IBM System/36

Sort Guide

IBM System/36

Sort Guide

File Number
S36-33

Order Number
SC21-7903-2

Third Edition (April 1985)

This major revision obsoletes SC21-7903-1. Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Changes are periodically made to the information herein; any such changes will be reported in subsequent revisions or Technical Newsletters.

This edition applies to Release 3, Modification Level 0, of IBM System Support Program Product (SSP) (Program no. 5727-SS1), and to all subsequent releases and modifications until otherwise indicated in new editions or Technical Newsletters. Also, this publication contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the address given below. Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

This publication could contain technical inaccuracies or typographical errors. A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Canada Ltd., Information Development, Department 849, 1150 Eglinton Avenue East, Don Mills, Ontario, Canada M3C 1H7. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1983, 1984, 1985

Contents

About This Manual

Who should use this manual . . .

The purpose of the *Sort Guide* is to familiarize readers with the types of sorted output offered by the system and to serve as a reference for the programmer who will design and code sort jobs or the system operator who will run them.

Using this manual, the reader can:

- Identify the type of sorted output needed
- Define the sort specifications
- Enter the sort specifications into the system
- Run a sort job using the SORT procedure or OCL statements
- Supply OCL statements and sort specifications when using procedures to run a sort job
- Efficiently use sort run time.

How this manual is arranged . . .

The information in this manual is organized into two major sections:

- A *guide section*. This section, which includes chapters one through six, contains information about how to design, code, and run a sort job, and uses examples of some column entries for the sort specifications.
- A *reference section*. This section, which includes chapters seven through nine, contains the sort specifications column entries.

The remainder of this *Sort Guide* includes several appendixes, a glossary, and an index.

What you should know . . .

- *Learning About Your Computer,* SC21-9018, is an introduction to data processing on the system.

If you need more information . . .

- *Operating Your Computer,* SC21-9026, explains how to operate the devices associated with System/36.
- *Source Entry Utility Guide,* SC21-7901, explains how to enter source programs on System/36.
- *Ideographic Sort Guide,* SC09-1054, explains how to design, code, and run sort programs and procedures that use ideographic data characters.
- *System Reference,* SC21-7938, is a reference manual for all System/36 statements, procedures, and commands.
- *System Messages,* SC21-7938, contains all sort messages that are printed or displayed by System/36 during a sort job run.
- *Before Calling for Service,* SC21-7919, helps you determine whether a problem is in your own program or in the computer.

You should also have . . .

- *Sort Specifications,* GX21-9089, is a form used to code sort jobs.
- *Translation Table and Alternate Collating Sequence Coding Form,* GX21-9096, is used to change the standard collating sequence.

Summary of changes

The following changes have been made for Release 3 Modification 0:

- Chapters 4 and 5 contain references to the use of remote files. See the Distributed Data Management manual, SC21-8011, for more information.
- Chapter 4 contains more information on sorting multiple master files.
- Various technical and editorial changes have been made to improve the quality and usability of this manual.

Chapter 1. An Introduction to Sort

Sort is one of the system utilities in the IBM System/36 System Support Program (SSP) that allows you to do the following:

- **Arrange records** in a file into ascending order, descending order, or a combination of the two
- **Drop records** from a file
- **Reformat records** in a file
- **Merge records** (up to eight input files) into one file.

The input data to a sort program is never changed except when you specify that the output file overlay (that is, be written over) the input file. When you specify that the sort program write the output file over the input file, all the input data is lost. When you do not overlay the files, the sort program simply makes a copy of the input data and uses the copied records when processing sort jobs.

What You Can Do Using Sort

The sort program provides an easy way to arrange records in a data file into a specific sequence. You identify to the system (using sort specifications) the type of sorted output you want and the records and fields to be used by the sort program to arrange the records and fields in a certain order.

You must specify which input fields will control the sorting. You can use fields within the input records to control the order in which the records will be arranged. You can define one or more of these fields, called **control fields**, in a sort job. For example, you can specify that one control field arrange some records in a certain order while another control field arranges other records in the opposite order.

To specify how you want a file sorted, you use sort specifications. The sort specifications are described in detail in *Chapter 2.*

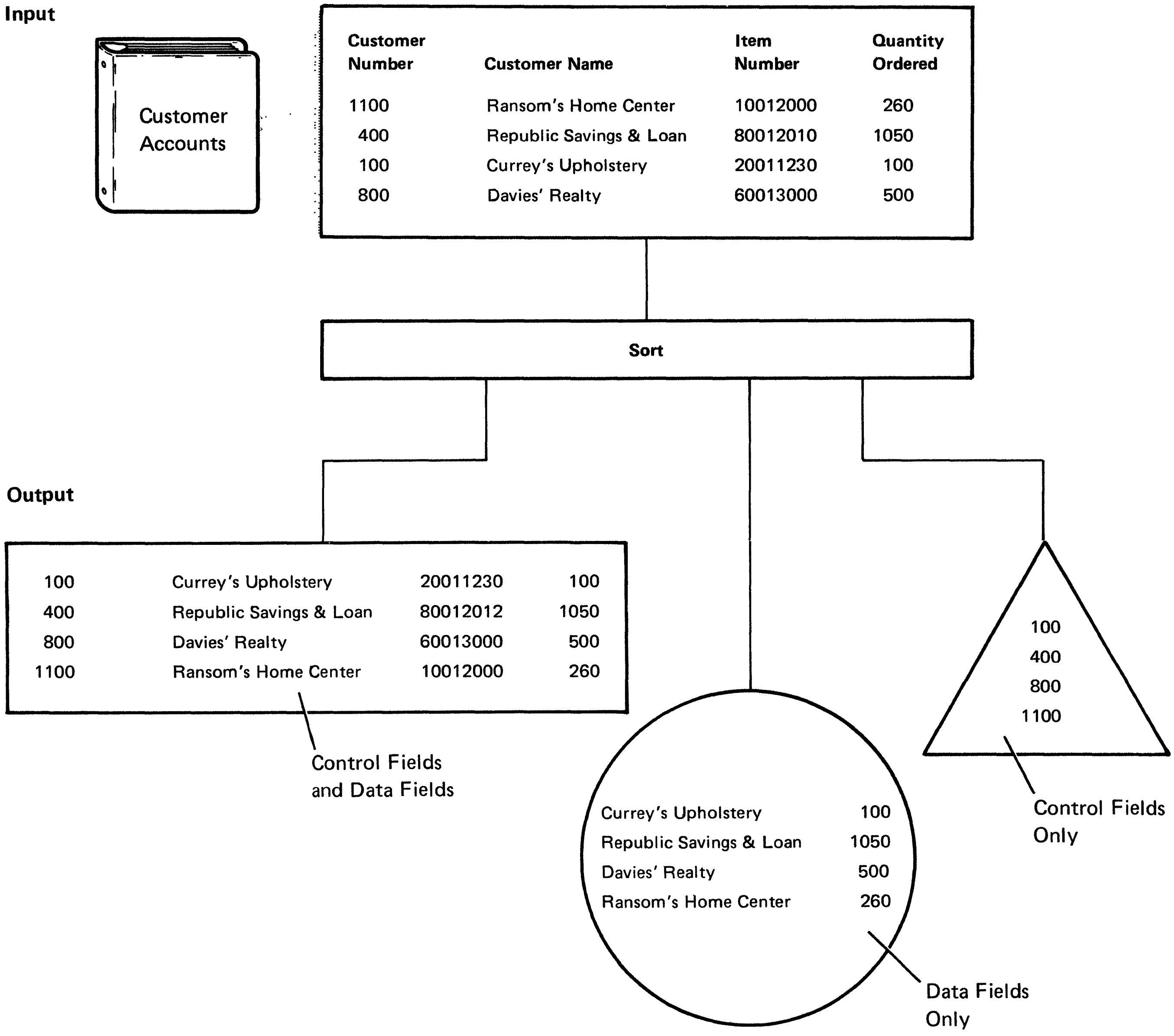
The sort program performs operations that generate the following types of sorted output:

- Regularly sorted files (**SORTR**)

  The records are ordered as you specify to the sort program. Specified data fields (pieces of information in a file) can be included with control fields in these sorted records. A single output record is created for each input record the sort program selects for processing. The output from a regularly sorted file is a file of sorted records that can contain:

  - Control fields only (Control fields are fields in the input record that are used to sort the records.)
  - Data fields only (Data fields are fields within the input and work records that have no effect on the sorting process. They are written into the output file as specified.)
  - Control fields and data fields.

  The sort program generates a regularly sorted file as follows (the input records are sorted by customer number):



Customer Number	Customer Name	Item Number	Quantity Ordered
1100	Ransom's Home Center	10012000	260
400	Republic Savings & Loan	80012010	1050
100	Currey's Upholstery	20011230	100
800	Davies' Realty	60013000	500

100	Currey's Upholstery	20011230	100
400	Republic Savings & Loan	80012012	1050
800	Davies' Realty	60013000	500
1100	Ransom's Home Center	10012000	260

Control Fields and Data Fields

Currey's Upholstery	100
Republic Savings & Loan	1050
Davies' Realty	500
Ransom's Home Center	260

Data Fields Only

100
400
800
1100

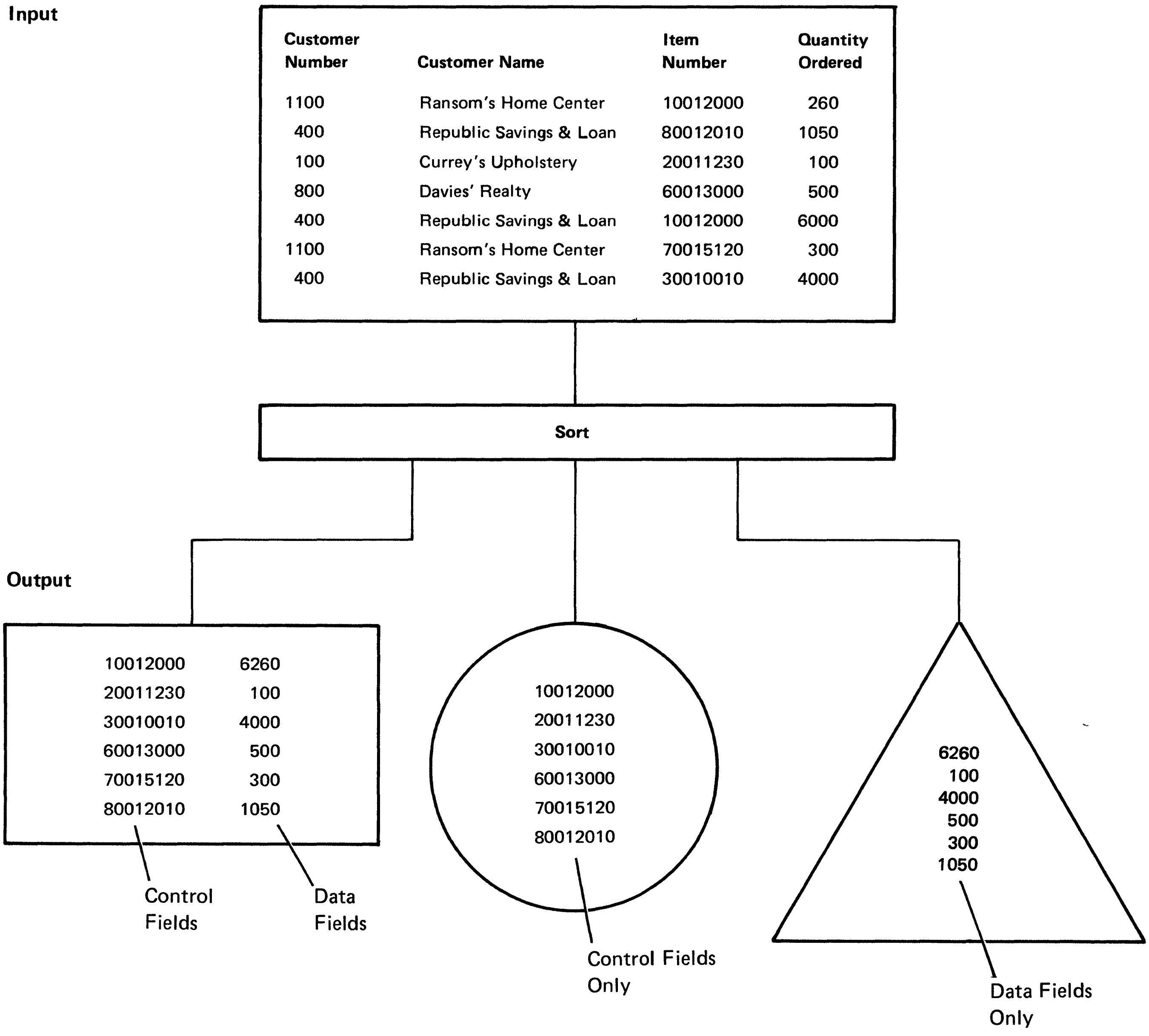
Control Fields Only

- Sorted files with accumulated totals, sometimes called summary output **(SORTRS)**.

  A total or totals can be accumulated in the sorted records. A single output record is created for each unique field used to sort records, with some values added together for a single total. A summary sorted file contains:

  - Summary fields only
  - Control fields only
  - Control fields and summary fields.

  The sort program generates a summary sort as follows (the input file is sorted by item number):



Input

Customer Number	Customer Name	Item Number	Quantity Ordered
1100	Ransom's Home Center	10012000	260
400	Republic Savings & Loan	80012010	1050
100	Currey's Upholstery	20011230	100
800	Davies' Realty	60013000	500
400	Republic Savings & Loan	10012000	6000
1100	Ransom's Home Center	70015120	300
400	Republic Savings & Loan	30010010	4000

Output

Control Fields	Data Fields
10012000	6260
20011230	100
30010010	4000
60013000	500
70015120	300
80012010	1050

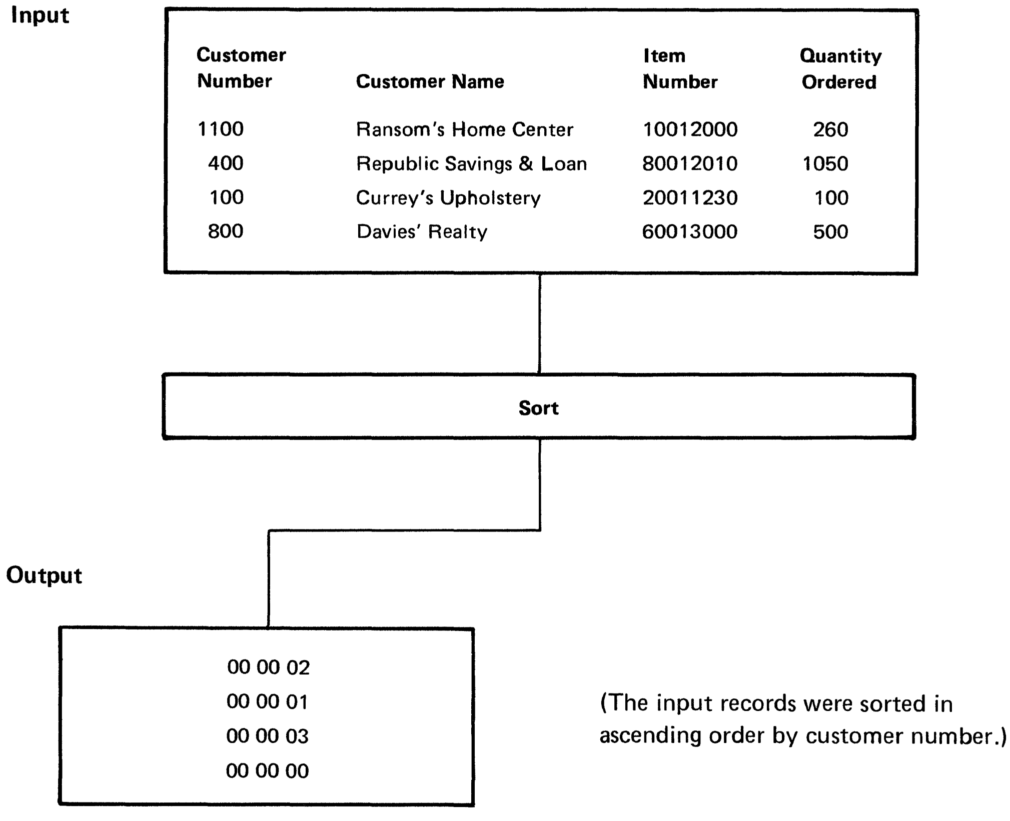
10012000
20011230
30010010
60013000
70015120
80012010

6260
100
4000
500
300
1050

- Files of record addresses; sometimes called addrout sort (**SORTA**).

  This sorted output contains only locations of records in files. The output from a record address sort consists of relative record numbers of some or all the records in the input file, although the relative record numbers never appear in the input file. The relative record number value of the first record in the input file is always zero.

  The sort program generates a record address sort as follows:



Input

Customer Number	Customer Name	Item Number	Quantity Ordered
1100	Ransom's Home Center	10012000	260
400	Republic Savings & Loan	80012010	1050
100	Currey's Upholstery	20011230	100
800	Davies' Realty	60013000	500

Output

```
00 00 02
00 00 01
00 00 03
00 00 00
```

(The input records were sorted in ascending order by customer number.)

Relative Record Numbers Only

Several types of files can be used as input to a sort job: indexed, sequential, or direct disk files. However, once a file is sorted, the output will always be a single file with sequential organization.

As many as eight files can be sorted at any one time with either all or some of the records in the input file(s) being written into the output file. If you sort more than one file at a time, all the input files must have the same record length.

The sort program does not require unique control fields. However, if duplicate control fields are found in the input records of jobs whose output will be a file containing regularly sorted data or record addresses, the records with the same control field will be grouped together in the output file. You can, however, specify that the grouped output records be written into the output file in the same or the opposite order they had in the input file. Otherwise, their order will be unpredictable.

The sort program ignores any descriptions associated with the files. Sort treats both the input and output files as unformatted strings of characters, so fields are defined by starting and ending positions rather than by field names. The only format characteristics used by the sort program (other than the record length) are those you specify using the sort specifications statements.

Regularly Sorted Files

Regularly sorted files are the most commonly used sorted output. For this kind of output file, a single output record is created for each input record selected. Then the records are placed in the order you specified and contain the data you selected from the input records.

Regularly sorted files can consist of the following information:

- Records containing both the field used to sort the records (the control field) and data fields

- Records containing data fields only

- Records containing control fields only.

You can specify that other information be included in the sorted output, such as:

- Record identification information

  This information identifies different types of records within the input file, if multiple record types are present, and indicates the input records that will be processed.

You can also specify the following:

- Control field identification

  The sort specifications identify the location of the control fields within the input string for each record type. The location of a control field is identified by its starting and ending positions.

- Output format

  The output format indicates the fields within each input record that will be included in the output record. This format must be defined for each record type. The entire input record, or any part of it, can be placed in the output record. If only portions of the input record will be included, each portion must be identified by starting and ending positions. Each portion is assigned an order by the sort specifications statements; this order determines how the input portions will be placed in the output record.

The data in regularly sorted files can be rearranged from the highest value to the lowest value (descending order), or from the lowest value to the highest value (ascending order). (See *Collating Sequences* in Chapter 2 for more information.) For example, assume you want to sort the following records in a file by customer number with the smallest customer number written into the output file first. You also want all the data in the input file, including the customer numbers, to be written into the output file. Assume your input file contains the following records:

Input Records

Customer Number	Customer Name	Item Number	Quantity Ordered
1100	Ransom's Home Center	10012000	260
400	Republic Savings & Loan	80012012	1050
100	Currey's Upholstery	20011230	100
800	Davies' Realty	60013000	500

After the records are sorted, their order in the output file is changed to the following:

Customer Number	Customer Name	Item Number	Quantity Ordered
100	Currey's Upholstery	20011230	100
400	Republic Savings & Loan	80012012	1050
800	Davies' Realty	60013000	500
1100	Ransom's Home Center	10012000	260

Control Field   Data Fields

In the preceding example, not only is the order of the records changed to ascending order; also the control field (customer number) and all the data fields (customer name, item number, and quantity ordered) are written into the output file.

In the previous example, had you sorted the records by customer number and specified that customer names and quantities ordered be written into the output file, your output would look like the following:

```
                                  Quantity
Customer Name                     Ordered

Currey's Upholstery                 100
Republic Savings & Loan            1050
Davies' Realty                      500
Ransom's Home Center                260
```

Data Fields Only

In the preceding example, the output file contains data fields only. The data in the control field (customer number) is not written into the output file.

If you had sorted the records by customer number (in ascending order) and specified that customer numbers only be written into the output file, your sorted file would look like the following:

Customer Numbers
100
400
800
1100

Control Fields Only

Here the output contains control fields only. Data fields are not written into the output file.

