BD
		190	276	BE
		191	277	BF
	A			
	B	192	300	C0
	C	193	301	C1
		194	302	C2
	D	195	303	C3
	E			
	F	196	304	C4
	G	197	305	C5
		198	306	C6
	H	199	307	C7
	I			
		200	310	C8
		201	311	C9
		202	312	CA
		203	313	CB
		204	314	CC
		205	315	CD
		206	316	CE
		207	317	CF

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal Octal Hexadecimal		
	J			
	K	208	320	D0
	L	209	321	D1
		210	322	D2
	M	211	323	D3
	N			
	O	212	324	D4
	P	213	325	D5
		214	326	D6
	Q	215	327	D7
	R			
		216	330	D8
		217	331	D9
		218	332	DA
		219	333	DB
		220	334	DC
		221	335	DD
		222	336	DE
		223	337	DF
	\			
	S	224	340	E0
	T	225	341	E1
		226	342	E2
	U	227	343	E3
	V			
	W	228	344	E4
	X	229	345	E5
		230	346	E6
	Y	231	347	E7
	Z			
		232	350	E8
		233	351	E9
		234	352	EA
		235	353	EB
		236	354	EC
		237	355	ED
		238	356	EE
		239	357	EF

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal	Octal	Hexadecimal
	J			
	K	208	320	D0
	L	209	321	D1
		210	322	D2
	M	211	323	D3
	N			
	O	212	324	D4
	P	213	325	D5
		214	326	D6
	Q	215	327	D7
	R			
		216	330	D8
		217	331	D9
		218	332	DA
		219	333	DB
		220	334	DC
		221	335	DD
		222	336	DE
		223	337	DF
	\			
	S	224	340	E0
	T	225	341	E1
		226	342	E2
	U	227	343	E3
	V			
	W	228	344	E4
	X	229	345	E5
		230	346	E6
	Y	231	347	E7
	Z			
		232	350	E8
		233	351	E9
		234	352	EA
		235	353	EB
		236	354	EC
		237	355	ED
		238	356	EE
		239	357	EF

Table B-1. ASCII/EBCDIC Character Sets (Cont.)

ASCII Control/ Graphic	EBCDIC Control/ Graphic	Character Code Values Decimal Octal Hexadecimal
	0	
	1	240 360 F0
	2	241 361 F1
	3	242 362 F2
		243 363 F3
	4	
	5	244 364 F4
	6	245 365 F5
	7	246 366 F6
		247 367 F7
	8	
	9	248 370 F8
		249 371 F9
		250 372 FA
		251 373 FB
		252 374 FC
		253 375 FD
		254 376 FE
		255 377 FF

NUL	=	Null
SOH	=	Start of Heading
STX	=	Start of Text
ETX	=	End of Text
EOT	=	End of Transmission
ENQ	=	Enquiry
ACK	=	acknowledge
BEL	=	Bell
BS	=	Backspace
HT	=	Horizontal Tabulation
LF	=	Line Feed
VT	=	Vertical Tabulation
FF	=	Form Feed
CR	=	Carriage Return
SO	=	Shift Out
SI	=	Shift In
DLE	=	Data Link Escape
DC1	=	Device Control 1 (X-ON)
DC2	=	Device Control 2
DC3	=	Device Control 3 (X-OFF)
DC4	=	Device Control 4
NAK	=	Negative Acknowledge
SYN	=	Synchronous Idle
ETB	=	End of Transmission Block
CAN	=	Cancel
EM	=	End of Medium
SUB	=	Substitute
ESC	=	Escape
FS	=	File Separator
GS	=	Group Separator
RS	=	Record Separator
US	=	Unit Separator
SP	=	Space (Blank)
DEL	=	Delete

C

Native Language Collating

Native Language Support (NLS) for the 900 Series HP 3000 provides collating for a variety of native languages. A number of collating algorithms, from simple to very complex, have been employed in defining the collating sequences for these languages, depending on the requirements of the native users of the languages.

Native language collating sequences are accessed in SORT-MERGE/XL by using the key type `CHARACTER` and the `>LANGUAGE` command to define which native language collating sequence is to be used. In addition to actual native languages, an artificial language, NATIVE-3000, has been defined to handle all language aspects in a traditional computer manner. Thus, for example, one collating sequence for NATIVE-3000 treats keys of *type* `CHARACTER` the same as keys of *type* `BYTE` and collates them according to the value of the ASCII code for each character.

For a list of languages supported on your 900 Series HP 3000 run `NLUTIL.PUB.SYS`. A list of language names and language IDs is displayed. The exact list depends on the configuration chosen by your System Manager. Configured languages may include, but are not limited to, those shown in Figure C-1 below. The program `NLUTIL.PUB.SYS` also offers to print the definition, including the collating sequence, of each language supported. Refer to the *Native Language Programmer's Guide* (32650-90022) for additional information.

Lang ID	Lang Name	Char Set ID	Char Set Name
0	NATIVE-3000	0	USASCII
1	American	1	ROMAN8
2	Canadian-French	1	ROMAN8
3	Danish	1	ROMAN8
4	Dutch	1	ROMAN8
5	English	1	ROMAN8
6	Finnish	1	ROMAN8
7	French	1	ROMAN8
8	German	1	ROMAN8
9	Italian	1	ROMAN8
10	Norwegian	1	ROMAN8
11	Portuguese	1	ROMAN8
12	Spanish	1	ROMAN8
13	Swedish	l	ROMAN8
41	Katakana	2	KANA8

Figure C-1. MPE XL - Supported Native Languages

Glossary

Glossary

Access
The process of obtaining data from files or acquiring the use of a device. Access implies an input/output (I/O) operation and is used as a synonym for I/O.

Actual File Designator
The file name provided by the user. The system then uses the file name in place of the formal file designator to accomplish some task. The actual file designator is the file name listed in the directory. Refer to formal file designator.

Algorithm
A step-by-step procedure for solving a problem in a finite amount of time.

American National Standards Institute (ANSI)
A non-governmental agency that establishes standards, including those for the data processing industry.

American Standard Code for Information Interchange (ASCII/USASCII)
The standard method of representing character data (seven data bits plus one that is sometime used for parity). This method was established by the American National Standards Institute (ANSI) to achieve compatibility between data devices when they are interchanging information.

Arithmetic Logic Unit
The part of the system that performs arithmetic and logic operations as part of the Central Processing Unit (CPU). The CPU may contain one or more Arithmetic Logic Units.

Ascending Record
A record that is collated in an ascending order (A to Z or 0 to 9).

ASCII Refer to American Standard Code for
Information Interchange.

Batch
A data processing method. Batch processing allows users to submit, for processing as a single unit, commands that request various operations such as program compilation and execution, file manipulation, or utility functions. Such a unit is called a job.

Once a job has been submitted no further interaction between the user and the job is necessary. The opposite of Interactive.

Cathode Ray Tube (CRT)
A video display screen used as a means of communicating with a computer is called a terminal. A CRT produces soft copy.

Central Processing Unit (CPU)
A part of a system. The CPU interprets and executes instructions and contains all or part of internal storage. The central processor contains an Execution Unit and a Control Unit.

Character
A letter, number, or symbol represented by one byte of data.

Chevron
SORT-MERGE/XL uses a chevron character (>) as its subsystem prompt. All SORT-MERGE/XL commands are entered at the chevron (>) prompt.

Collating Sequence
The sequence by which characters are listed and records are sorted or merged. In SORT-MERGE/XL it is possible to collate characters or records according to ASCII, EBCDIC, Native Language, or user-defined sequences.

Column
A method of measuring the length of a record or line. A standard line consists of 80 columns.

Command
A system-defined word that directs the operating system, subsystem, or a utility program to perform a specific operation.

Compatibility Mode (CM)
Compatibility Mode provides object code compatibility between Mode allows Hewlett-Packard customers to move applications and data from their current systems to the 900 Series HP 3000 without changes or recompilation.

Continuation Character
SORT-MERGE/XL uses the ampersand ( ) as its recognizable continuation character. By entering an as the last character on a line, the record is continued onto a second, third, or any number of subsequent lines.