Not only can you use the sort program to rearrange data in your files, but you can also reformat your data. In this example, assume you want a file containing all your customers names in alphabetical (ascending) order followed by customer number, item number, and quantity ordered. The format of the records in the input file, however, differs. Therefore, you want to change the format of the records so the customer name becomes the first field in each record followed by the other data fields. To do this, you could reformat the following input records and sort them by customer name:

Customer Number	Customer Name	Item Number	Quantity Ordered
400	Republic Savings & Loan	80012010	1050
100	Currey's Upholstery	20011230	100
800	Davies' Realty	60013000	500
1100	Ransom's Home Center	10012000	260

Once the records are sorted, your output would look like the following:

Customer Name	Customer Number	Item Number	Quantity Ordered
Currey's Upholstery	100	20011230	100
Davies' Realty	800	60013000	500
Ransom's Home Center	1100	10012000	260
Republic Savings & Loan	400	80012010	1050

You could also drop records from this file. For example, assume you want to sort the previous records by customer number with the smallest number written into the output file first (ascending order) and include only those records with an order of 500 or more items in the output. Also, all the data in the input file will be written into the output file. After the records are sorted, your output will contain the following records only:

Customer Number	Customer Name	Item Number	Quantity Ordered
400	Republic Savings & Loan	80012010	1050
800	Davies' Realty	60013000	500

Quantity Ordered

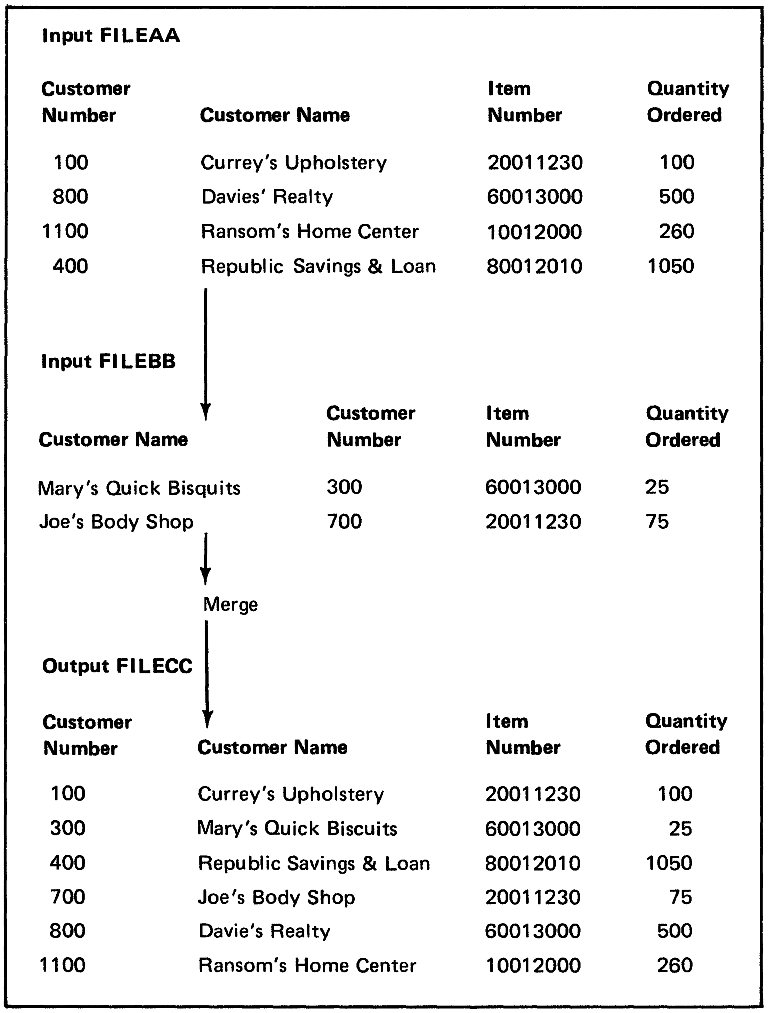
In the preceding example, the sort program omitted the records containing orders of less than 500 items, or the following records:

Customer Number	Customer Name	Item Number	Quantity Ordered
100	Currey's Upholstery	20011230	100
1100	Ransom's Home Center	10012000	260

Quantity Ordered

The sort program can also merge files. It can combine the records from two or more input files into one output file by **merging** them as shown in the following example:

In this example, two input files (FILEAA and FILEBB) containing the same type of fields and the same record length are merged into one file (FILECC).



Input FILEAA

Customer Number	Customer Name	Item Number	Quantity Ordered
100	Currey's Upholstery	20011230	100
800	Davies' Realty	60013000	500
1100	Ransom's Home Center	10012000	260
400	Republic Savings & Loan	80012010	1050

Input FILEBB

Customer Name	Customer Number	Item Number	Quantity Ordered
Mary's Quick Bisquits	300	60013000	25
Joe's Body Shop	700	20011230	75

Output FILECC

Customer Number	Customer Name	Item Number	Quantity Ordered
100	Currey's Upholstery	20011230	100
300	Mary's Quick Biscuits	60013000	25
400	Republic Savings & Loan	80012010	1050
700	Joe's Body Shop	20011230	75
800	Davie's Realty	60013000	500
1100	Ransom's Home Center	10012000	260

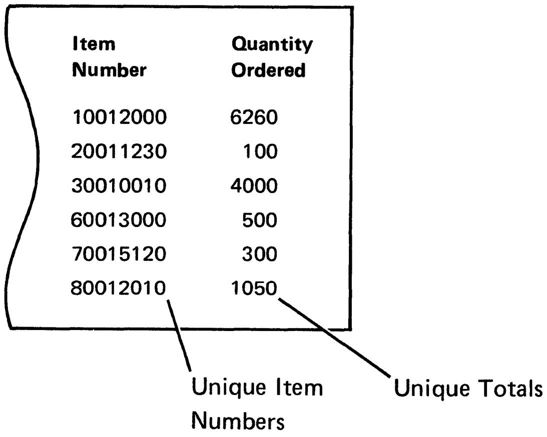
Although the records in FILEBB have a different format than those in FILEAA, (customer number is the first field in FILEAA but the second field in FILEBB), this difference in format does not prohibit you from doing a merge. Actually, the data in the files can be rearranged, reformatted, dropped, and sorted either before, during, or after the merge.

Sorted Files with Accumulated Totals (or Summary Data)

Usually, you produce sorted files with accumulated totals (or summary data) when you want to add data in certain fields in records such as adding the number of items ordered and producing a file containing the totals. Assume, for example, you want the following records sorted by item number (in ascending order) with quantity ordered as a summary data field.

Customer Number	Customer Name	Item Number	Quantity Ordered
1100	Ransom's Home Center	10012000	260
400	Republic Savings & Loan	80012010	1050
100	Currey's Upholstery	20011230	100
800	Davies' Realty	60013000	500
400	Republic Savings & Loan	10012000	6000
1100	Ransom's Home Center	70015120	300
400	Republic Savings & Loan	30010010	4000

Within the file, an output record is created that consists of a unique item number identifying the total number of items ordered (quantity ordered) for that item number. One output record is created for item number 10012000, for which a total of 6260 (260 + 6000) items were ordered.



Item Number	Quantity Ordered
10012000	6260
20011230	100
30010010	4000
60013000	500
70015120	300
80012010	1050

Unique Item Numbers
Unique Totals

When you generate an output file of summary data, the output can be any of the following kinds of records:

- Records containing data fields with accumulated totals only
- Records containing both data fields with accumulated totals and control fields
- Records containing data fields with accumulated totals, control fields, and other data fields.

You can generate the same kinds of output for a sorted file of accumulated totals as you can for a regularly sorted file, except that you cannot generate accumulated totals when your output is a regularly sorted file.

Files of Record Addresses

The sort program can resequence records selected from a *single* file and generate a record address file that reflects the new order. The record address file will contain 3-character relative record numbers only, of some or all the records in the input file. Relative record numbers identify the physical location of records in a file. The relative record number of the first record in an input file is always zero (00 00 00).

This type of output is normally used with programs that process files of record addresses (addrout files). During processing, a single output record is also created for each input record selected for sorting. Only one input file at a time can be used to produce this type of output.

A sort specification statement is used to specify that the output will be a record address file. In addition, sort specifications statements can provide the sort program with the following information:

- Record identification information

  Record identification information is used to identify record types within the physical input file whenever multiple record types are present. This information is also used to select a subset of record types for processing by the utility.

- Control fields

  A control field is the key definition that is used to reorder the records. Control fields are defined by their starting and ending positions within the input record.

Assume, for example, that you want to sort the following records by customer number and produce an output file containing only record addresses (locations). The records will be sorted by customer number in ascending order (with the smallest customer number being written into the output file first).

Customer Number	**Customer Name**	**Item Number**	**Quantity Ordered**
1100	Ransom's Home Center	10012000	260
400	Republic Savings & Loan	80012010	1050
100	Currey's Upholstery	20011230	100
800	Davies' Realty	60013000	500
400	Republic Savings & Loan	10012000	6000
1100	Ransom's Home Center	70015120	300
400	Republic Savings & Loan	30010010	4000

For purposes of this example, the relative record numbers associated with the input records are shown with the following data records. Keep in mind, however, that relative record numbers never appear in the records with the data:

	Customer Number	Customer Name	Item Number	Quantity Ordered
00 00 00	1100	Ransom's Home Center	10012000	260
00 00 01	400	Republic Savings & Loan	80012010	1050
00 00 02	100	Currey's Upholstery	20011230	100
00 00 03	800	Davies' Realty	60013000	500
00 00 04	400	Republic Savings & Loan	10012000	6000
00 00 05	1100	Ransom's Home Center	70015120	300
00 00 06	400	Republic Savings & Loan	30010010	4000

Relative Record Numbers

Data in Input Records

The order of the records with duplicate control fields cannot be determined unless you specify that the sort program write them into the output file either in the same or the opposite order they had in the input file.

Note: Relative record numbers never appear in the input file. They are shown here only for the purposes of this example.

After the records are sorted by customer number, the output looks like the following:

00 00 02
00 00 01
00 00 04
00 00 06
00 00 03
00 00 00
00 00 05

Relative Record Numbers

In the preceding output, the relative record number (00 00 02) for customer number (100, which is third in the input file) is placed first in the output file.

What You Need to Use Sort

To use the sort program, you need the following:

Input

Input A: One or more **input files** you want to sort

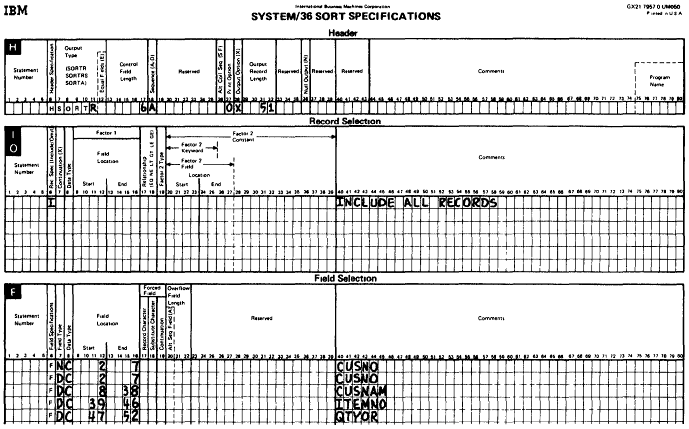
The input file can be an indexed, sequential, or direct file with fixed length records; sort processes each file sequentially.

Input File

Customer Number	Customer Name	Item Number	Quantity Ordered
1100	Ransom's Home Center	10012000	260
400	Republic Savings & Loan	80012010	1050
100	Currey's Upholstery	20011230	100
800	Davies' Realty	60013000	500

Input B: **Sort specifications** that tell the sort program how to arrange the data provided by your input files

The sort specifications are your instructions that tell the sort program how you want the input files to be sorted.



IBM

International Business Machines Corporation

SYSTEM/36 SORT SPECIFICATIONS

GX21 7957 0 UM050 Printed in U.S.A.

Header

Statement Number	Header Specification	Output Type (SORTR SORTRS SORTA)	Equal Fields (E)	Control Field Length	Sequence (A,D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
	H	SORTR		6	A			0	X	51						

Record Selection

Statement Number	Rec Spec (Include/Omit)	Continuation (X)	Data Type	Factor 1 Field Location Start	Factor 1 Field Location End	Relationship (EQ NE LT GT LE GE)	Factor 2 Type	Factor 2 Constant / Keyword / Field Location	Comments
	I								INCLUDE ALL RECORDS

Field Selection

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Alt Seq Field (A)	Overflow Field Length	Reserved	Comments
	F	N	C	2	7							CUSNO
	F	D	C	2	7							CUSNO
	F	D	C	8	38							CUSNAM
	F	D	C	39	46							ITEMNO
	F	D	C	47	52							QTYOR

Input C: Something that will tell the sort program you want to run a sort job

You can tell the sort program that you want to run a sort job in one of two ways:

- By using the **SORT** command
- By using a **procedure** you create with Operation Control Language (OCL) statements

The SORT command or OCL statements tell the system that you want to process a sort job.

```
SORT ITEMBALN,EX1,EX1OUT,500,JMGLIB,Y
```

```
// LOAD #GSORT
// FILE NAME-INPUT,LABEL-ITEMBALN
// FILE NAME-WORK,LABEL-SORTWORK,
     BLOCKS-20,RETAIN-S
// FILE NAME-OUTPUT,LABEL-CASLY,
     BLOCKS-10,RETAIN-T
// RUN
// SOURCE REORDER,JMGLIB
```

Process: Once you provide the sort program with input, sort then processes the input in two steps:

- The sort program first reads and processes the sort specifications to determine the type of sort to be done.
- The sort program then reads input files and sorts them using a work file that is on disk.

Output: The output is a file sorted according to your specifications. The sorted file can contain:

- Parts or all of the records in the input file.
- Summarized fields for each record type in the input file.
- The relative record numbers of records in the input file.

Note: You can also call the system's sort program directly from an Assembler or COBOL program. See Appendix C, *Calling Sort From an Assembler Program* for more information.

Getting Started

Now that you know what the sort program needs to run a sort, you can code your sort specifications as will be discussed in Chapter 2.

Chapter 2. Sort Coding Considerations

Designing a Sort Job

When designing the sort job, consider each task required to produce the output you want. To design efficient and effective sort jobs, you should decide the following:

- The kind of sorted output you want
- Which field(s) in the input records will serve as the control field(s) to actually do the sorting
  - Whether control fields can contain different data types (if so, what they are and when and how to use them)
  - How to sort records having duplicate control fields
  - How many characters control fields can contain
  - How many control fields are allowed in a job.
- The order in which the data will be sorted
- Whether an alternative collating sequence is allowed (in place of the standard collating sequence)
- Which input records (and fields within those records) will be selected and/or omitted and written into the output file
  - How the sort program compares the data in the input file(s), and the relationship(s) that must exist in the data being compared
  - What kind of data can be compared
  - How to specify the data that will be compared.
- Which data fields in the input file will be written into the output file
- Whether you want to put comments in the sort specifications
- What to do if an error occurs when you run a sort job.

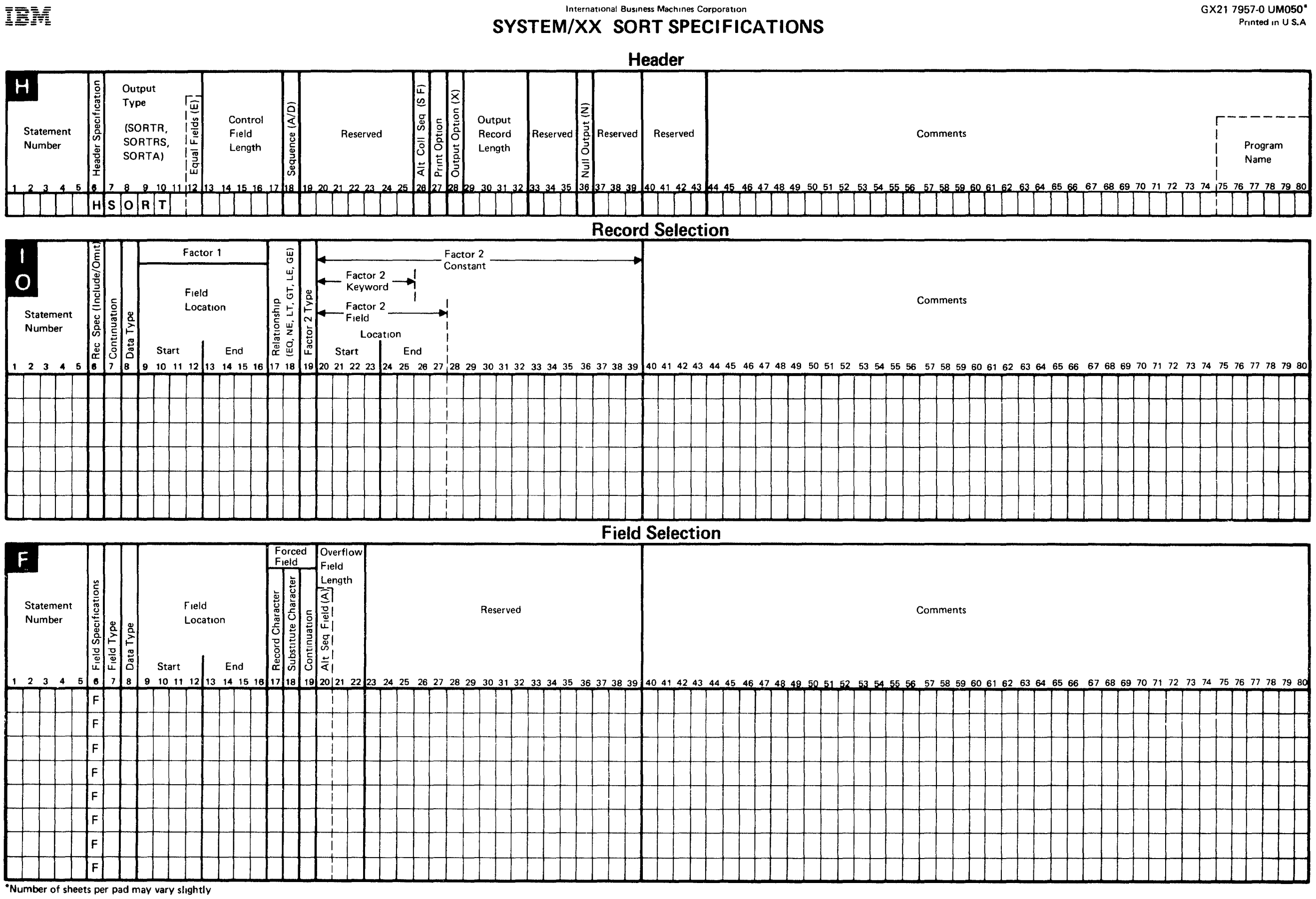
Consider what you want to do when you want to sort a file. Consider why you want to sort the records and how you want your data to look when the sort is complete. To help you create the most efficient sort job possible, tasks you should consider when designing and coding a job are discussed and examples are provided in the following topics. Examples of complete sort jobs are also found at the end of this chapter.

Become Familiar With the Sort Coding Form

A tool useful in the designing and coding processes is the sort specifications coding form. Before you design and code your job, you should become familiar with the coding form. It will contain your instructions that identify to the sort program the operations to be performed and the records and fields to be used in the sorting process. (It can also serve as a backup copy of the sort job.) The sort specifications coding form contains three types of specifications:

1. A **header specification** H that describes the type of sort you want to run.
2. **Record selection specifications** I O describe which input records you want to include in or omit from the sort.
3. **Field selection specifications** F indicate how you want the records to be sorted and identify which data fields will be written into the output file.

The following shows a sort specifications coding form. See the reference section of this manual for information about the entries you make in each column.



IBM

International Business Machines Corporation

SYSTEM/XX SORT SPECIFICATIONS

GX21 7957-0 UM050*
Printed in U S.A

Header

H

Statement Number (1–5) | Header Specification (6) | Output Type (SORTR, SORTRS, SORTA) (7–11) | Equal Fields (E) (12) | Control Field Length (13–17) | Sequence (A/D) (18) | Reserved (19–25) | Alt Coll Seq (S F) (26) | Print Option (27) | Output Option (X) (28) | Output Record Length (29–32) | Reserved (33–35) | Null Output (N) (36) | Reserved (37–39) | Reserved (40–43) | Comments (44–74) | Program Name (75–80)

H S O R T

Record Selection

I
O

Statement Number (1–5) | Rec Spec (Include/Omit) (6) | Continuation (7) | Data Type (8) | Factor 1 Field Location Start (9–12) End (13–16) | Relationship (EQ, NE, LT, GT, LE, GE) (17–18) | Factor 2 Type (19) | Factor 2 Constant (20–39) | Factor 2 Keyword | Factor 2 Field Location Start (20–23) End (24–27) | Comments (40–80)

Field Selection

F

Statement Number (1–5) | Field Specifications (6) | Field Type (7) | Data Type (8) | Field Location Start (9–12) End (13–16) | Forced Field: Record Character (17) Substitute Character (18) | Continuation (19) | Overflow Field Length (20–22) | Alt Seq Field (A) (20) | Reserved (23–39) | Comments (40–80)

F
F
F
F
F
F
F
F

*Number of sheets per pad may vary slightly

What Kind of Sorted Output Do You Want?

When designing your sort job, first decide which type of sorted output you want. You can specify one of the following:

- **SORTR** for regularly sorted output
- **SORTRS** for sorted output containing data that was added together, or summarized (called summarized data)

  For example, this type of sorted output could produce a file containing a single total of all balances owed by each customer.
- **SORTA** for sorted record addresses.

  For example, you could sort data, such as customer names, in a particular order and produce an output file containing only 3-byte relative record numbers indicating positions of records in a file.

An example of the entry for a job producing regularly sorted output follows:

Header

H Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
1 2 3 4 5	6	7 8 9 10 11	12	13 14 15 16 17	18	19 20 21 22 23 24 25	26	27	28	29 30 31 32	33 34 35	36	37 38 39	40 41 42 43	44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	75 76 77 78 79 80
	H	S O R T R														

Which Input Field(s) Will Control the Sorting

Once you know which type of sorted output you want, you should next decide:

- Which field(s) in the input records will control the sorting process, that is, serve as the **control field(s)**

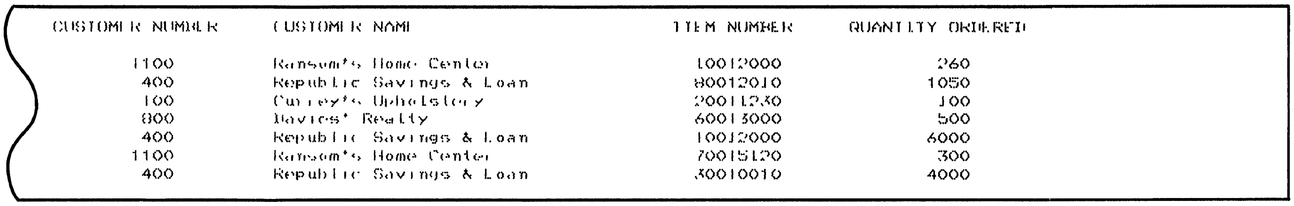
  Any field (containing 1 to 256 characters) in the input record can be used as the control field to sort the records.

- Which sequence you want to use for the sorted output.

Assume you want to sort the following records in a file. You want the records to be sorted by customer number with the lowest customer number written into the output file first (ascending order). You also want all the data in the input file to be written into the output file.

Assume the input file has the following format:

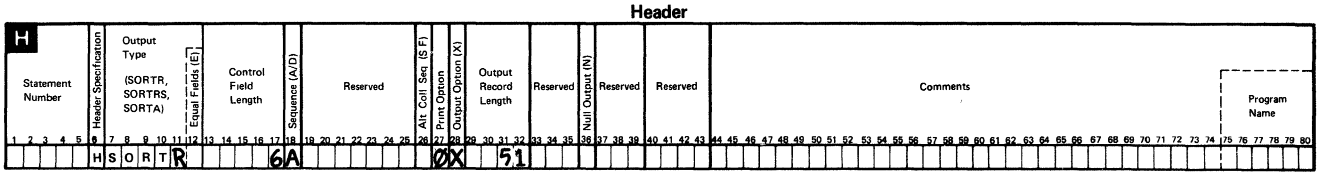
Record Code (RCODE)	Customer Number (CUSNO)	Customer Name (CUSNAM)	Item Number (ITEMNO)	Quantity Ordered (QTYOR)	
1	2 7	8 38	39 46	47 52	



```
CUSTOMER NUMBER    CUSTOMER NAME                       ITEM NUMBER    QUANTITY ORDERED

     1100          Ransom's Home Center                 10012000             260
      400          Republic Savings & Loan              80012010            1050
      100          Curley's Upholstery                  20011230             100
      800          Davies' Realty                       60013000             500
      400          Republic Savings & Loan              10012000            6000
     1100          Ransom's Home Center                 70015120             300
      400          Republic Savings & Loan              30010010            4000
```

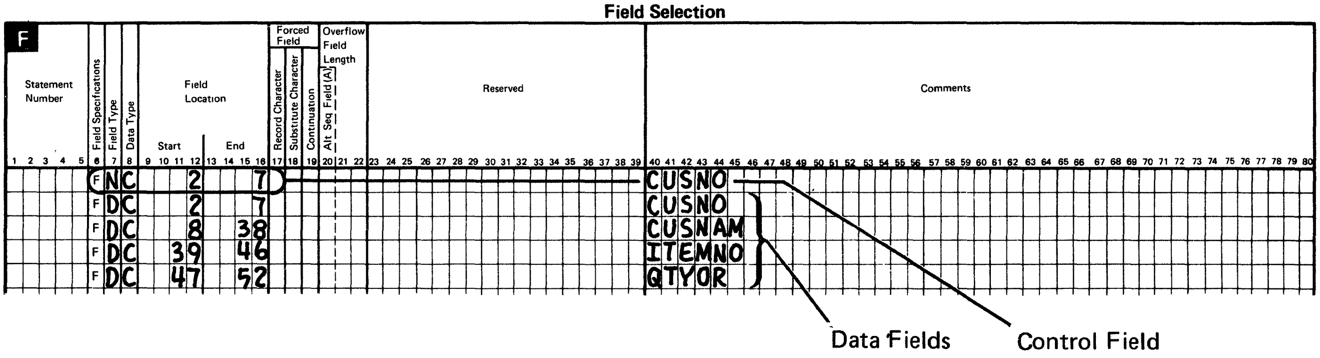
Note: In this and other (following) examples, the record code will not be shown in the input or output records.

To sort this input file using customer number as the control field, you would define the sort specifications as follows:



Header

Statement Number (1-5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7-11)	Equal Fields (E) (12)	Control Field Length (13-17)	Sequence (A/D) (18)	Reserved (19-25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29-32)	Reserved (33-35)	Null Output (N) (36)	Reserved (37-39)	Reserved (40-43)	Comments (44-74)	Program Name (75-80)
	H	SORTR		6	A			Ø	X	51						



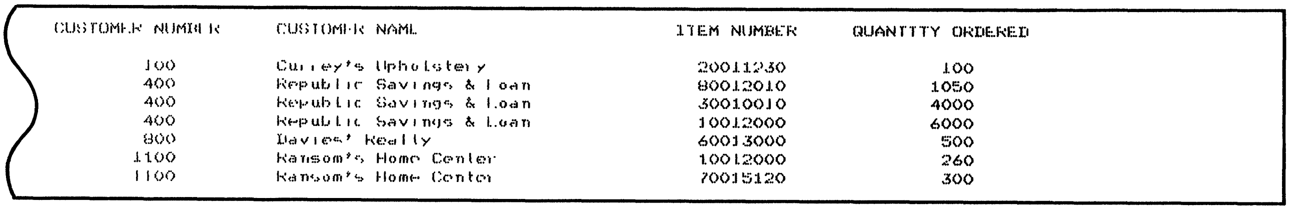
Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	2	7						CUSNO
	F	D	C	2	7						CUSNO
	F	D	C	8	38						CUSNAM
	F	D	C	39	46						ITEMNO
	F	D	C	47	52						QTYOR

The output option column of the header specification lets you indicate whether you want to drop control fields from output records after the records are sorted. In this example, the letter X was placed in the output option column, indicating that only data fields will be written into the output records.

Because all the records are included in the sort, no record selection statements are needed.

The sorted output will look similar to the following:



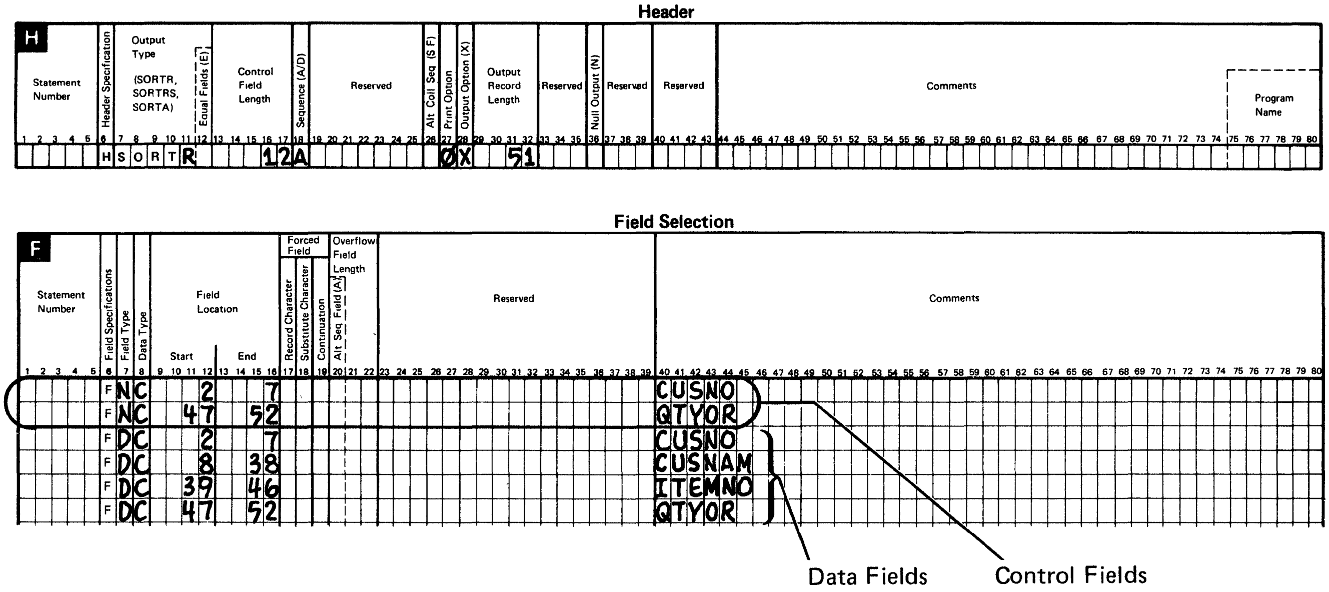
```
CUSTOMER NUMBER   CUSTOMER NAME                    ITEM NUMBER    QUANTTTY ORDERED

      100         Curley's Upholstery                20011230          100
      400         Republic Savings & Loan            80012010         1050
      400         Republic Savings & Loan            30010010         4000
      400         Republic Savings & Loan            10012000         6000
      800         Davies' Realty                     60013000          500
     1100         Ransom's Home Center               10012000          260
     1100         Ransom's Home Center               70015120          300
```

Note: Throughout this manual comments are included in many of the defined sort specifications. Comments are not required to run a sort job. They are used only to document certain information about a job.

In the preceding example, customer number is the control field. Control fields, which are defined in the header and field specifications, affect the work and output records only. The sort program does not change your input records (unless you specify that the output file be written over the input file); normally sort copies the input records into main storage for processing.

Also in this example, the order of the records with duplicate control fields cannot be determined unless you specify that the sort program write them into the output file in either the same or the opposite order they had in the input file.

You can use several control fields in a sort job. In the following example two control fields, customer number and quantity ordered, will sort the data in ascending order. The following sort specifications were used in the job:



Header

```
     HSORTR    12A        ØX  51
```

Field Selection

```
     FNC   2   7                       CUSNO
     FNC  47  52                       QTYOR
     FDC   2   7                       CUSNO
     FDC   8  38                       CUSNAM
     FDC  39  46                       ITEMNO
     FDC  47  52                       QTYOR
```

Data Fields

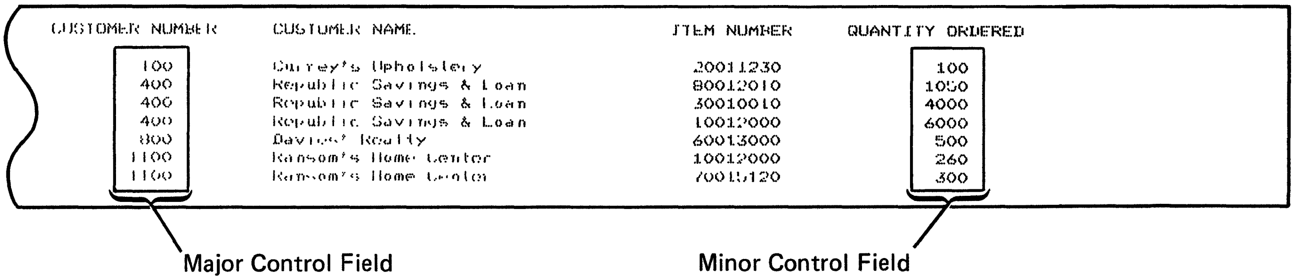
Control Fields

When you sort records using more than one control field, the first control field you define for a record type is always the **major control field**; it is used in the sorting process first. In the preceding example, customer number is the major control field.

The other control fields you define for that record type are **minor control fields**; they are used in the sorting process last. In the preceding example, quantity ordered is considered a minor control field. (Sort jobs with more than two control fields have intermediate control fields.)

The control fields′ orders are determined by the order in which they are defined in the field specifications statements. In the preceding example, customer number is defined first; it is the major control field. Quantity ordered is defined next; it is the minor control field.

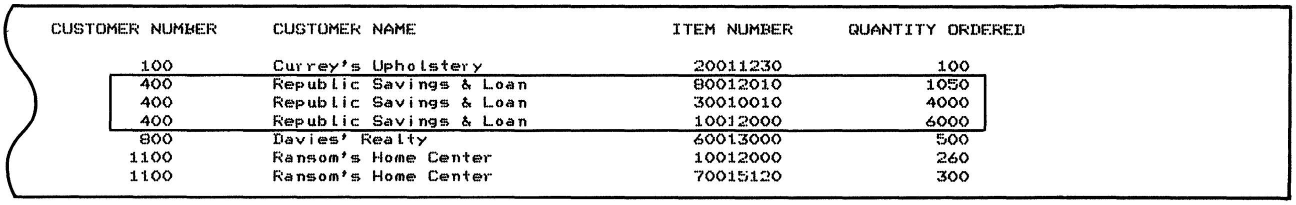
The output for the example using two control fields looks like the following:



```
CUSTOMER NUMBER    CUSTOMER NAME                       ITEM NUMBER    QUANTITY ORDERED

      100          Currey's Upholstery                  20011230            100
      400          Republic Savings & Loan              80012010           1050
      400          Republic Savings & Loan              30010010           4000
      400          Republic Savings & Loan              10012000           6000
      800          Davies' Realty                       60013000            500
     1100          Ransom's Home Center                 10012000            260
     1100          Ransom's Home Center                 70015120            300
```

Major Control Field
Minor Control Field

In the preceding example, the records are first sorted using customer number and sorted again using quantity ordered with the smallest number of items ordered for each customer placed first within that group of records. For example, the output for customer 400 looks like the following:



```
CUSTOMER NUMBER    CUSTOMER NAME                       ITEM NUMBER    QUANTITY ORDERED

      100          Currey's Upholstery                  20011230            100
      400          Republic Savings & Loan              80012010           1050
      400          Republic Savings & Loan              30010010           4000
      400          Republic Savings & Loan              10012000           6000
      800          Davies' Realty                       60013000            500
     1100          Ransom's Home Center                 10012000            260
     1100          Ransom's Home Center                 70015120            300
```

You can specify one or more control fields when sorting files containing one type of record or when sorting files containing multiple record types.

When the input file has more than one type of record:

- The control fields can be the same or different for each type of record.

  However, be sure you specify the correct data type (such as numeric or character data, in the field specifications) for each control field specified. If you do not, your sorted output will contain results you did not intend.

- The length of the control field can be the same or different for each type of record. See *Calculating Control Field Length(s)* later in this chapter.

What are the Different Types of Control Fields?

You can define different types of control fields in a sort job:

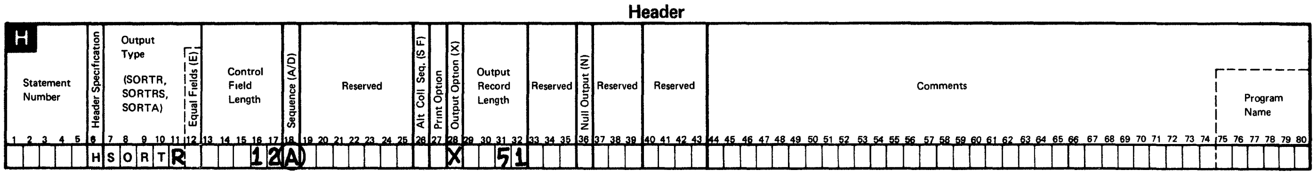
- Normal control fields
- Opposite control fields
- Forced control fields.

Normal Control Fields

A **normal control field (specified as N)** is any field in the input records that the sort program uses to sort the records. Normal control fields indicate that the data in that particular field will be sorted in ascending or descending order, whichever is specified in the header specification.

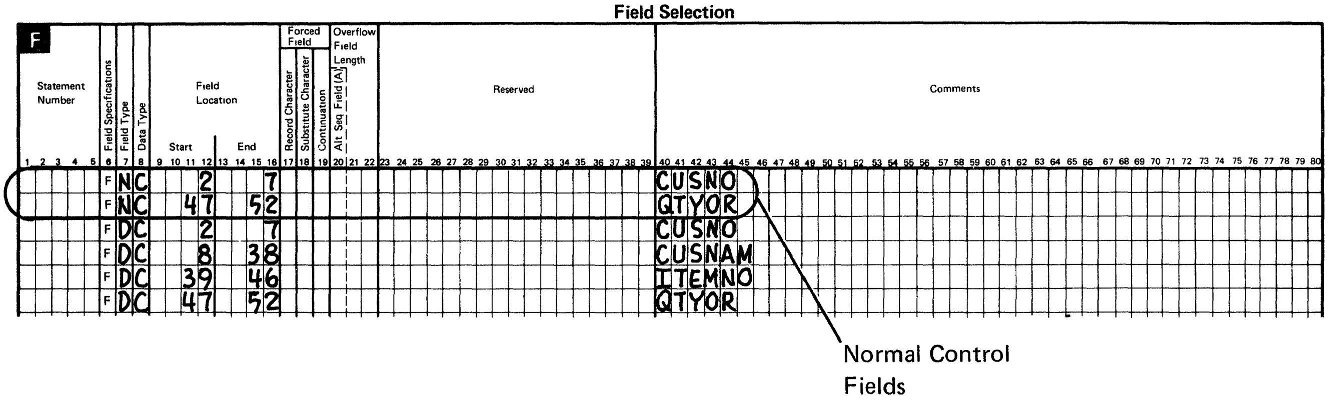
Ascending order means the record containing the item with the lowest value is placed first in the output file, whereas **descending order** means that the record containing the item with the highest value is the first record that is written into the output file. For more information about ascending and descending order, see *Determining the Order of Records in a File* later in this chapter.

The following example shows how to define a normal control field:



Header

Statement Number (1-5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7-12)	Equal Fields (E) (12)	Control Field Length (13-17)	Sequence (A/D) (18)	Reserved (19-25)	Alt Coll Seq. (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29-32)	Reserved (33-35)	Null Output (N) (36)	Reserved (37-39)	Reserved (40-43)	Comments (44-74)	Program Name (75-80)
	H	SORTR		12	A				X	51						



Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	2	7						CUSNO
	F	N	C	47	52						QTYOR
	F	D	C	2	7						CUSNO
	F	D	C	8	38						CUSNAM
	F	D	C	39	46						ITEMNO
	F	D	C	47	52						QTYOR

In our example, an A (for ascending order) is placed in the sequence column of the header specification and an N (for normal control field) is entered in the field type columns of the field specifications.

Normal control fields can contain both numeric data and data consisting of alphabetic characters.

When you use normal control fields that contain packed or zoned data and you want the data written into the output file, but you do not want the control field information in a form other than the original input form when it is written into the output records, you must describe the fields twice: once as a control field and once as a data field. Data fields are not involved in the sorting process and are not changed by the sort program.

Opposite Control Fields

Opposite control fields (specified as O), let you define a sequence for certain fields that is just the opposite of the sequence specified for other fields in the header specification.

Opposite control fields sort records in the following ways:

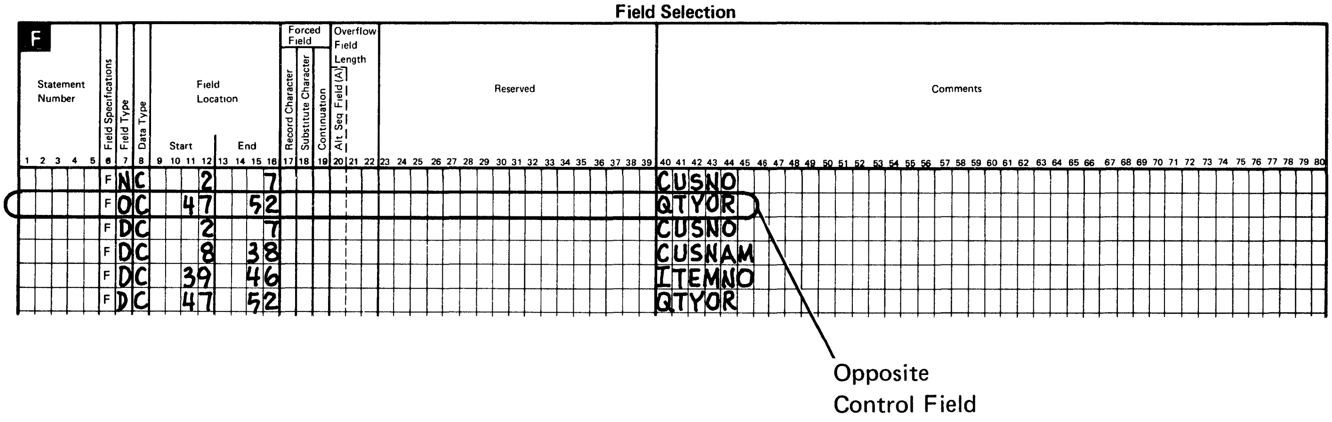
- In ascending order (if you specify descending order in the header specification sequence column)
- In descending order (if you specify ascending order in the header specification sequence column).

Like normal control fields, opposite control fields can also contain both alphameric and numeric data.

Suppose, in this example, you want to sort records and produce an output file containing customer orders. You want the customer numbers sorted in ascending order and the quantity ordered sorted in the opposite (descending) order. To do this, you would define your sort specifications as follows:

Header

Statement Number (1-5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7-11)	Equal Fields (E) (12)	Control Field Length (13-17)	Sequence (A/D) (18)	Reserved (19-25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29-32)	Reserved (33-35)	Null Output (N) (36)	Reserved (37-39)	Reserved (40-43)	Comments (44-74)	Program Name (75-80)
	H	SORTR		12	A				X	51						

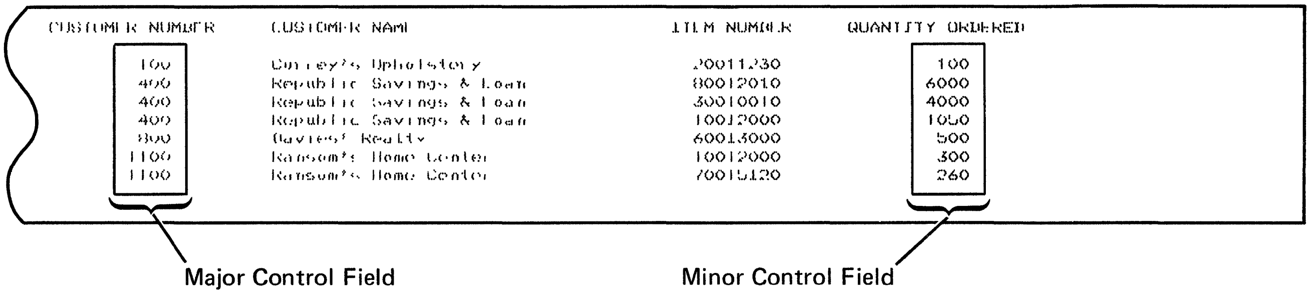


Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	2	7						CUSNO
	F	O	C	47	52						QTYOR
	F	D	C	2	7						CUSNO
	F	D	C	8	38						CUSNAM
	F	D	C	39	46						ITEMNO
	F	D	C	47	52						QTYOR

Opposite Control Field

Rather than quantity ordered also appearing in ascending order, the amounts appear in just the opposite order as shown in the following example:



```
CUSTOMER NUMBER   CUSTOMER NAME                    ITEM NUMBER   QUANTITY ORDERED

      100         Curley's Upholstery                20011230          100
      400         Republic Savings & Loan            80012010         6000
      400         Republic Savings & Loan            30010010         4000
      400         Republic Savings & Loan            10012000         1050
      800         Davies' Realty                     60013000          500
     1100         Ransom's Home Center               10012000          300
     1100         Ransom's Home Center               70015120          260
```

Major Control Field
Minor Control Field

If you use opposite control fields and you want their contents written into the output file, but you do not want the control field information in hexadecimal form in the output records, you must describe the fields twice: once as a control field and once as a data field. Data fields are not involved in the sorting process and are not changed by the sort program.

Forced Control Fields

You can use sort to force a character constant into a control field. **Forced control fields** are control fields you select to sort your records and with which you can substitute in or add information to your output records. Each forced control field can be only 1 character long, but you can specify more than one forced control field for a given field. To do this, you would define a field specification for each single character constant you want forced into a control field.

Note: Forced control fields do not affect your input records (unless you are writing your output over your input, which is not recommended). Only the work and output records are affected.

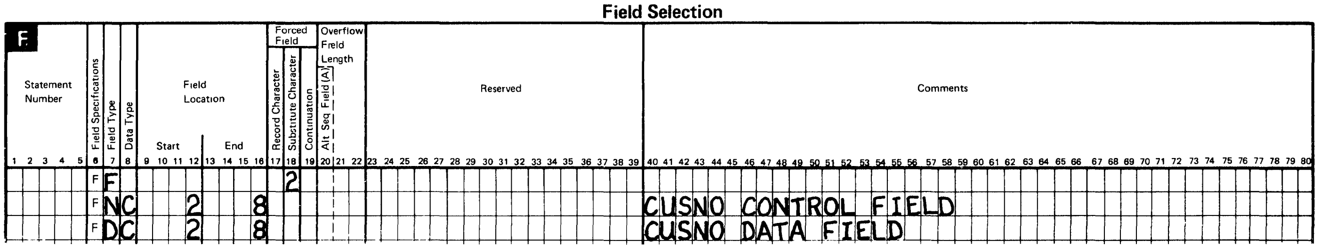
You can specify three types of forced control fields:

- Unconditional
- Conditional
- Force-all.