Control Unit
A part of the Central Processing Unit (CPU) that regulates the Execution Unit (EU) and oversees the instruction cycle.

CPU Time
The amount of time, in seconds, that a user, group, or account has used the CPU (Central Processing Unit).

Cursor
A flashing rectangle or blinking underline character on a display screen that marks the position where text or data can be entered, changed, or deleted.

Decimal Value
A decimal representation of an ASCII character. For example, the character "A" has the ASCII binary code value 01000001 and the decimal code value of 65.

Delimiter
A character that marks the end of a string of characters such as those comprising a command. Common delimiters are a comma (`,`), semicolon (`;`), equal sign (`=`), or a Return.

Descending Record
Characters or records are collated in a descending order when the sequence is Z to A or 9 to 0.

Display File
When the `>SHOW` command is used to display either the translation table or the collating sequence, the information is sent to a system-created file known as the display file.

EDIT/V
An HP 3000 text editor, supplied with MPE XL, used to create and manipulate ASCII files.

Error Messages
Messages describing errors occurring during either an interactive session or a batch job. The messages are reported to the standard list device, which is usually a terminal (for a session) or a line printer (for a job).

Execute
To carry out an instruction or perform a routine.

Execution Unit (EU)
The part of the Central Processing Unit (CPU) containing the Arithmetic Logic Unit (ALU) and the registers. Data is held in registers and manipulated in the ALU.

Extended Binary Coded Decimal Interchange Code (EBCDIC)
An 8-bit code that is an extension of Binary-Coded Decimal (BCD) notation. EBCDIC can represent up to 256 different characters.

File Equation
The result of using the MPE XL :FILE command to equate a file name to a device or another file, or to override the file's characteristics. Generally used to direct the input to or output from a program, job, or session to a particular device by referencing the device class, such as TAPE or LP.

Formal File Designator
A name used programmatically or in a file equation to reference a file. The formal file designator is not the file name found in the directory. Refer to actual file designator.

Hard Copy
The output from a printer or plotter, usually onto paper. The opposite of soft copy.

Hexadecimal
A method of representing a single alphanumeric character with a

16 numbering system, in which the first 10 digits are 0 through 9, and the last six are A through F. When a number is written in base 16, it is preceded by a dollar sign "$" (for example, $F3 is the hexadecimal representation for 243).

Implied :RUN
The ability to run a program without explicitly using the MPE XL :RUN command. For MPE XL it is not necessary to specify

It is only necessary to enter :SORT.

Input File
The input file is designated by using the >INPUT command as the file containing the information you want to sort or merge.

Input/Output (I/O)
The process of, or equipment used in, transmitting information to or from the computer.

Interactive
Interactive processing allows you to enter commands and data at the terminal and receive an immediate response from the system. This is called a session. Sessions are useful for data entry and retrieval, text editing, and program development where direct dialog with the computer is preferred. The opposite of batch.

Intrinsic
A system routine accessible by user programs providing interface to common tasks such as file access, message formatting, or data conversion.

I/O
Refer to Input/Output.

Job
A single file, submitted by a user, containing operating system and utility commands and references to the files to be manipulated. Once submitted, a job executes independently of the initiating user or session. Jobs are used to compile source programs, modify files, or perform other functions not requiring user interaction. Submitting a job is also called streaming or batch processing. The opposite of session.

Key Data Item
A key is that section of the record that SORT-MERGE/XL uses as a reference to arrange the desired data in a defined order.

List File
The list file issues error messages and prompts during interactive sessions while using SORT-MERGE/XL.

Localizable
That quality of software or documentation that facilitates changes to the punctuation characters, key words, and command names to fit a particular native language so that applications can be used in different countries. The user interface is in the user's native language depending upon country.

Major Key
In SORT-MERGE/XL, the first key data item specified with the `>KEY` command is considered the major key and is the first key used for sorting or merging operations.

MERGE/XL
A subsystem of the MPE XL operating system for the 900 Series HP 3000 that allows you to merge two or more previously sorted files into a new file containing the merged data.