Unconditional Forced Field

An unconditional force automatically places your specified character constant into the next available position of the sort control field work record.

You can define an unconditional forced field as follows:



Field Selection

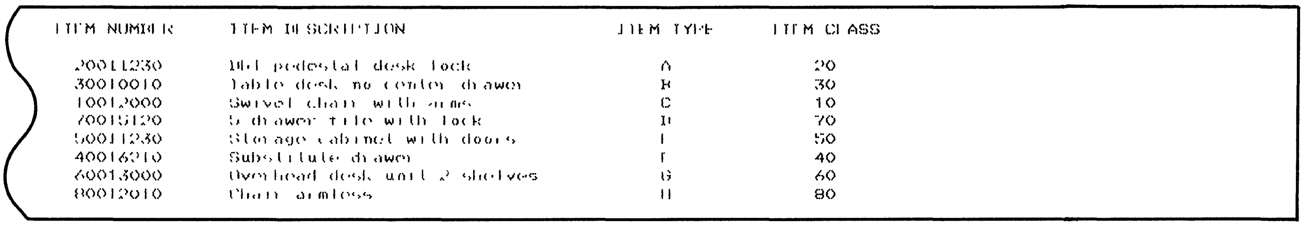
Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length Alt. Seq. Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	F					2				
	F	N	C	2	8						CUSNO CONTROL FIELD
	F	D	C	2	8						CUSNO DATA FIELD

In this example, character 2 is forced into the first available work record position (position 1); and the data from columns 2 through 8 of the input record is moved to positions 2 through 8 of the work record (rather than positions 1 through 7 of the work records).

In this example, assume you want to review your item master file for an upcoming sale on desk locks and table desks. Your output file should contain all data about table desks first followed by all data about desk locks. Within each group, the data will be sorted by item number. Because the item type for table desks is B and the item type for desk locks is A, a normal sort on item number would sort the desk locks before the table desks.

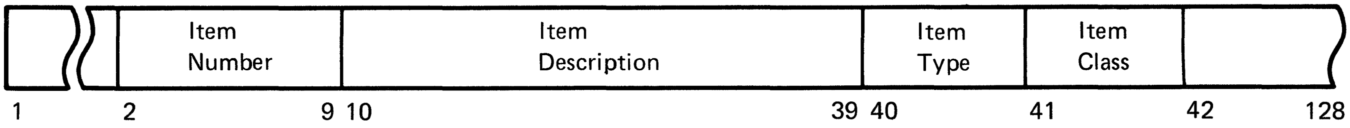
To cause the table desks records to be sorted before the desk lock records, you can use an unconditional force. You can force a 1 into the control field for table desks, and a 2 into the control field for desk locks and thereby cause the data to be ordered as needed.

The following input file was used:



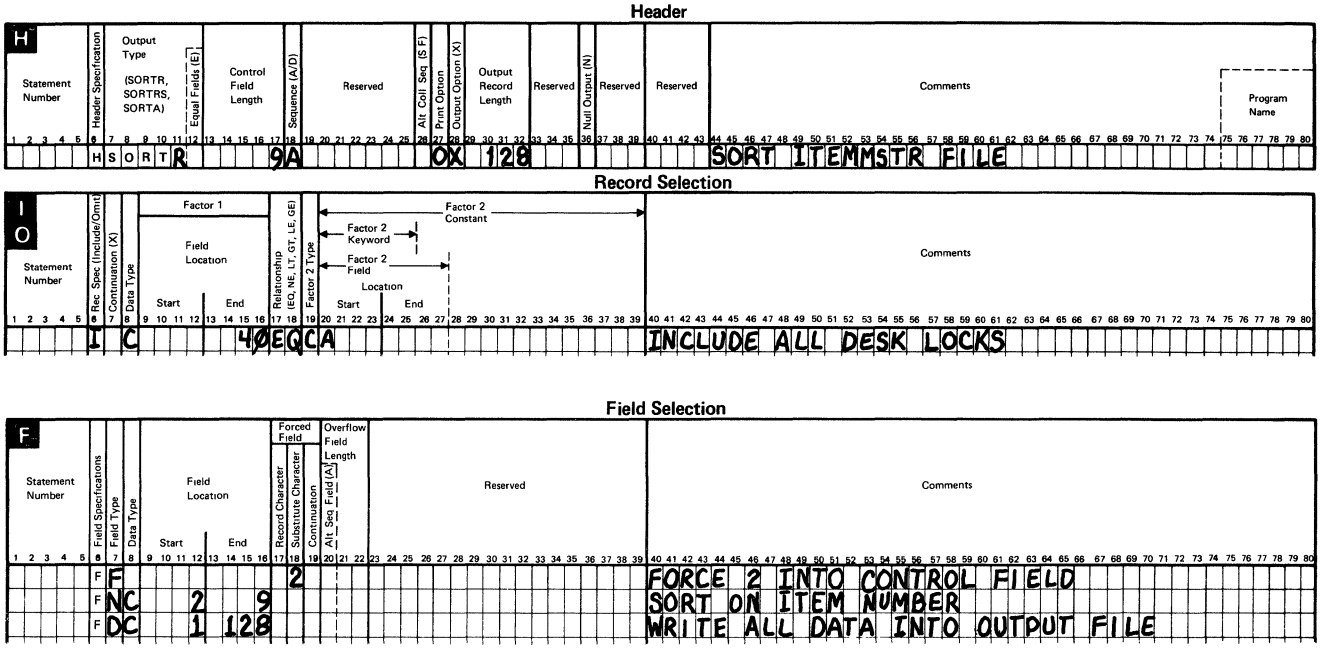
```
ITEM NUMBER    ITEM DESCRIPTION                  ITEM TYPE    ITEM CLASS

 20011230      Dbl pedestal desk lock                A            20
 30010010      Table desk no center drawer           B            30
 10012000      Swivel chair with arms                C            10
 70015120      5 drawer file with lock               D            70
 50011230      Storage cabinet with doors            E            50
 40016210      Substitute drawer                     F            40
 60013000      Overhead desk unit 2 shelves          G            60
 80012010      Chair armless                         H            80
```

The format of the input file follows:



	Item Number	Item Description	Item Type	Item Class	
1	2 – 9	10 – 39	40	41	42 – 128

The following example shows how you could define the sort specifications for this job. In the example, two sets of sort specifications are used in the job; one for sorting table desks and the other for sorting desk locks:



Header

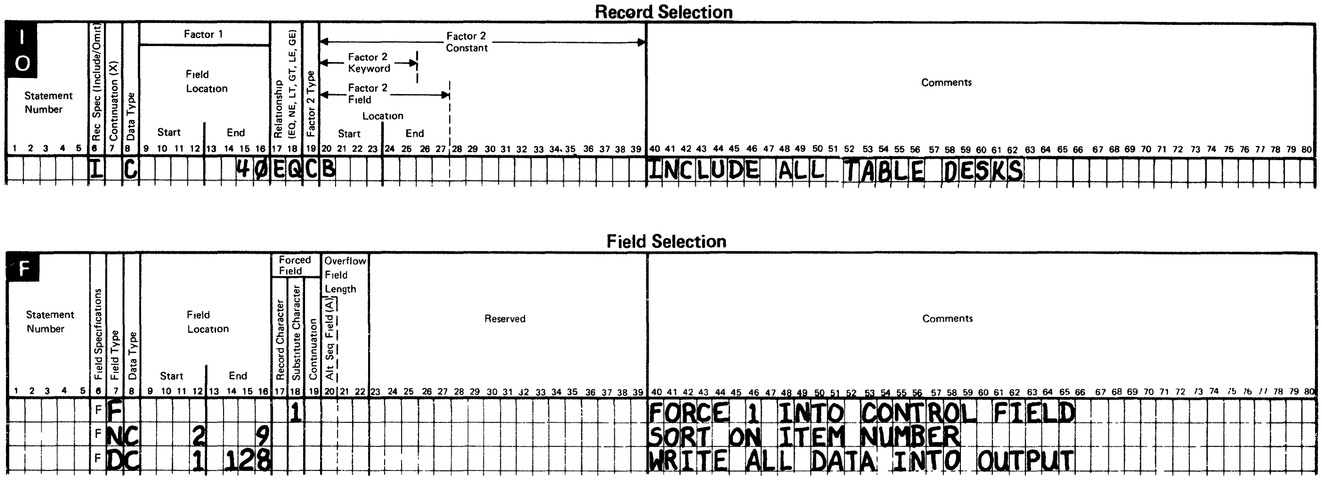
```
     HSORTR     9A        OX 128           SORT ITEMMSTR FILE
```

Record Selection

```
     I C      4ØEQCA                   INCLUDE ALL DESK LOCKS
```

Field Selection

```
     FF          2                     FORCE 2 INTO CONTROL FIELD
     FNC    2   9                      SORT ON ITEM NUMBER
     FDC    1 128                      WRITE ALL DATA INTO OUTPUT FILE
```



Record Selection

```
     I C      4ØEQCB                   INCLUDE ALL TABLE DESKS
```

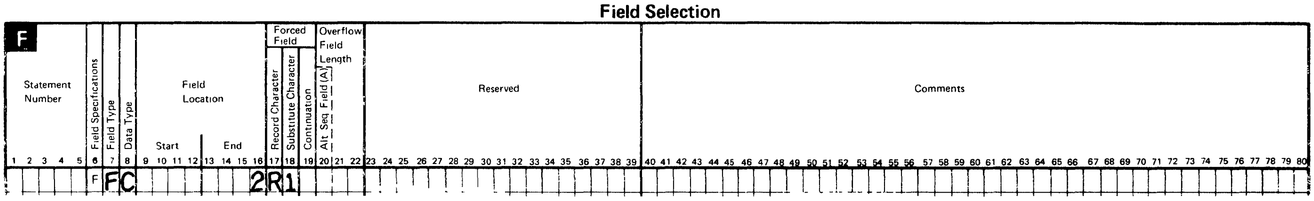
Field Selection

```
     FF          1                     FORCE 1 INTO CONTROL FIELD
     FNC    2   9                      SORT ON ITEM NUMBER
     FDC    1 128                      WRITE ALL DATA INTO OUTPUT
```

Conditional Forced Field

A conditional force allows you to test a character in the input records and force a character constant into the sort control field only if the test is successful. The test can be made on any character in the input record. You can compare the input character to see if it is equal to the character, zone, or digit portion of a specified constant.

When you define a conditional forced field, you can specify that the sort program either force the constant into the previously defined control field position, or into the next available control field position. The following is an example of how you would define a conditional force field:



Field Selection

Statement Number (1–5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9–12)	Field Location End (13–16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt. Seq. Field (A) (20–22)	Reserved (23–39)	Comments (40–80)
	F	F	C		2	R	1				

In this example, character 1 is forced into the first available position in the work record if the input record contains an R in position 2. The data type specified is character data (comparison done on both the zone and digit portions of characters).

The following entries are used in a conditional force:

- An F in column 6 indicates a field selection specification
- An F in column 7 indicates a forced field
- A C in column 8 indicates a test for a character
- A Z in column 8 indicates a test for a zone
- A D in column 8 indicates a test for a digit.

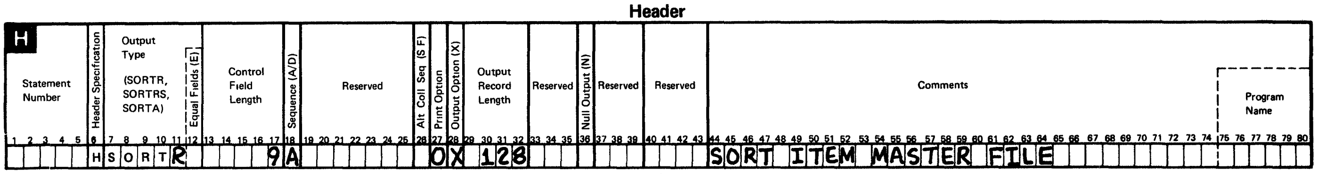
When you leave column 19 blank to force the character constant into the next available control field position, the sort program will initialize the control field position to hexadecimal FF for an ascending sort, or to hexadecimal 00 for a descending sort. Thus, if the specified record test is not successful, the character constant will not be forced into the control field, and the record will sort to the end of the output file.

In the following example, assume you want to review your item master file (as done in the example for an unconditional force). You want an output file containing grouped data. You want data about table desks written into the output file first and data about desk locks next.

To cause the table desks data to sort before the desk lock data, you could use the conditional force function of sort to test the item type field. The sort program will replace the character A (for desk locks) with the character 2, or the character B (for table desks) with the character 1.

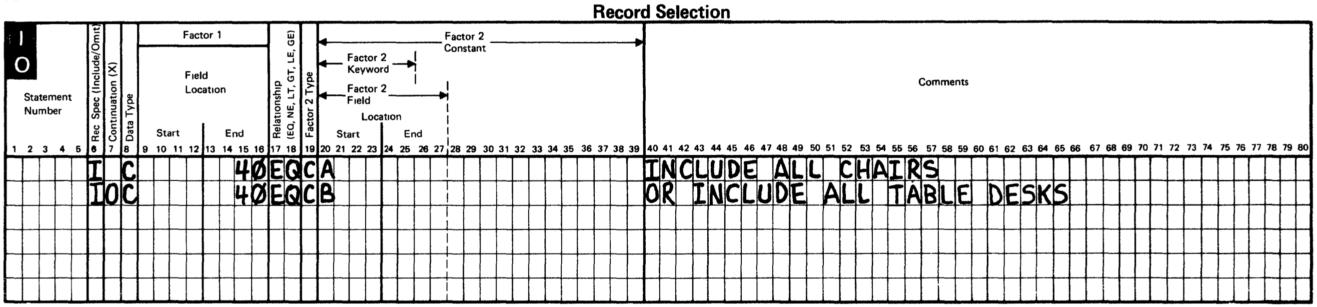
The sort specifications for this job could be defined as follows. In this example, sort processed the data for both the table desks and desk locks as one set.

Header



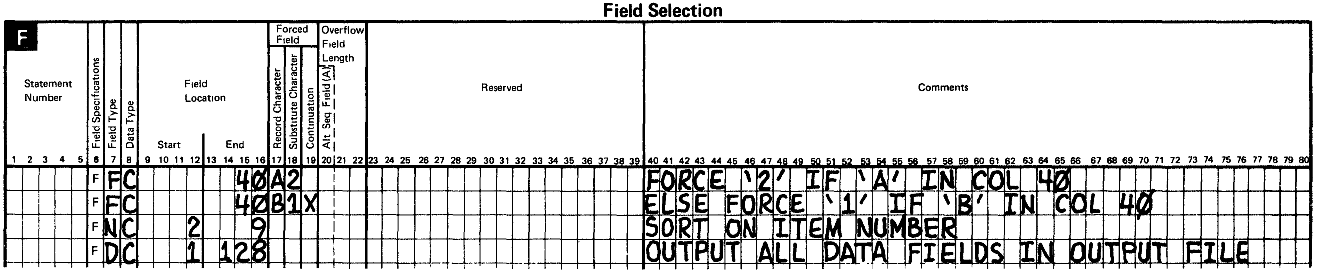
```
     HSORTR     9A        OX 128           SORT ITEM MASTER FILE
```

Record Selection



```
     I C      4ØEQCA                   INCLUDE ALL CHAIRS
     IOC      4ØEQCB                   OR INCLUDE ALL TABLE DESKS
```

Field Selection

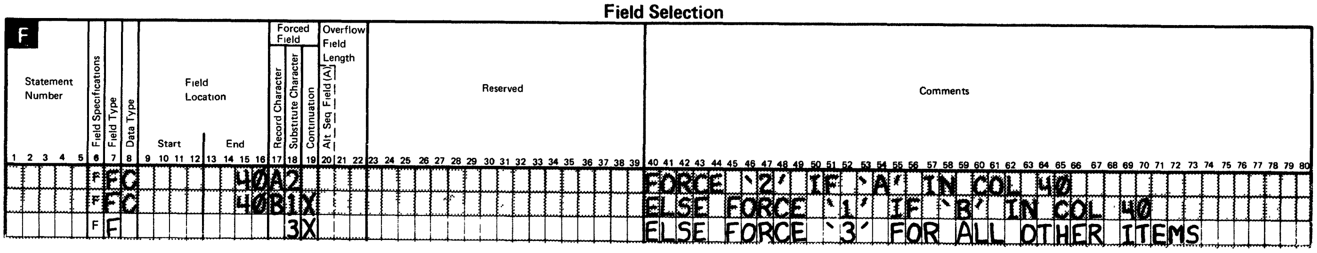


```
     FFC      4ØA2                     FORCE '2' IF 'A' IN COL 4Ø
     FFC      4ØB1X                    ELSE FORCE '1' IF 'B' IN COL 4Ø
     FNC   2   9                       SORT ON ITEM NUMBER
     FDC   1 128                       OUTPUT ALL DATA FIELDS IN OUTPUT FILE
```

Force-All

A force-all can be used after a series of conditional force statements. It allows you to specify a character constant to be forced into the control field when none of the preceding conditional force tests conditions are successfully met. An example of how you define a force-all follows:

Field Selection

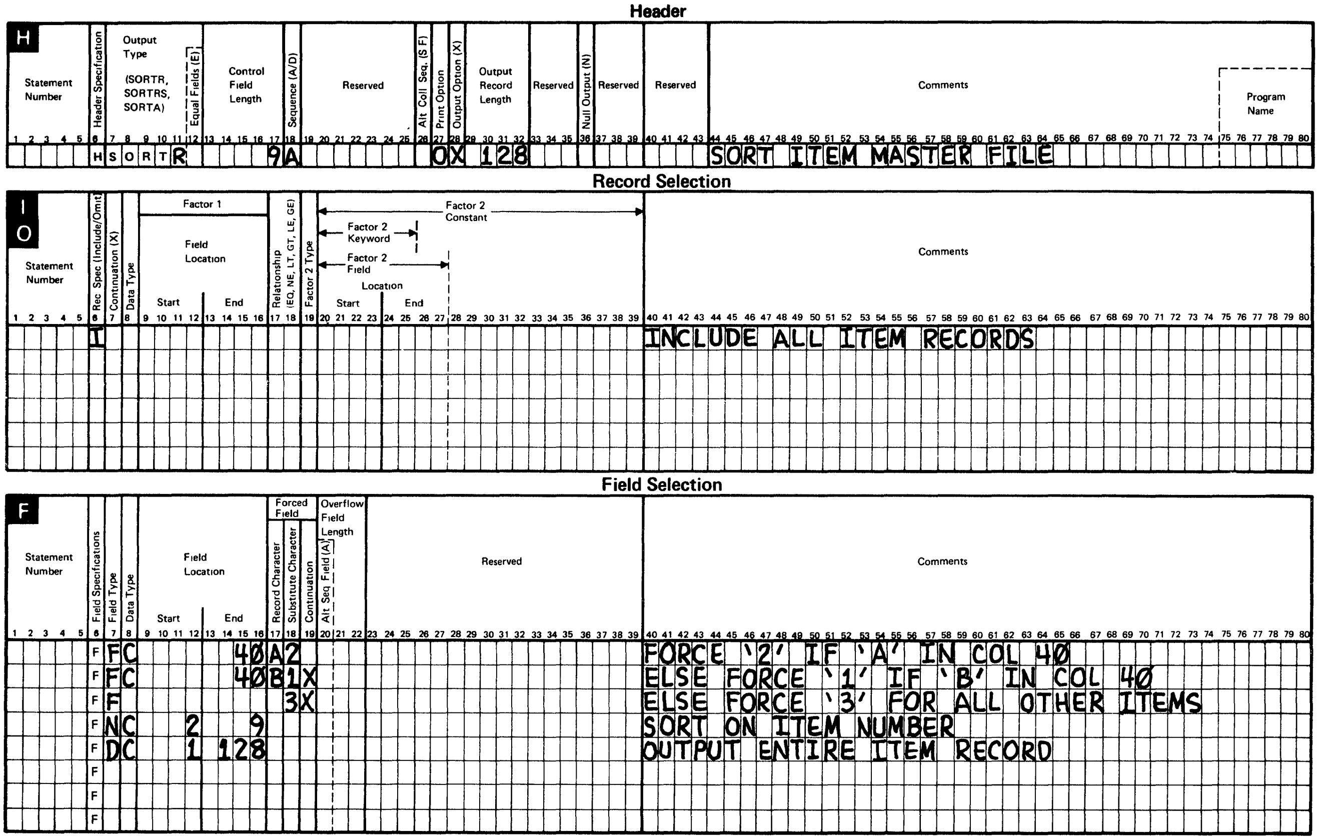


```
     FFC      4ØA2                     FORCE '2' IF 'A' IN COL 4Ø
     FFC      4ØB1X                    ELSE FORCE '1' IF 'B' IN COL 4Ø
     FF          3X                    ELSE FORCE '3' FOR ALL OTHER ITEMS
```

Assume in this example, that you want to sort the previously mentioned item master file so the output file contains data about table desks first, data about desk locks next, and data about all other items last.

As done previously, force the character 1 into the control field if the item type is a table desk and force the character 2 into the control field if the item type is a desk lock. To force all other items to be sorted into as a single group (after the table desks and desk locks), use the force-all function to force the character 3 into the control field.

Your sort specifications for this job could look as follows:



Header

H Statement Number 1-5	Header Specification 6	Output Type (SORTR, SORTRS, SORTA) 7-11	Equal Fields (E) 12	Control Field Length 13-17	Sequence (A/D) 18	Reserved 19-25	Alt Coll Seq (S F) 26	Print Option 27	Output Option (X) 28	Output Record Length 29-32	Reserved 33-35	Null Output (N) 36	Reserved 37-39	Reserved 40-43	Comments 44-74	Program Name 75-80
	H	SORTR		9	A			O	X	128					SORT ITEM MASTER FILE	

Record Selection

I O Statement Number 1-5	Rec Spec (Include/Omit) 6	Continuation (X) 7	Data Type 8	Factor 1 Field Location Start 9-12	Factor 1 Field Location End 13-16	Relationship (EQ, NE, LT, GT, LE, GE) 17-18	Factor 2 Type 19	Factor 2 Constant / Factor 2 Keyword / Factor 2 Field Location Start 20-23	End 24-27	28-39	Comments 40-80
	I										INCLUDE ALL ITEM RECORDS

Field Selection

F Statement Number 1-5	Field Specifications 6	Field Type 7	Data Type 8	Field Location Start 9-12	Field Location End 13-16	Forced Field Record Character 17	Forced Field Substitute Character 18	Continuation 19	Alt Seq Field (A) 20	Overflow Field Length 21-22	Reserved 23-39	Comments 40-80
	F	F	C		4Ø	A	2					FORCE '2' IF 'A' IN COL 4Ø
	F	F	C		4Ø	B	1	X				ELSE FORCE '1' IF 'B' IN COL 4Ø
	F	F					3	X				ELSE FORCE '3' FOR ALL OTHER ITEMS
	F	N	C	2	9							SORT ON ITEM NUMBER
	F	D	C	1	128							OUTPUT ENTIRE ITEM RECORD
	F											
	F											
	F											

At least one conditional force statement must come immediately before the force-all statement. In the preceding example, two conditional force statements come immediately before the force-all statement.

Sorting Records with Identical Control Fields

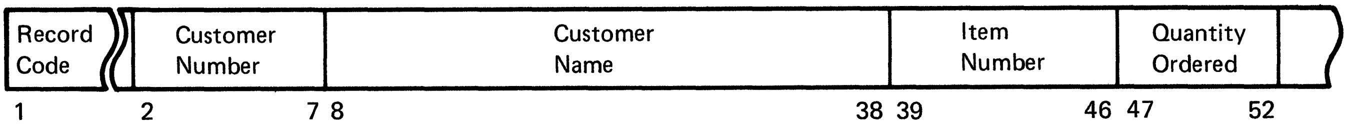
Unless you use identical control field ordering, you generally cannot predict the order in which records with identical control fields will be written into the output file. Equal control field ordering allows you to do the following:

- Specify that the input records be written into the output file in the *same* order they were read from the input file when sorting in ascending sort

- Specify that the input records be written into the output file in the *opposite* order they were read from the input file when sorting in descending order.

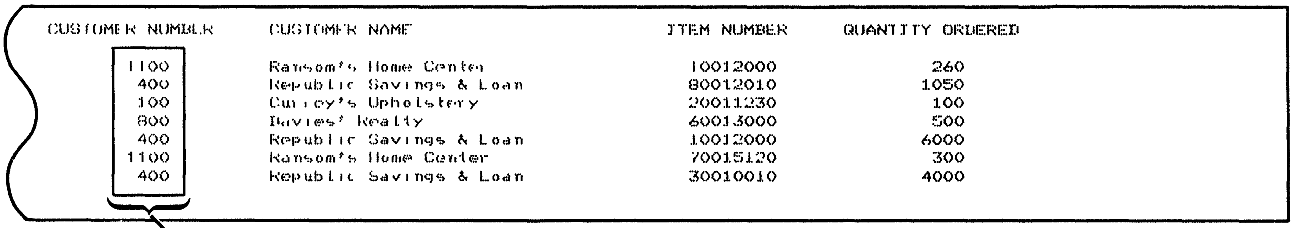
Assume you want to produce an output file containing customer orders. You want the orders (within each group of customer numbers) that were placed first by your customers to be written into the output file first, in ascending order.

The input file has the following format:



Record Code	Customer Number	Customer Name	Item Number	Quantity Ordered
1	2 – 7	8 – 38	39 – 46	47 – 52

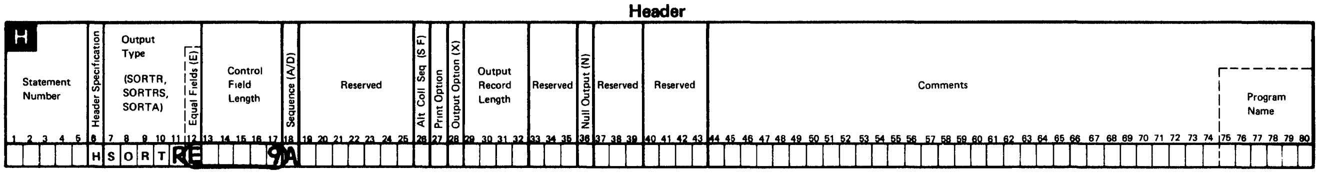
The orders for each customer were entered into the input file as they were received, as shown in the following:



```
CUSTOMER NUMBER    CUSTOMER NAME                   ITEM NUMBER    QUANTITY ORDERED

      1100         Ransom's Home Center               10012000          260
       400         Republic Savings & Loan            80012010         1050
       100         Curley's Upholstery                20011230          100
       800         Davies' Realty                     60013000          500
       400         Republic Savings & Loan            10012000         6000
      1100         Ransom's Home Center               70015120          300
       400         Republic Savings & Loan            30010010         4000
```

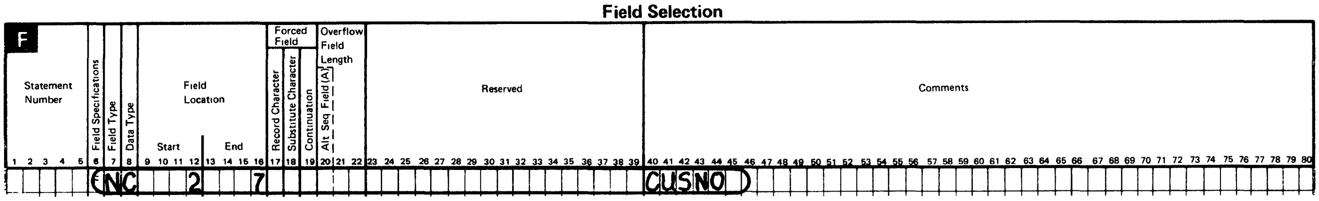
Control Field

To do this sort job, you could use customer number as the control field and define the header specification entries and the control field information as follows:



Header

Statement Number (1-5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7-11)	Equal Fields (E) (12)	Control Field Length (13-17)	Sequence (A/D) (18)	Reserved (19-25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29-32)	Reserved (33-35)	Null Output (N) (36)	Reserved (37-39)	Reserved (40-43)	Comments (44-74)	Program Name (75-80)
	H	SORTR	E	9	A											



Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Alt Seq Field (A) (20)	Overflow Field Length (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	2	7							CUSNO

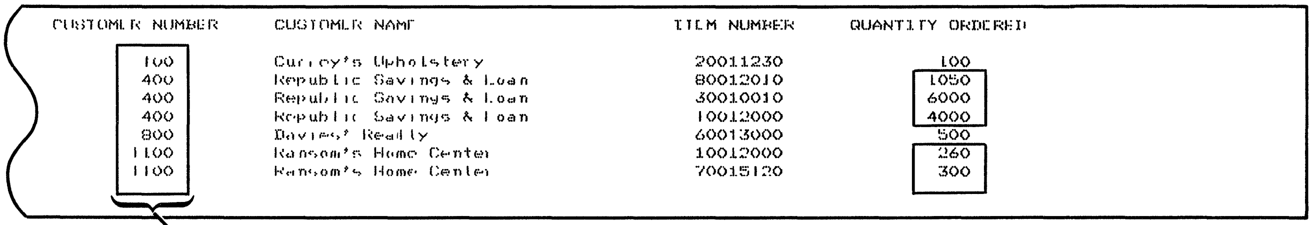
- The E in the equal control field column of the header specification specifies that records with identical control fields should be placed in an order based on how they appear in the input file.

- The A in the sequence column of the header statement specifies that the customer numbers should be placed in ascending order.

When you sort records with identical control fields (equal control field ordering), the sort program automatically places the 3-byte relative record number of each input record into the last three positions of the control field for that record type in the work record. See *Calculating Control Field Length(s)* later in this chapter for information about the effect of identical control field ordering on the lengths of control fields.

Note that the control field length in the header specification includes the length of the control field (CUSNO, which has 6 characters) plus three positions for the 3-byte relative record number of each input record.

Once the records are sorted, the output would look like the following:



```
CUSTOMER NUMBER   CUSTOMER NAME                     ITEM NUMBER   QUANTITY ORDERED

      100         Curley's Upholstery                20011230          100
      400         Republic Savings & Loan            80012010         1050
      400         Republic Savings & Loan            30010010         6000
      400         Republic Savings & Loan            10012000         4000
      800         Davies' Realty                     60013000          500
     1100         Ransom's Home Center               10012000          260
     1100         Ransom's Home Center               70015120          300
```

Ascending Order

When you specify ascending order, records with identical control fields are written into the output file in the same order they were read from the input file, as shown in the previous example.

The 1050 items ordered by customer 400 was the first order made by that customer and was the first order written into the input file. Also, customer 400 placed a second order for 6000 items and a third order for 4000 items. Had an E not been specified in column 12 of the header specification, the sequence of the output orders for each customer could not have been predicted.

Whenever you sort records with identical (or equal) control fields and you enter an E in the equal fields column of the header specification, you are using what is sometimes called **equal control field ordering**. Equal control field ordering is valid for regular sort jobs (SORTR) and record address (SORTA) jobs only.

Equal control field ordering can be used to order records based on the input order, as shown in the example. Or you can use it to change the location of data fields or to select, omit, and generate output in the same order as it is written into the input file. Control fields are only necessary if the output records are in an order different from the input records. You can select and omit records without changing their order.

When you sort records with identical control fields and you want their contents written into the output file, it is a good idea to drop the control fields (place an X in the output option column of the header specification), then redefine the fields as data fields. Otherwise, their contents will not be in the same form as the input data, but instead will be in hexadecimal form.

Calculating Control Field Length(s)

When you determine which field(s) in a record will serve as the control field(s), you must determine the size of the field(s) and specify it to the sort program.

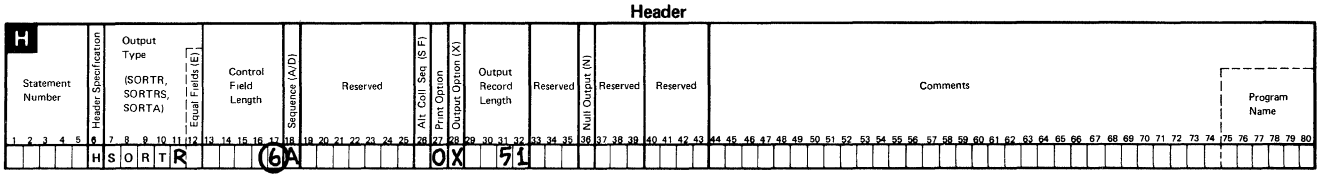
The way you calculate the length of your control fields depends on which of the following you are using:

- Single control field for a single record type
- Multiple control fields for a single record type
- Single control field for multiple record types
- Multiple control fields for multiple record types
- Forced control fields
- Records with identical control fields (equal control field ordering).

Some examples of these cases follow.

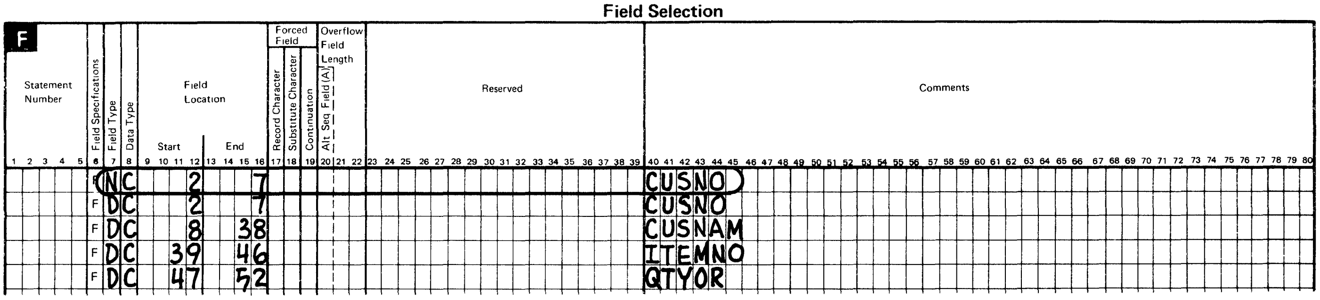
Calculating the Length of a Single Control Field for a Single Record Type

This example will calculate the length of a normal control field (used in a previous example) where customer number is used to sort the records. The sort specifications look as follows:



Header

H Statement Number (1-5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7-11)	Equal Fields (E) (12)	Control Field Length (13-17)	Sequence (A/D) (18)	Reserved (19-25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29-32)	Reserved (33-35)	Null Output (N) (36)	Reserved (37-39)	Reserved (40-43)	Comments (44-74)	Program Name (75-80)
	H	SORTR		6	A			0	X	51						



Field Selection

F Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	2	7						CUSNO
	F	D	C	2	7						CUSNO
	F	D	C	8	38						CUSNAM
	F	D	C	39	46						ITEMNO
	F	D	C	47	52						QTYOR

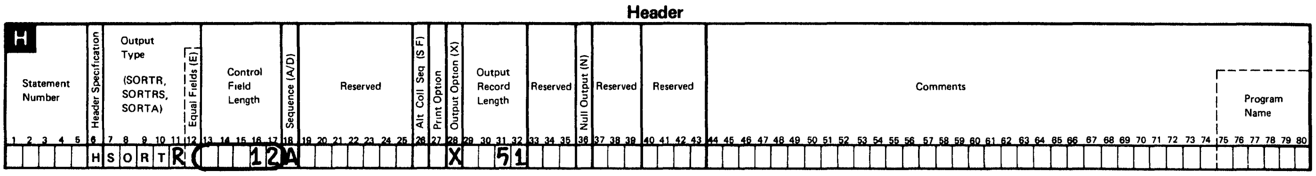
In the example, CUSNO (customer number) is the control field. To specify its length to the sort program, do the following:

1. Count the number of characters in the field. (CUSNO is in positions 2 through 7 of the input records and has 6 characters).

2. Enter the number (6 in our example) in the control field length columns of the header specification and right adjust the entry.

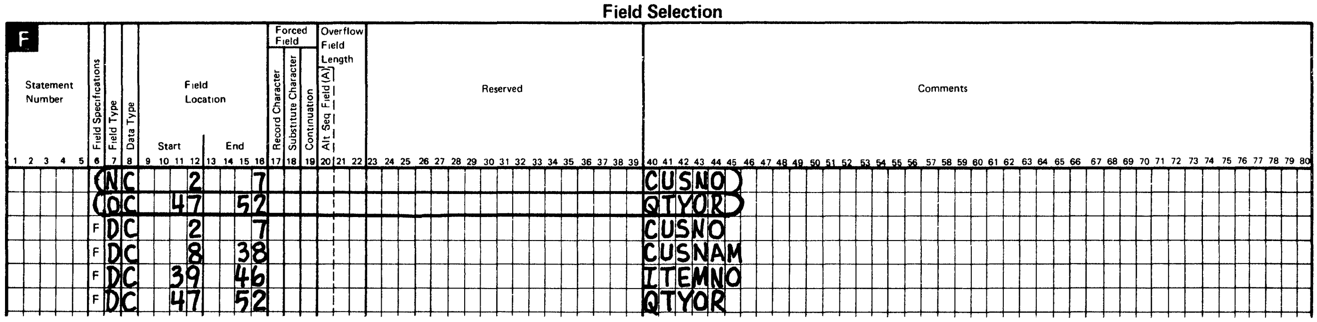
Calculating the Length of Multiple Control Fields for a Single Record Type

In the following example, the same input file is sorted using two control fields: CUSNO (customer number) and QTYOR (quantity ordered). Both control fields contain 6 characters each. Therefore, a total length of 12 is entered in the control field length columns of the header specification:



Header

Statement Number (1–5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7–11)	Equal Fields (E) (12)	Control Field Length (13–17)	Sequence (A/D) (18)	Reserved (19–25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29–32)	Reserved (33–35)	Null Output (N) (36)	Reserved (37–39)	Reserved (40–43)	Comments (44–74)	Program Name (75–80)
	H	SORTR		12	A				X	51						



Field Selection

Statement Number (1–5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9–12)	Field Location End (13–16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length Alt Seq Field (A) (20–22)	Reserved (23–39)	Comments (40–80)
		N	C	2	7						CUSNO
		O	C	47	52						QTYOR
	F	D	C	2	7						CUSNO
	F	D	C	8	38						CUSNAM
	F	D	C	39	46						ITEMNO
	F	D	C	47	52						QTYOR

Calculating the Length of a Single Control Field for Multiple Record Types

This example will use the same records used in the previous examples with the following format:

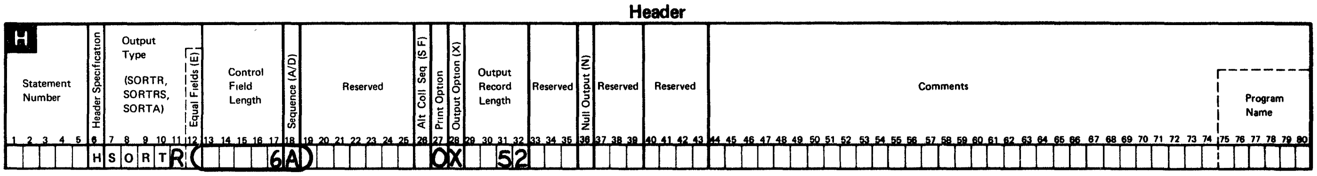
Record Code (0)	Customer Number	Customer Name	Item Number	Quantity Ordered	
1	2 7	8 38	39 46	47 52	

Also sorted are records with the following format:

Record Code (S)	Salesperson Number	Salesperson Name	Item Number	Quantity Sold	
1	2 7	8 38	39 46	47 52	

In this example, two different types of records are sorted: type O records, which contain customer orders, and type S records, which contain sales information.

Because multiple record types are being sorted, you would enter the largest total of the control field lengths (6) into the control field length columns as shown in the following:



Header

H Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
1 2 3 4 5	6	7 8 9 10 11	12	13 14 15 16 17	18	19 20 21 22 23 24 25	26	27	28	29 30 31 32	33 34 35	36	37 38 39	40 41 42 43	44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	75 76 77 78 79 80
	H	SORTR		6	A			O	X	52						

The type O records will be sorted by customer number (positions 2 through 7), and type S records will be sorted by salesperson number (positions 2 through 7).

Calculating the Length of Multiple Control Fields for Multiple Record Types

When sorting multiple record types using multiple control fields, you should define the control field as follows:

Record Types To Be Sorted:

Type O

Record Code	Customer Number	Customer Name	Item Number	Quantity Ordered
1	2 7	8 38	39 46	47 52

Type S

Record Code	Salesperson Number	Salesperson Name	Item Number	Quantity Sold
1	2 7	8 38	39 46	47 52

Control Fields For Type O Records:

Record code (1 character)

Customer number (with 6 characters)

The total control field length is 7 characters.

Control Fields For Type S Records:

Record code (1 character)

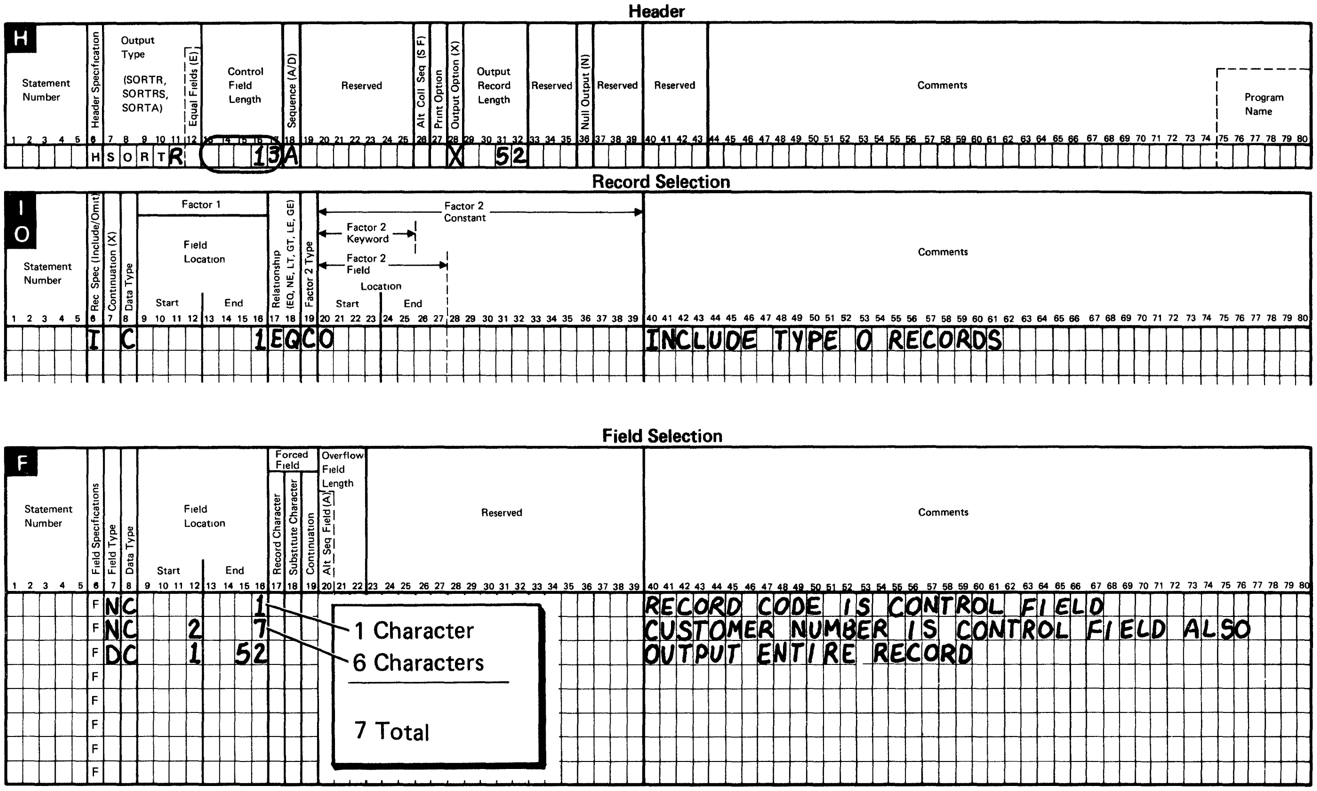
Salesperson number (with 6 characters)

Quantity sold (with 6 characters)

To calculate the proper length, add the lengths of all the control fields for each given record type:

Type 0 records: 1 + 6 = 7
Type S records: 1 + 6 + 6 = 13

Then enter the largest total (13) in the control field length columns of the header specification and right-adjust the entry as shown in the following example:



Header

```
     HSORTR    13A         X  52
```

Record Selection

```
     I C       1EQC0                   INCLUDE TYPE 0 RECORDS
```