Multi-Programming Executive With Extended Large Addressing (MPE XL)
MPE XL is the operating system for the 900 Series HP 3000 computers. It consists of programs that handle exchanges between Hewlett-Packard terminals, printers, storage devices, memory, and executing programs. A disc-based operating system, MPE XL manages all system resources and coordinates the execution of all programs running on the system.

Native Language Support(NLS)
MPE XL utilities and intrinsics that facilitate the development of applications for users in different countries. NLS includes such features as currency symbol handling and character translation.

Native Mode
The native run-time environment of MPE XL. In Native Mode source code has been compiled into the native instruction set of the 900 Series HP 3000.

Octal The base eight numbering system, in which digits
0-7 are used. One octal digit can be represented by three binary digits. Octal numbers are preceded by a percent sign "%" (for example, %101 which is the octal representation for the character "A").

Operating System
The software that allows the computer to operate. It consists of programs such as basic file and I/O manipulators. All subsystems run upon the operating system.

Output File
The results of a sort or merge operation are sent to the output file. This file is specified by using the `>OUTPUT` command.

Privileged Mode (PM)
A mode of running in MPE XL that frees the user from most system constraints.

Prompt File
The prompt file asks you for input when the text file is the session terminal but the list file is not.

Range
All of the values that a function or word may have. For example, the range "A-Z" would include each of the characters in the range ABC ... Z.

Record
A collection of fields or related data treated as a unit, residing in a file. A contiguous group of bytes whose structure is known by the file system. A record can consist of more than one line of data in a file continued with the ampersand ( ) character at the end of each line.

Scratch File
SORT/XL uses the scratch file as a work area. MERGE/XL does not use the Scratch File.

Session
A mode in which the HP 3000 is used interactively by entering commands and data through a terminal's keyboard and receiving immediate responses to the input from the system. A session is initiated with the `:HELLO` command. A session is ended with the `:BYE` command, or a second `:HELLO` command that logs the user off the first session and onto another session. The opposite of job.

Soft Copy
The display on a video terminal. The opposite of hard copy.

SORT/XL
A subsystem of the MPE XL operating system for the 900 Series HP 3000 that allows you to sort information in files, based upon single or multiple key data items either alphabetically or numerically.

$STDIN
A system-defined file name that refers to the standard input device used to initiate a session or job; usually a terminal keyboard or tape drive.

$STDINX
A system-defined file name that refers to the standard input device used to initiate a session or job. Unlike `$STDIN`, `$STDINX` treats the colon (:) prompt appearing in the first column as part of the data file, rather than an end-of-file indicator.

$STDLIST
A file name indicating the standard job or session listing file corresponding to the particular input device being used. The listing device is usually a printer for batch jobs and a terminal for sessions.

Subsystem
SORT/XL and MERGE/XL are subsystems of MPE XL. A subsystem is a software program that performs a specific function such as compile programs, copy files, sort/merge files, or edit text. Subsystems are accessed by entering a single command at the MPE XL colon prompt. Then a different prompt is displayed (a chevron > for SORT-MERGE/XL) and a set of commands, specific to the subsystem, becomes available to the user. The user must explicitly exit the subsystem, usually by entering `E` or `EXIT`. To exit the SORT-MERGE/XL subsystem you enter either `EXIT` or `EX`.

Terminal
A hardware device connected to a computer. A terminal is used for entering and receiving data. It consists of a keyboard and a display screen.

Text and Document Processor/V (TDP/V)
An HP 3000 line editor (with a screen editor option). TDP/V is used to create, manipulate, and format ASCII text files.

Text File
Both SORT/XL and MERGE/XL read commands directly from the text file.

Translation Table
The default translation table for SORT-MERGE/XL follows the standard 128-character ASCII sequence, where each character is represented internally by a numeric value of from 0 to 127.

USASCII
Refer to American Standard Code for Information Interchange.

Utility Program
An operating system program that performs specific functions such as file copying, sorting and merging, memory dump analysis, or monitoring available disc space. SORT-MERGE/XL is a utility program.

Index