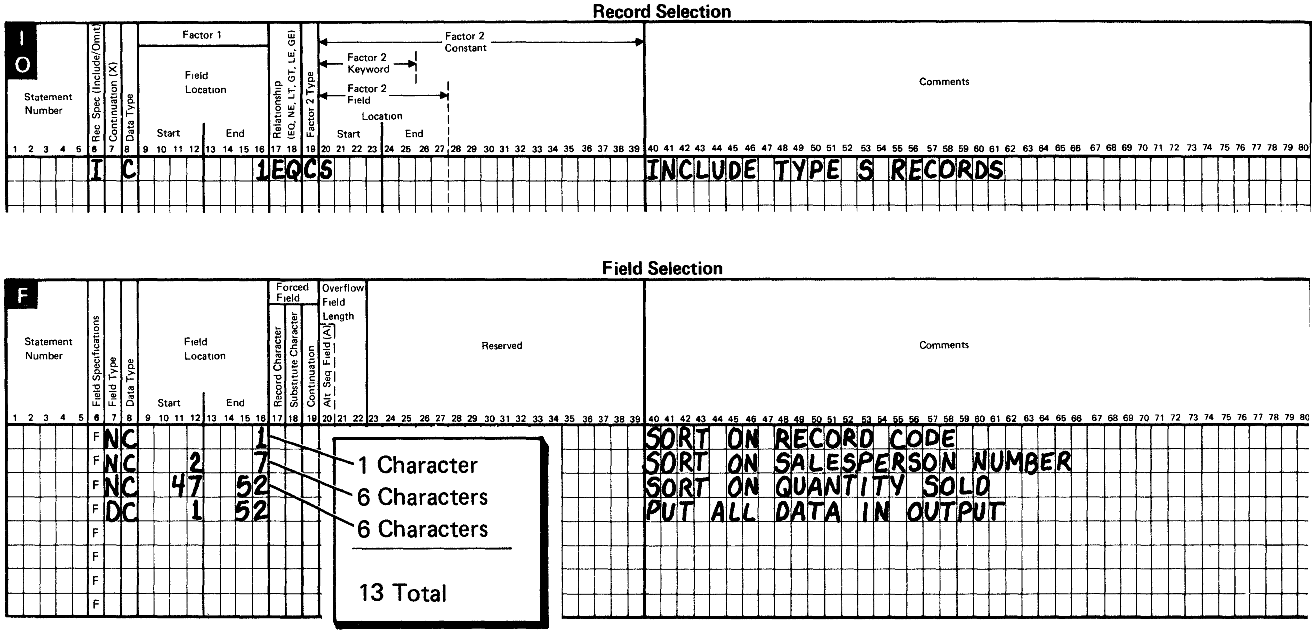
Field Selection

```
     FNC       1                       RECORD CODE IS CONTROL FIELD
     FNC   2   7                       CUSTOMER NUMBER IS CONTROL FIELD ALSO
     FDC   1  52                       OUTPUT ENTIRE RECORD
     F
     F
     F
     F
     F
```

1 Character

6 Characters

7 Total



Record Selection

```
     I C       1EQCS                   INCLUDE TYPE S RECORDS
```

Field Selection

```
     FNC       1                       SORT ON RECORD CODE
     FNC   2   7                       SORT ON SALESPERSON NUMBER
     FNC  47  52                       SORT ON QUANTITY SOLD
     FDC   1  52                       PUT ALL DATA IN OUTPUT
     F
     F
     F
     F
```

1 Character

6 Characters

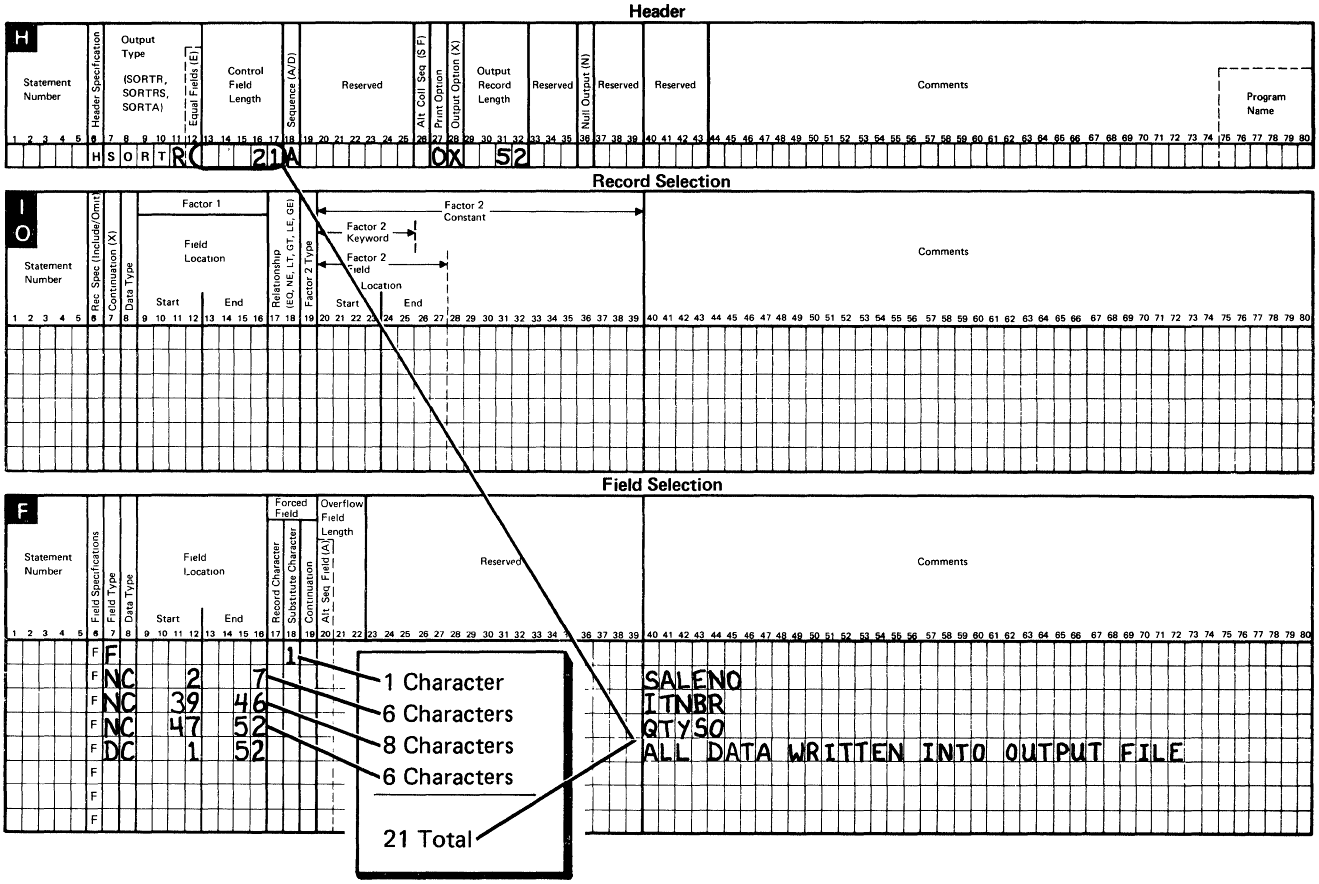
6 Characters

13 Total

Calculating the Length of Control Fields When Using Forced Control Fields

When you are sorting records and using forced control fields to insert a character into that control field, you should calculate the control field length in one of the ways previously described and add 1 to the total before entering it in the header specification. The additional character is needed to reserve space for the single character constant that is forced into the control field.

For example, if you force a character into a field (assume a 1 is being forced in front of the SALENO field), the sort specifications would look as follows:



Header

```
HSORTR   21A         0X  52
```

Record Selection

Field Selection

```
FF        1
FNC  2   7           SALENO
FNC 39  46           ITNBR
FNC 47  52           QTYSO
FDC  1  52           ALL DATA WRITTEN INTO OUTPUT FILE
F
F
F
```

1 Character
6 Characters
8 Characters
6 Characters

21 Total

Note that you should add 1 only for those forced fields that define a new control field position, and are not continuations of the previous position. Add 1 only when column 19 is blank.

Calculating the Length of Control Fields When Sorting Records with Identical Control Fields

When you sort records with identical control fields, calculate the control field length in one of the ways previously discussed for single and multiple record types and control fields. Then add 3 to the sum before you enter it in the control field length columns of the header specification. The additional 3 characters are needed to reserve space for the 3-character record address that the sort program uses in the work records.

Steps to Follow When Calculating Control Field Lengths

- For each record type that is defined, calculate the total control field length as follows:

  - Add the lengths of all normal and opposite control fields defined for the record type (N or O in the *Field Type* column of the field specifications).
    - Add 1 to the sum for each forced control field (F in the *Field Type* column and a blank in the *Continuation* column of the field specifications).
    - Add 3 to the sum if you are sorting identical control fields and you specify equal control field ordering (E in the *Equal Fields* column of the header specification).

- Enter the largest sum of all the record types in the *Control Field Length* columns of the header specification. Right-adjust the entry.

Specifying the Order of Sorted Records

The order of sorted records is defined in the sequence columns of the header specification. Usually, records are sorted according to the **standard collating sequence**. However, you can change the collating sequence to a different one by specifying an **alternative collating sequence**. See Appendix B for more information about defining an alternative collating sequence. Collating sequences are logical sequences used to put items of data into a particular order.

The sort program uses a collating sequence to compare characters in control fields to determine whether one character is equal to, greater than, or less than another character, and then sort the records in the order specified.

Standard Collating Sequence

The Standard EBCDIC Collating Sequence is the sequence used most frequently in sort jobs. Standard EBCDIC Collating Sequences are arrangements of data based on the EBCDIC character set. The standard collating sequence specifies that all characters will be arranged in the following order:

1. Blanks

2. Special characters (such as #, &, and *)

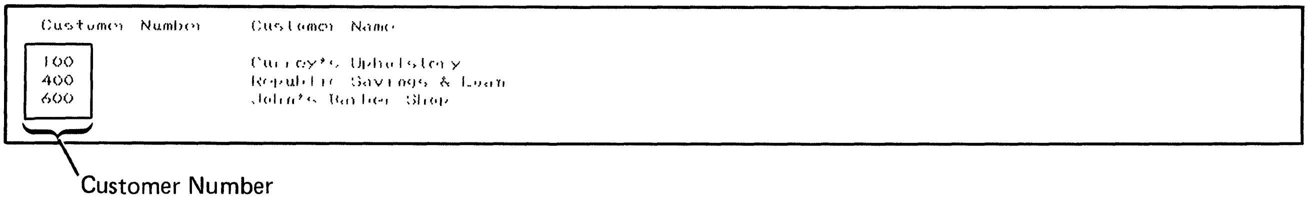
3. Alphabetic characters

   a. Lowercase characters (a)
   b. Uppercase characters (A)

4. Numeric characters.

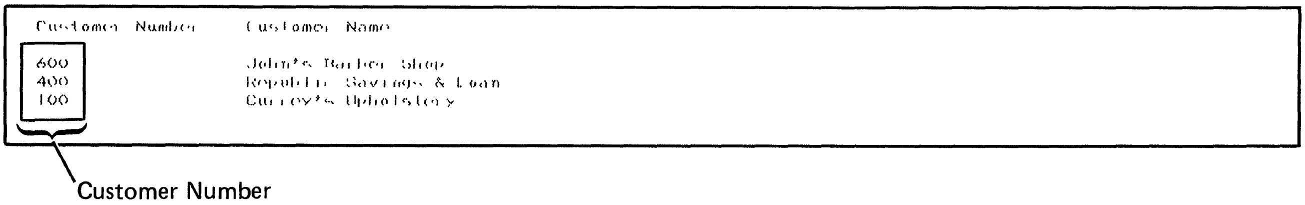
Using the standard collating sequence, you can place records in the following orders:

- **Ascending Order** with the record containing the lowest numbered item or the item with the lowest value written into the output file first. For example, records sorted in ascending order by customer number might look as follows:



```
Customer Number    Customer Name

100                Curley's Upholstery
400                Republic Savings & Loan
600                John's Barber Shop
```

- **Descending Order** with the record containing the highest numbered item or the highest value as the first record written into the output file, as shown in the following:



```
Customer Number    Customer Name

600                John's Barber Shop
400                Republic Savings & Loan
100                Curley's Upholstery
```

How The Sort Program Interprets and Compares Characters

When the sort program compares alphameric data, it looks at letters, numbers, and special characters. The sort program sees characters as EBCDIC characters composed of 8 bits that make up a byte of data.

Each 8-bit character has two parts: the four bits on the left are the **zone** portion and the four bits on the right are the **digit** portion. For example, the character A looks like this to the system:

Zone	Digit
1100	0001

If you intend to compare just the zone or the digit portion of characters, you should know which characters have identical zone or digit portions. Otherwise, different characters will compare as equals, because some characters have identical zone portions and some have identical digit portions. However, two different characters cannot have both identical zone and identical digit portions. The following chart shows some characters and their EBCDIC representations as bits of data.

	How the System/36 Sees the Character	
How We See the Character	**Zone Portion**	**Digit Portion**
*	0101	1100
1	1111	0001
2	1111	0010
3	1111	0011
K	1101	0010
?	0110	1111
P	1101	0111
Blank	0100	0000
0	1111	0000

Notice that the digit portion of a zero and a blank are identical. In most sort jobs, the entire character (that is, both the zone and digit portions) is compared. Just the zone or digit is compared in special cases.

When you define a sort job, you must specify the type of data to be sorted (in the data type columns of the record and field specifications). These entries identify which portions of the 8-bit characters will be compared during the sorting process.

For example, if you tell the sort program (by putting a D in the data type column of the record specifications) to use only the digit (D) portions of characters, characters with identical digit portions will look alike and compare as equal. Likewise, if you tell the sort program to use only the zone portion of characters (by putting a Z in the data type column of the record selection specifications), characters with identical zone portions will look alike and compare as equal. *So the data type entry is critical in ensuring that your compare operations produce the results you intend.*

Suppose, for example, you want to sort only records with a 2 in column 15 and a 2 in column 50. To include those records in your sort, you must enter a C (for character data) in the data type column of the record selection specifications. The C tells the sort program to use both the zone and digit portions of characters in its compare operations. No other character has the same zone and digit portions as a 2.

If you put a D in the data type column, hoping you would get the records with a 2 in column 15 and a 2 in column 50, you would find that the results are not what you intended. In this case, because several characters have the same digit portions as a 2, you would get records you did not want.

When you compare letters, numbers, and special characters, you must specify these types of data:

- Unsigned (alphameric) data
- Signed (numeric) data.

Unsigned Data

Unsigned data (specified by C, Z, or D) consists of alphabetic or alphameric data, including some special characters such as punctuation marks and mathematical symbols.

You can compare the following types of unsigned data:

- **Character data** (specified by C): Sort uses the entire character, both the zone and digit portions, in compare operations.
- **Zoned data** (specified by Z): Sort compares only the zone portions of characters.
- **Digit data** (specified by D): Sort compares only the digit portions of characters.

Signed Data

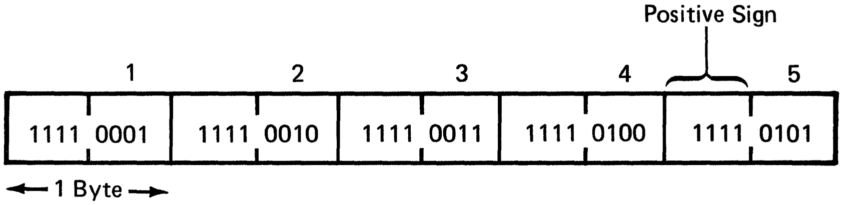
Signed data (specified by U or P) is represented by either positive or negative numbers. The ordering of signed numbers is based on both their numeric value and their sign (either plus (+) or minus (-)).

Signed numbers can be represented in the following forms:

- Unpacked
- Packed.

Unpacked decimal numbers (specified by U) are represented by 8 bits, or both their *zone and digit*. Each character in an unpacked data field contains a numeric value in the digit portion of the character. The sign of the number is contained in the zone portion of the rightmost character in the field, as shown in the following example:

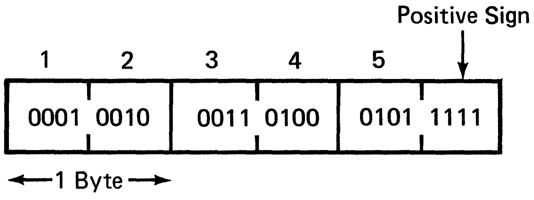
In unpacked format, the number +12345 looks like the following:



In the preceding example, each digit portion (the 4 bits on the right) of each character of data indicates one *digit* of the number. The digit portion of the first byte (0001) indicates the digit 1; the digit portion of the second character (0010) indicates the digit 2, and so on.

Packed decimal numbers (specified by P) are represented by only 4 bits, or their *digit* portions only. Packed numbers are used primarily to limit the amount of space used by the disk file. For this data type, a numeric value is contained in both the zone and digit portions of each character in a field. The sign of the number is placed in the digit portion of the rightmost character in the field.

For example, in packed format, the number +12345 looks like this:



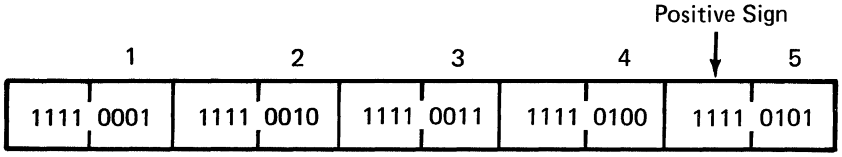
In the preceding example, each byte indicates two digits of the number. The first byte 0001 0010 indicates the first two digits (**12**) of the number (**12**345).

The sign of a number is indicated by a 4-bit binary code. Normally, the positive sign is represented by a hexadecimal F, or a binary value of 1111. Any value except a hexadecimal B or D will represent a positive sign.

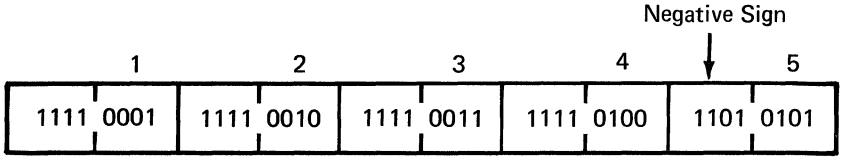
The negative sign is indicated by a hexadecimal D, or a binary value of 1101.

The signs and the numbers +12345 and -12345 look like the following:

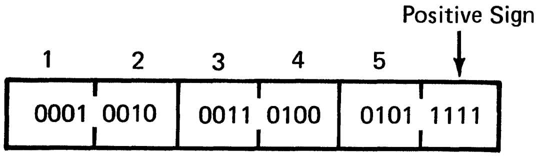
Unpacked (5 characters)



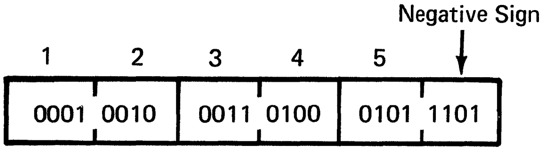
Unpacked (5 characters)



Packed (3 characters)



Packed (3 characters)



Including and Omitting Records in a Sort Job

Record selection specifications identify the records in a file to be sorted. If all the records in a file are to be sorted and they all have the same format, record selection specifications are not required.

You can use these record selection specification statements:

- Include statements to identify which records will be sorted.
- Omit statements to identify which records will not be sorted.

Using Include Statements

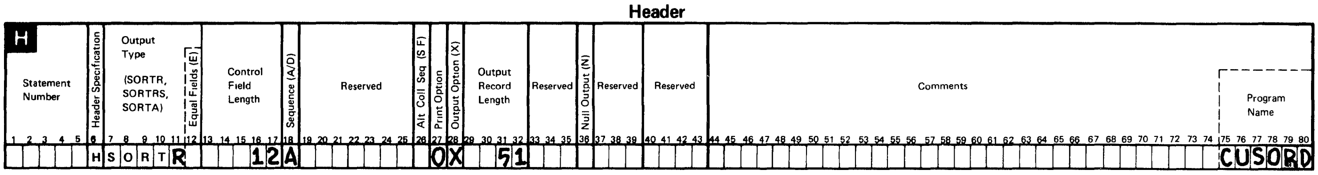
Include statements, identify which records in a file should be sorted and are especially important when you sort files with more than one record type.

If you need to use record selection statements in a sort job, you must use **include (I) statements** to describe the records you want sorted.

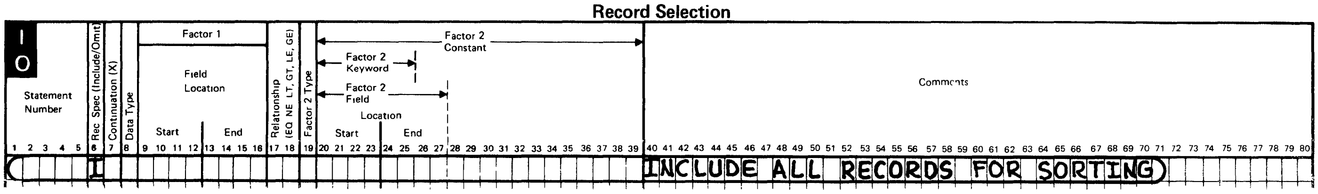
You can specify two kinds of include statements:

- Include-all statements
- Conditional include statements.

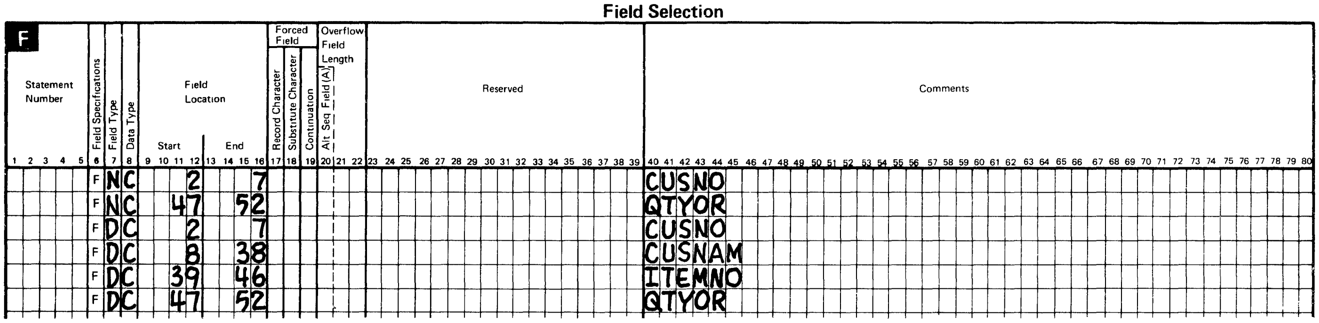
The **include-all statement** indicates to the sort program that all records that have not been described by any preceding include or omit statement will be sorted. Include-all statements are generally used when all the records in a file have the same record type. Such records must have the same field specifications. The following example shows an include-all statement with specified data fields:



```
     HSORTR    12A        0X  51                                          CUSORD
```



```
     I                                 INCLUDE ALL RECORDS FOR SORTING
```

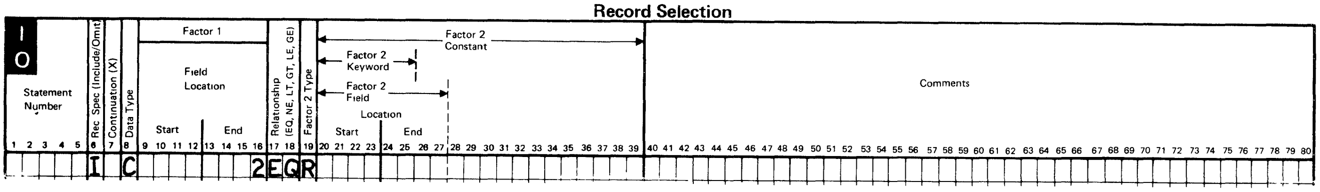


```
     FNC   2   7                       CUSNO
     FNC  47  52                       QTYOR
     FDC   2   7                       CUSNO
     FDC   8  38                       CUSNAM
     FDC  39  46                       ITEMNO
     FDC  47  52                       QTYOR
```

If you do not enter a record selection specification, the sort program assumes that all the records will be sorted (assumed include-all), in either ascending or descending order as specified in the header specification.

The placement of an include-all statement is important when you use omit statements. See *Using Omit Statements* later in this chapter for more information. An include statement must follow any omit statement in the sort specifications. Only one include-all statement can be used in each sort job. If used, it must be the last record selection statement for that job, as shown in the previous example.

The **conditional-include statement** tells sort to test data in the input records to see if it meets a condition before it is included. You must describe the fields in the record to be tested (in the record selection specifications), as shown in the following example:

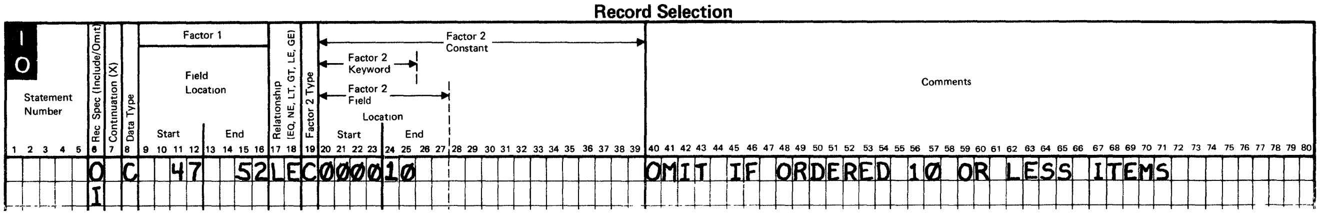


```
     I C       2EQR
```

The include statements identify one or more types of records you want to sort. If you want to sort only certain record types in a file, use an include statement with the associated record type and field statements for each type of record specified in the sort job.

Using Omit Statements

Omit (specified as O) statements identify records you do not want the sort program to sort. They are not required but can be helpful when you have many types of records to sort and just a few to exclude from the sort. Omit statements are normally followed by an include-all statement that tells the sort program to sort all of the records that are not described by the omit statements.



Record Selection

I/O Statement Number (1–5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9–12)	Factor 1 Field Location End (13–16)	Relationship (EQ, NE, LT, GT, LE, GE) (17–18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location (20–39)	Comments (40–80)
	O		C	47	52	LE	C	ØØØØ1Ø	OMIT IF ORDERED 1Ø OR LESS ITEMS
	I								

Omit statements also identify one or more types of records you do not want to sort. Often the records in any record type have at least one characteristic in common, such as all the records in record type 1 having an X in position 23. If you want to sort all but a few records in a file, use omit statements followed by an include statement and its associated record type and field statements for each type of record you want to sort.

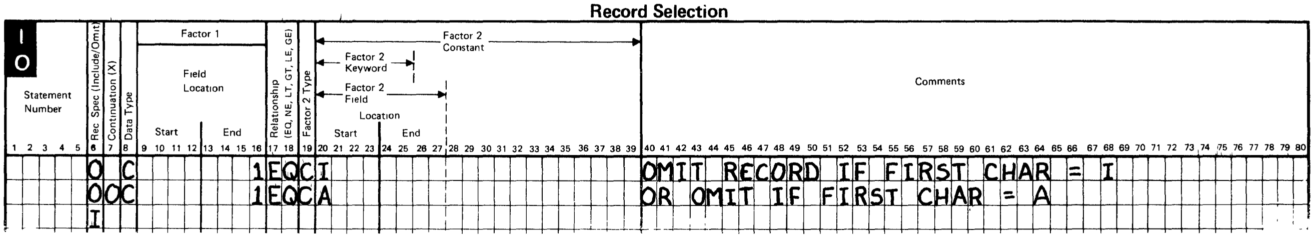
Mixing Include and Omit Statements

You can mix include and omit statements. But because the sort program processes the statements in the order they are coded, you should be particularly careful when you do this. Omit statements must be defined before include statements are defined for a record type.

Here are four helpful hints to remember when using include and omit statements:

1. End all groups of include statements with a field specification statement.
2. The last statement specified must be an include statement.
3. Every group of omit statements must be followed by an include statement.
4. Omit statements are never followed by field specifications.

The following example mixes include and omit sets:



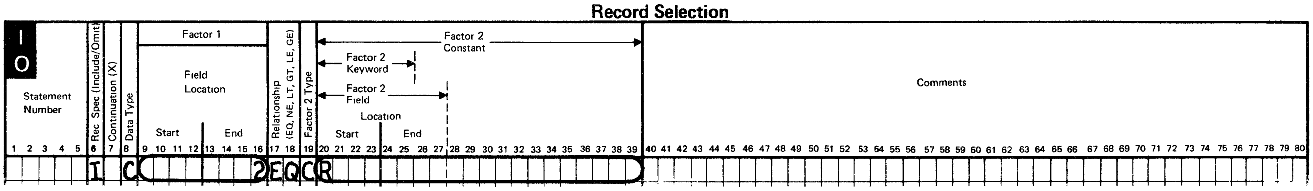
Record Selection

Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ, NE, LT, GT, LE, GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location (20-39)	Comments (40-80)
	O		C		1	EQ	C	I	OMIT RECORD IF FIRST CHAR = I
	O	O	C		1	EQ	C	A	OR OMIT IF FIRST CHAR = A
	I								

Specifying the Records to Include or Omit

You must identify the input records you want sorted to the sort program. The sort program, in turn, uses your specifications to determine which records to include or omit from the job. In order to determine which input records should be selected and/or omitted, sort compares an input field with test data to determine their relationship (that is, whether they are equal, not equal, and so on). To determine the relationship between data items, sort compares the input field located in factor 1 with the field or data specified as factor 2.

The following example shows how factors 1 and 2 are defined on the record selection specifications:


Record Selection
I
O
Statement Number
Rec. Spec (Include/Omit)
Continuation (X)
Data Type
Factor 1
Field Location
Start
End
Relationship (EQ, NE, LT, GT, LE, GE)
Factor 2 Type
Factor 2 Constant
Factor 2 Keyword
Factor 2 Field Location
Start
End
Comments
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

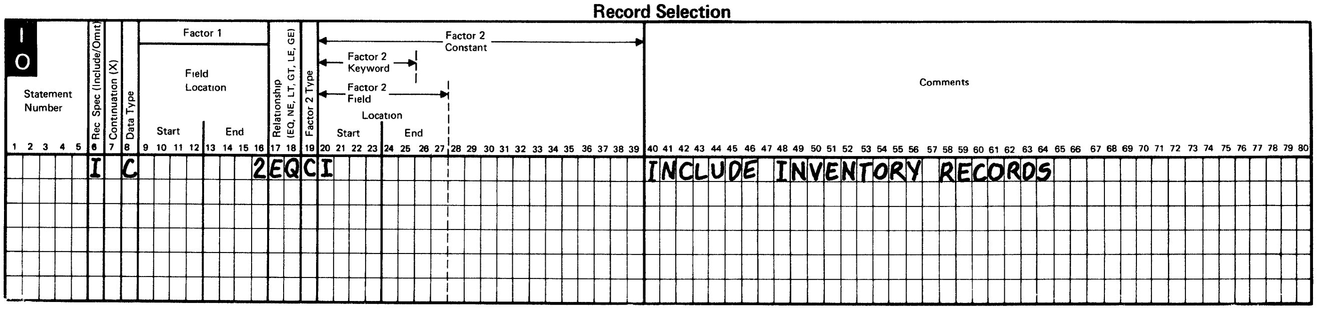
```
     I C       2EQCR
```

Factor 1

Factor 1 identifies the location of fields in the input records that sort will test to see if a condition exists. Factor 2 identifies the data against which these fields will be tested; this data can be:

- Another field in the same record
- Constants
- All or part of the program date.

For example, suppose you want to sort a file containing inventory records. The input file, however, contains not only inventory records, but also order entry records. All order entry records have an O in column 2; all inventory records have an I in column 2. To sort the file and include only those records containing an I in position 2 in the output file, you would define an include statement in the record selection specifications as follows:

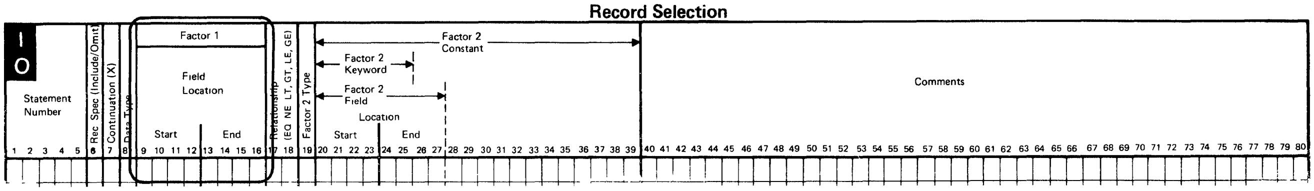

Record Selection
I
O
Statement Number
Rec. Spec (Include/Omit)
Continuation (X)
Data Type
Factor 1
Field Location
Start
End
Relationship (EQ, NE, LT, GT, LE, GE)
Factor 2 Type
Factor 2 Constant
Factor 2 Keyword
Factor 2 Field Location
Start
End
Comments
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

```
     I C       2EQCI                   INCLUDE INVENTORY RECORDS
```

The sort program compares the field in factor 1 with the data in factor 2, which is a constant in this example; if the contents of the field are equal to the character I, that record is included in the sort.

Location of Factor 1 Fields

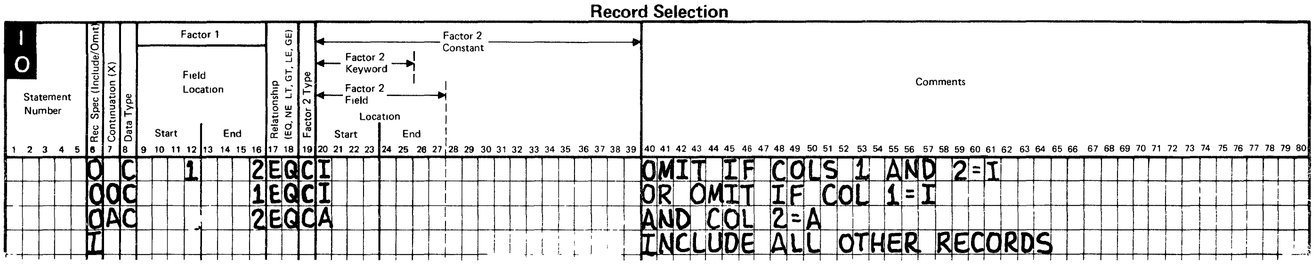
The field location columns on the record selection specifications identify the positions of factor 1 fields in the input records as shown in the following example.


Record Selection
I O
Statement Number
Rec Spec (Include/Omit)
Continuation (X)
Data Type
Factor 1
Field Location
Start
End
Relationship (EQ, NE, LT, GT, LE, GE)
Factor 2 Type
Factor 2 Constant
Factor 2 Keyword
Factor 2 Field
Location
Start
End
Comments

Both the starting and ending positions of factor 1 are identified. If factor 1 contains only one character, you need only identify the ending position.

If there is more than one factor 1 for the records you are describing, you must also identify the location of each and do the following:

- Use a separate record type line to describe each test the sort program will do.


Record Selection
I O
Statement Number
Rec Spec (Include/Omit)
Continuation (X)
Data Type
Factor 1
Field Location
Start
End
Relationship (EQ, NE, LT, GT, LE, GE)
Factor 2 Type
Factor 2 Constant
Factor 2 Keyword
Factor 2 Field
Location
Start
End
Comments

```
     O C   1   2EQCI                   OMIT IF COLS 1 AND 2=I
     OOC       1EQCI                   OR OMIT IF COL 1=I
     OAC       2EQCA                   AND COL 2=A
     I                                 INCLUDE ALL OTHER RECORDS
```

- Put an O (for OR) in column 7 of each statement that defines a different record type than that defined in any previous statement.

- Put an A (for AND) in column 7 of every statement (except the first) to tell the sort program that all of the statements apply to the same record type.

Factor 2

The sort program identifies records you want to sort by comparing the factor 1 field to **factor 2**. Factor 2 can be:

- A constant (C)
- Another field (F) in the same record
- A keyword (K) that represents all or part of the program date.

Factor 2 and factor 1 are compared to determine whether their relation is one of the following:

Relation	Meaning
EQ	Factor 1 must equal factor 2.
NE	Factor 1 must not equal factor 2.
LT	Factor 1 must be less than factor 2.
GT	Factor 1 must be greater than factor 2.
LE	Factor 1 must be less than or equal to factor 2.
GE	Factor 1 must be greater than or equal to factor 2.

If you want the sort program to compare zone portions of characters (Z in the data type column of the record selection specifications), EQ and NE are the only entries you can use.

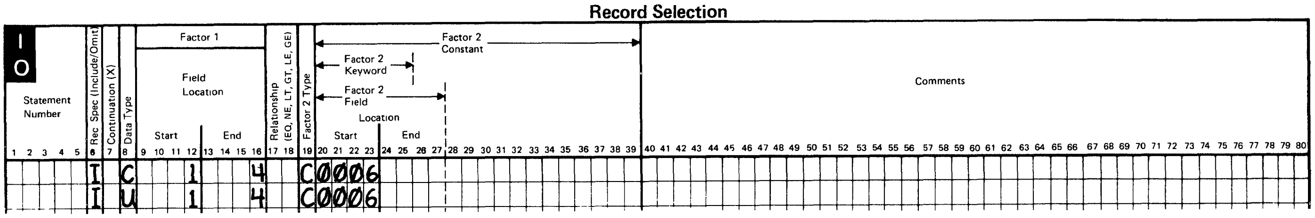
Location of Factor 2

The location of factor 2 depends on how it is used in the sort job, as discussed in the following sections.

Factor 2 as a Constant

Factor 2 can be a constant, which is fixed data. If factor 2 is a constant, it can be any arrangement of characters and blanks that are defined in columns 20 through 39, starting in column 20 of the record selection specifications. The factor 2 constant must also be the same length as the factor 1 field, except for packed constant fields that are twice as long as the input data field.

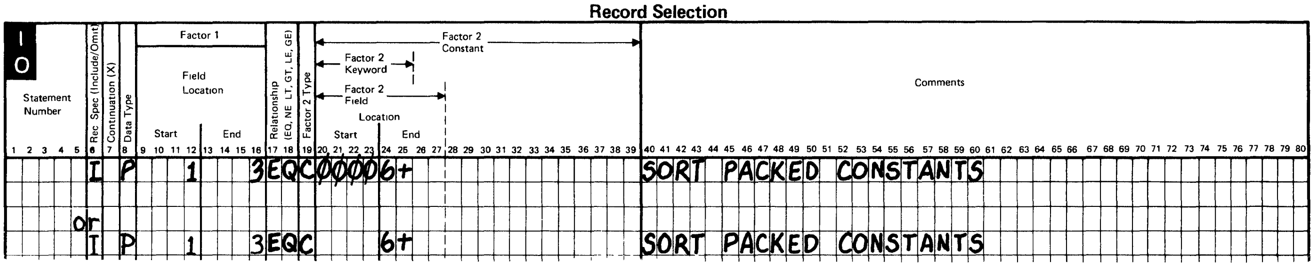
For example, if you are sorting a file and you have a 4-digit number in the factor 1 field, the constant must take up four positions. If the constant is the number 6, put the 6 in column 23 and either leave columns 20, 21, and 22 blank or fill them with zeros.



Record Selection

Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ, NE, LT, GT, LE, GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant (20-39)	Comments (40-80)
	I		C	1	4		C	0006	
	I		U	1	4		C	0006	

If the factor 1 field contains a packed number, the length of the constant (including the sign) must be twice the length of the factor 1 field.



Record Selection

Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ, NE, LT, GT, LE, GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant (20-39)	Comments (40-80)
	I		P	1	3	EQ	C	00006+	SORT PACKED CONSTANTS
or									
	I		P	1	3	EQ	C	6+	SORT PACKED CONSTANTS

A factor 2 constant can be of two types:

- Unsigned constants (character, zone, and digit data types)
- Signed constants (unpacked and packed data types).

Factor 2 as an Unsigned Alphameric Constant

When factor 2 is an **alphameric constant**, you should indicate which portion of a character will be used in compare operations. Alphameric constants are unchanging values in your records. They are the actual data to be processed, not the name of a field containing the data.

You can specify that:

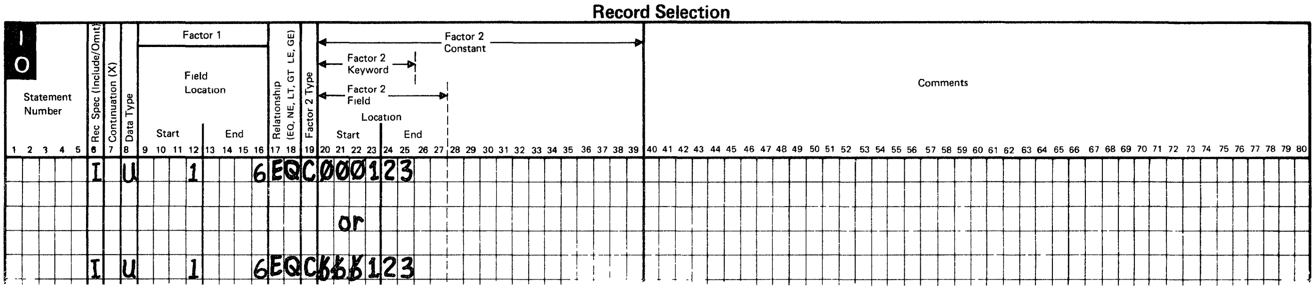
- The entire character (C) (that is both the zone and digit portions of characters) will be compared
- The zone (Z) portion only will be compared
- The digit (D) portion only will be compared.

Factor 2 as a Signed Numeric Constant

Numeric constants must be right-adjusted within the field length specified in factor 1 (within twice the field length if factor 1 is a packed number). Numeric constants are unchanging numbers in your records.

Zoned Decimal Fields: Constants can be specified as zoned decimal fields. Both the zone and digit portions of each character in zoned decimal fields are used in compare operations.

The following is an example of how to define zoned decimal fields (sometimes called unpacked fields) in a sort job. Assume that factor 1 defines a 6-position field in the input record, and that factor 2 is the numeric constant 123. To right-adjust the constant within six positions, you must put the constant in columns 23, 24, and 25. Leading zeros are not required. Because blanks and zeros in a numeric field look the same to the sort program, columns 20 through 25 could contain either 000123 or bbb123 (with b representing a blank).



Record Selection

I O Statement Number (1–5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9–12)	Factor 1 Field Location End (13–16)	Relationship (EQ, NE, LT, GT, LE, GE) (17–18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location Start (20–23)	End (24–27)	(28–39)	Comments (40–80)
	I		U	1	6	EQ	C	ØØØ1	23		
								or			
	I		U	1	6	EQ	C	ƀƀƀ1	23		

If factor 1 is a zoned decimal number with a sign and the constant is a negative number, the last character in the constant must indicate both the numeric value of the last digit and the negative sign for the entire constant. For binary numbers, the sign is in the rightmost 4 bits of the last character.

The following chart shows how you would define some negative zoned constants.

Defining Negative Zoned Constants

		How the Number Looks Inside the Computer	
If Last Digit in Constant is	Character That You Code	Zone Portion[1]	Digit Portion[2]
0	- (minus code)	0110	0000
1	J	1101	0001
2	K	1101	0010
3	L	1101	0011
4	M	1101	0100
5	N	1101	0101
6	O	1101	0110
7	P	1101	0111
8	Q	1101	1000
9	R	1101	1001

[1] The zone portion indicates the negative sign of the entire number.

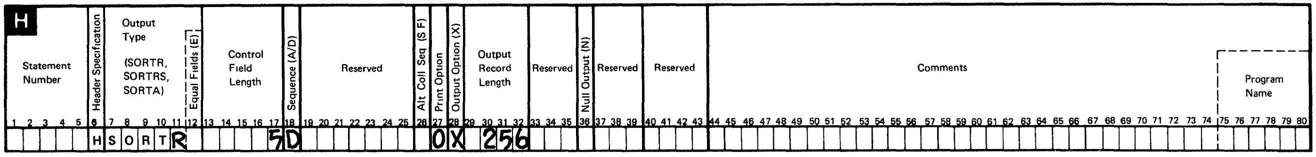
[2] The digit portion indicates the numeric value of the last digit in the number.

Packed Fields: If factor 1 is a packed number, the last character in the constant must be its sign (either +, blank, or -). If a sign is not entered, Sort will default to a plus (+) sign.

Do not let a plus (+) sign be the last nonblank character in a sort specification statement contained in a procedure. If necessary, add a comment after the plus (+) sign. Otherwise, the program will assume the next line is a continuation of the line with the plus (+) sign, and treat it that way.

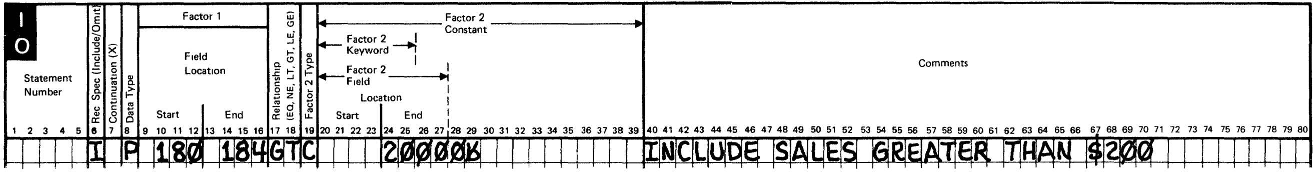
The following example shows a comparison between a packed field and a constant field. Customer sales are sorted in descending order by the number of sales last year. Only the records of customers with sales greater than $200.00 are included in the sorted output.

Header



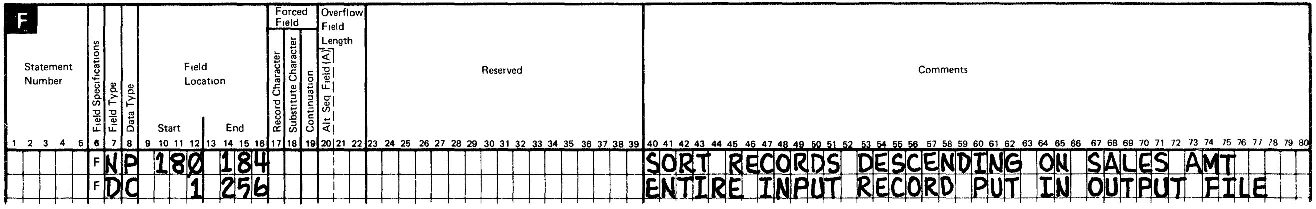
```
     HSORTR     5D        0X 256
```

Record Selection



```
     I P 18Ø 184GTC    2ØØØØb          INCLUDE SALES GREATER THAN $2ØØ
```

Field Selection

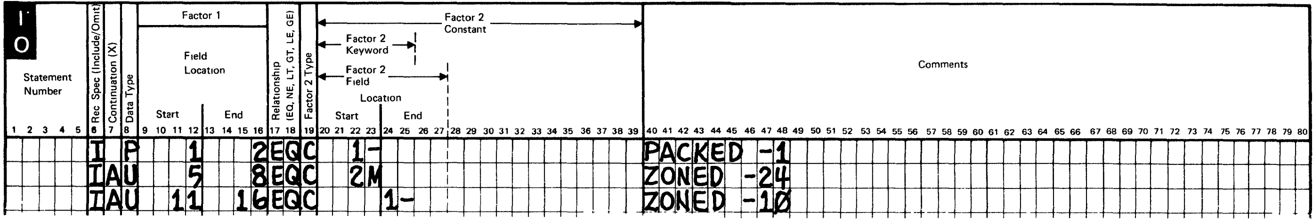


```
     FNP 18Ø 184                       SORT RECORDS DESCENDING ON SALES AMT
     FDC   1 256                       ENTIRE INPUT RECORD PUT IN OUTPUT FILE
```

The constant and its sign that are being compared, are twice as long as the packed input field. In the example, the constant is right-adjusted. The blank in column 29 represents a positive sign, optionally a plus sign (+) could be used.

For example, here are the specifications you would define for sort records with a packed negative 1 (-1) in positions 1 and 2, a zoned negative 24 (-24) in positions 5 through 8, and a zoned negative 10 (-10) in positions 11 through 16.

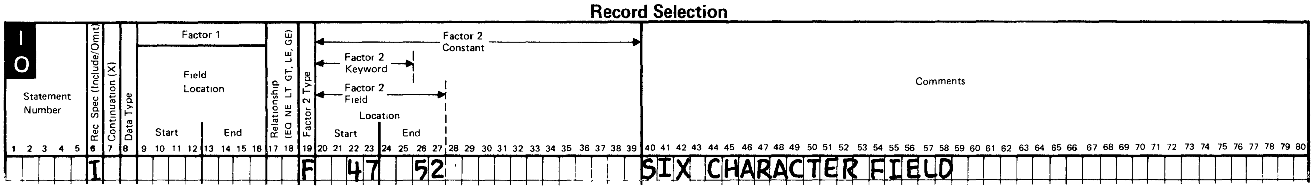
Record Selection



```
     I P   1   2EQC  1-                PACKED -1
     IAU   5   8EQC  2M                ZONED -24
     IAU  11  16EQC    1-              ZONED -1Ø
```

Factor 2 as Another Field

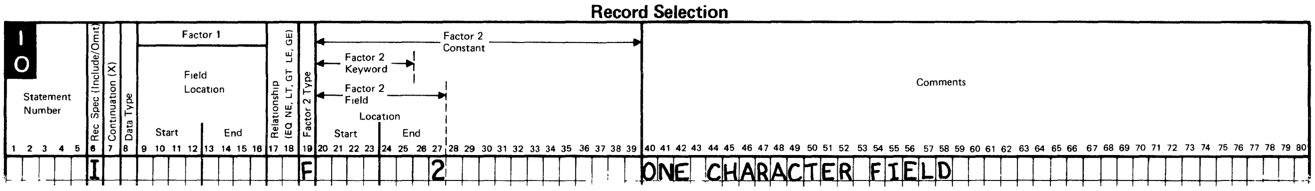
Factor 2 can be another field in the same record as the factor 1 field. When factor 2 is another field, you must specify its starting and ending locations in the field location columns of the record selection specifications.



Record Selection

I O Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ NE LT GT, LE, GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location Start (20-23)	Factor 2 Field Location End (24-27)	(28-39)	Comments (40-80)
	I						F	47	52		SIX CHARACTER FIELD

You can describe fields that are only 1 character long in the End columns. This entry must be right-adjusted.



Record Selection

I O Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ, NE, LT, GT, LE, GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location Start (20-23)	Factor 2 Field Location End (24-27)	(28-39)	Comments (40-80)
	I						F		2		ONE CHARACTER FIELD

Factor 2 as a Keyword

When factor 2 is a keyword, the sort program will compare all or part of the program date with the factor 1 field. You can use the following keywords:

- UDATE
- UMONTH
- UDAY
- UYEAR.

When factor 2 is a keyword, you must indicate that both the zone and digit portions of characters will be compared to the factor 1 field (enter a C, character data, on the record selection specifications).

Factor 2 keyword locations must also be identified to the sort program using the keyword location columns on the record selection specifications. You should always start the keyword in column 20.

Keyword Length: The factor 2 keyword can be a maximum of 6 characters long.

The following chart lists the keywords, the part of the program date the keyword permits sort to compare with the factor 1 field, and the allowable length of the factor 1 field.

Keyword	Part of Program Date	Factor 1 Field Length
UDATE	Entire program date	6 characters
UMONTH	Month portion of program date	2 characters
UDAY	Day portion of program date	2 characters
UYEAR	Year portion of program date	2 characters

The factor 1 field length must be 6 for UDATE, and 2 for UMONTH, UDAY, or UYEAR.

If the UDATE keyword is used, the program date must be in the same format as the date contained in the input records. (See the DATE procedure in the *System Reference* manual for information about program date formats.)

The date is specified either in the // DATE OCL statement or by the SET date command. For more information about how to specify the date in the // DATE OCL statement or how to use the SET command, see the *System Reference* manual.

If factor 2 is UDATE, record selection on or before, or on or after, a certain date (the relationship columns of the record selection specifications contain LT, GT, LE, or GE) works only with the international date format (yymmdd). If the program date and the input records date are not in the international date format (yymmdd), the keywords UYEAR, UMONTH, and UDAY, must be used to select the records.

Specifying Data Fields in a Sort Job

To include any data in the output file, you must identify (on the field specifications) which fields in the input records will be written into the output records. Data fields are not involved in the sorting process and are not changed by the sort program. In a sort job, fields are written into the output file as **data fields** in the following ways.

- You can specify the field as a control field (in the field specifications) and specify that sort keep the control fields (by leaving the output option column of the header specification blank).

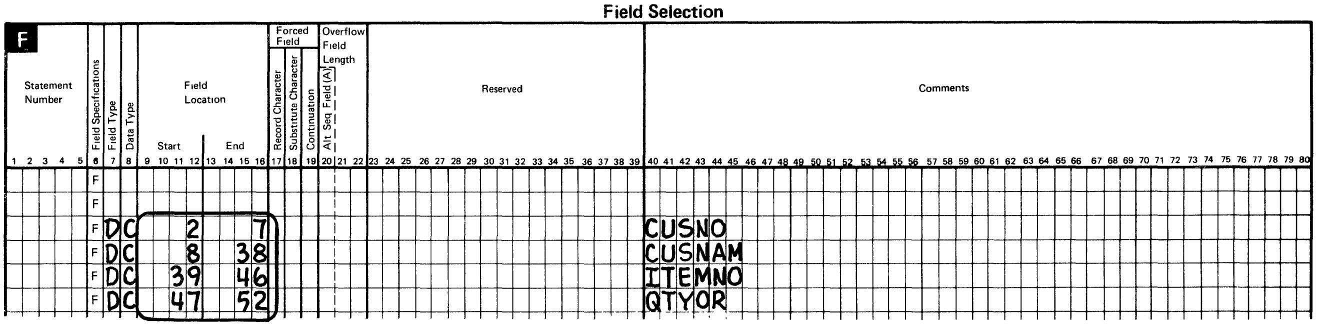
- If you use normal control fields with packed or digit data, opposite control fields, or an alternative collating sequence, and you want the control fields written into the output file, you must describe the fields twice: once as control fields and again as data fields. Then specify that sort drop the control fields (by placing an X in the output option column of the header specification). Otherwise, their contents will be changed to a form other than the original input.

- If you use equal control field ordering and you want the control field written into the output file, you must describe the field twice also: once as a control field and again as a data field. You should also specify that sort drop this control field. Otherwise, its contents will include the 3-byte relative record number sort uses to order the records. (See *Dropping and Keeping Control Field Data* earlier in this chapter.)

Location of Data Fields

When you specify data fields, you must indicate to the sort program where to find them in the input record.

The field location columns on the field specifications identify the positions of fields in records in the input files. Both the starting position and the ending position of a field are identified as shown in the following exa̍mple:



Field Selection

F Statement Number 1 2 3 4 5	Field Specifications 6	Field Type 7	Data Type 8	Field Location Start 9 10 11 12	Field Location End 13 14 15 16	Forced Field Record Character 17	Forced Field Substitute Character 18	Continuation 19	Overflow Field Length Alt Seq Field (A) 20 21 22	Reserved 23–39	Comments 40–80
	F										
	F										
	F	D	C	2	7						CUSNO
	F	D	C	8	38						CUSNAM
	F	D	C	39	46						ITEMNO
	F	D	C	47	52						QTYOR

For 1-character fields, you need only specify the *End* position.

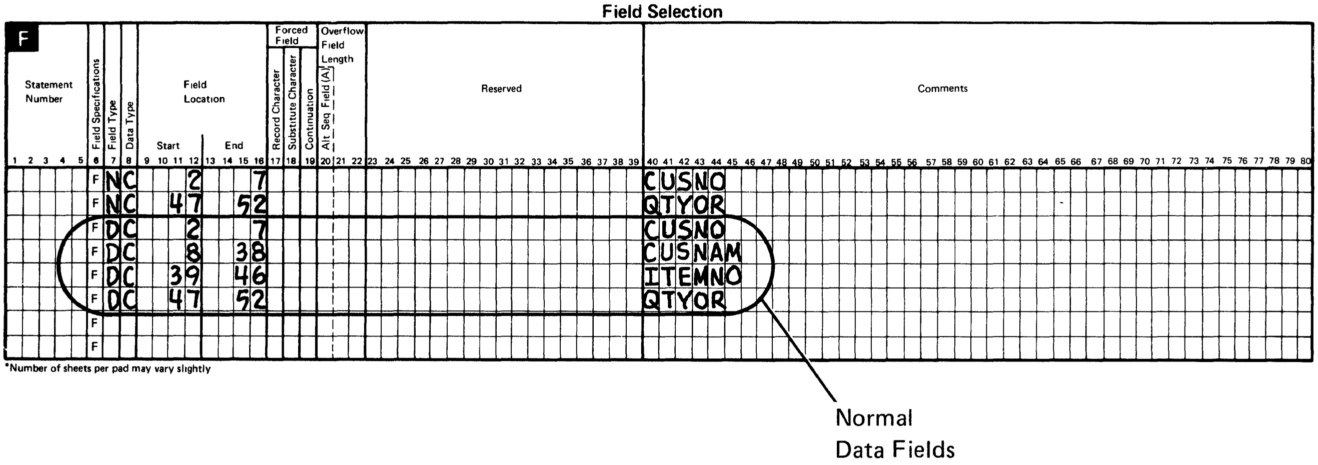
The sort program will go to the specified location in the input record and move that field to the specified location in the output record. The output location is determined by the order in which the fields are defined on the field specifications.

You can specify three kinds of data fields:

- Normal data fields
- Summary data fields (fields containing data that is to be added together)
- Forced data fields.

Normal Data Fields

The following example shows how to define a normal data field for a sort job.



Field Selection

F

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Record Character	Substitute Character	Continuation	Alt Seq Field (A)	Reserved	Comments
	F	N	C	2	7						CUSNO
	F	N	C	47	52						QTYOR
	F	D	C	2	7						CUSNO
	F	D	C	8	38						CUSNAM
	F	D	C	39	46						ITEMNO
	F	D	C	47	52						QTYOR
	F										
	F										

*Number of sheets per pad may vary slightly

Normal
Data Fields

Normal data fields apply to regular (SORTR) and summary (SORTRS) sort jobs only. They are fields you want the sort program to include in the sorted records, but that you do not want sort to use in sorting the records. The order in which you define the fields on the field specifications is the order in which the data is written into the output records. Data fields are always specified after control fields, as shown in the previous example.

If data fields are specified for a SORTA job, these specification statements will simply be ignored.

When your file has more than one type of record:

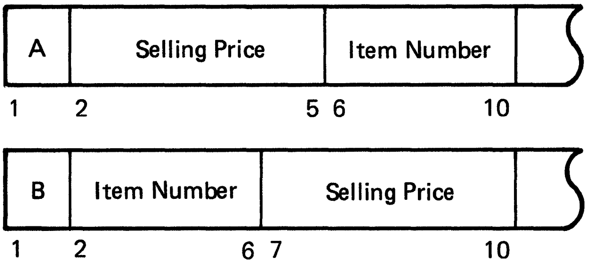
- The number of data fields need not be the same for all the record types.
- The total lengths of all the data fields need not be the same for all record types. Sort places blanks to the right of shorter records so that all total record lengths are equal.

Summary Data Fields

A **summary data field** is a field that is designated to hold selected accumulated totals. These totals can only be specified in summary (SORTRS) sort jobs.

In summary sort jobs, the output positions that are defined as a summary data field will be added for all input records. Therefore, if you have multiple input record types you must ensure that the summary output data for each record type is defined to match any summary data fields defined in the other record types.

For example, assume you want to sort a file with two different record types with formats, like the following:



Assume you want to sort this file by item number and add the selling price of each item. To do this, you could define the sort specifications as follows:

Header

H Statement Number (1-5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7-11)	Equal Fields (E) (12)	Control Field Length (13-17)	Sequence (A/D) (18)	Reserved (19-25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29-32)	Reserved (33-35)	Null Output (N) (36)	Reserved (37-39)	Reserved (40-43)	Comments (44-74)	Program Name (75-80)
	H	SORTRS		5	A			0		9					SUMMARY SORT	

Record Selection

I O Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ NE LT, GT, LE GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location Start (20-23)	Factor 2 Field Location End (24-27)	Factor 2 Constant (28-39)	Comments (40-80)
	I		C		1	EQ	C	A			INCLUDE 'A' TYPE RECORDS

Field Selection

F Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	6	10						SORT ON ITEM NUMBER
	F	S	U	2	5						ADD THE AMOUNT FOR EACH ITEM

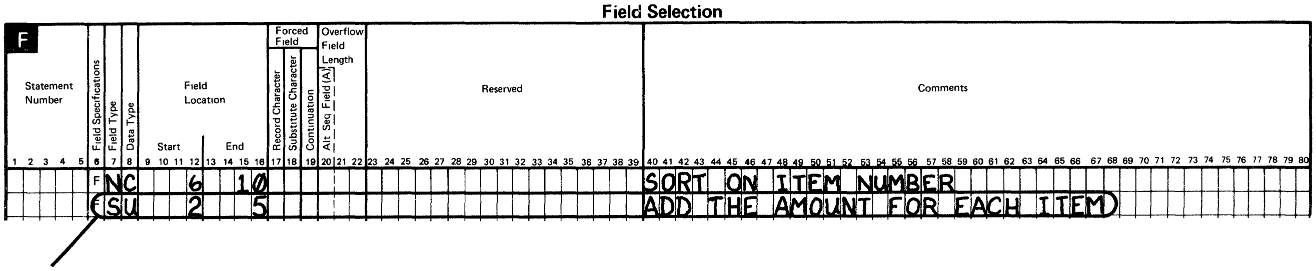
Record Selection

I O Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ NE LT GT LE GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location Start (20-23)	Factor 2 Field Location End (24-27)	Factor 2 Constant (28-39)	Comments (40-80)
	I		C		1	EQ	C	B			INCLUDE 'B' TYPE RECORDS

Field Selection

F Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	2	6						SORT ON ITEM NUMBER
	F	S	U	7	10						OUTPUT EQUALS TOTAL FOR ALL ITEMS

Because the input file has two different record types, you must describe the job and define the output in two separate include sets. In the field specifications for set 1, a summary data field was defined after the control field.



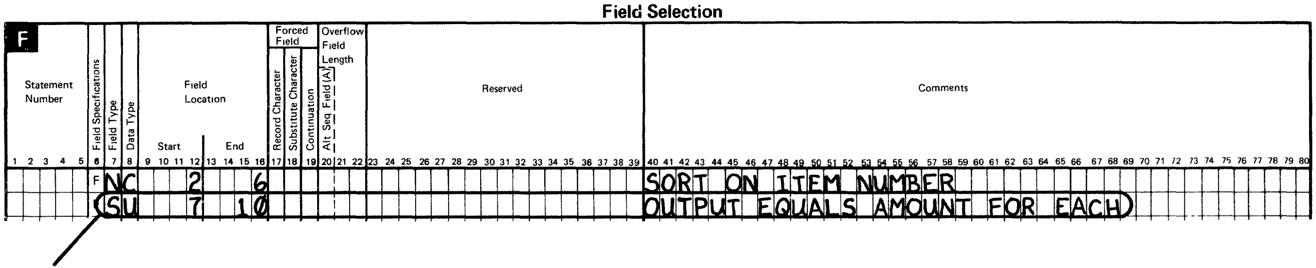
Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Alt Seq Field (A) (20) / Overflow Field Length (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	6	10						SORT ON ITEM NUMBER
	F	S	U	2	5						ADD THE AMOUNT FOR EACH ITEM

Summary Data Field

In this example, the data in positions 2 through 5 (selling price) will be added for all type A records.

To prevent the sort program from adding the data in the same positions in type B records (which would be item number), you should define another set of specifications, Set 2. The field specification statements define the second summary data field as being in positions 7 through 10 of the type B input records.



Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Alt Seq Field (A) (20) / Overflow Field Length (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	2	6						SORT ON ITEM NUMBER
	F	S	U	7	10						OUTPUT EQUALS AMOUNT FOR EACH

Summary Data Field

Defining a Summary Data Field

As you see from the preceding example, summary data fields are defined very much like normal output data fields:

- Column 6 must be an F to indicate a field selection specification.

- Column 7 must be an S to indicate a summary sort field.

- Column 8 defines the data type of the field to be added.

  The same data types that are used for normal data fields ar also used for summary data fields. This data type entry is critical because the method of adding the data varies depending on its type:

  - For P (packed) data type, the adding will include both the zone and digit portions of the input data, plus a possible sign.
  - For U (unpacked) data type, the adding will include only the digit portions of the input data, plus a possible sign.
  - For D (digit) only data type, the adding will include only the digit portions of the input data without a sign involved.
  - For C (character) data, the adding will include the entire character without a sign involved.
  - For Z (zone) data type, the adding will include only the zone portion of the input data without a sign involved.
  - Columns 9 through 16 must indicate the starting and ending positions of the summary data field in the input record. Like normal data fields, the starting position need not be specified for a field that is only one position long.

A maximum of 24 different output fields can be summarized in a SORTRS job.

When you define a summary sort job, you can also define normal data fields in the job. If you do, however, you cannot guarantee which input record data will become the output for each unique summarized control field value. The output record for several input records with identical control fields will contain added data for all input records for those fields defined as summary fields. For the fields defined as normal data fields, the output will be the data from the first output record processed by sort.

Summary Overflow

When you add data during a summary sort, the length of the field that will hold the total is normally the same length as the input data. The previous *Summary Data Fields* example added data in a field (selling price). The field is four characters long in the input record and the total in the output record is also four characters long.

It is possible for the selling price totals, for a given item number, to exceed the four positions allowed for the output total. If an amount exceeds the four positions allowed, only the 4 low-order positions of the total would be placed in the output file. Any higher order digits would be lost with no indication of the loss.

To prevent this situation from occurring, or to indicate that an overflow has occurred, you can do the following:

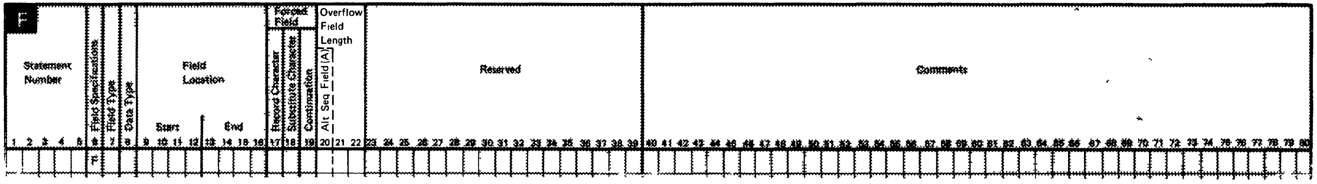
- You can specify a larger output area to hold the totals for a given summary data field.

- You can specify that the sort program place a character constant in the output record to warn you of a possible overflow.

Eliminating Summary Data Overflow

When you define a summary data field, you may specify the length of the overflow field in the *overflow field length* columns of the field specifications.

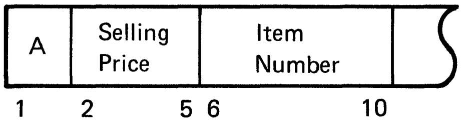
Field Selection



An entry in these columns defines the amount of main storage the sort program should reserve for any summary data totals. The length of this field cannot be longer than the maximum length allowed for the input data type specified in column 8.

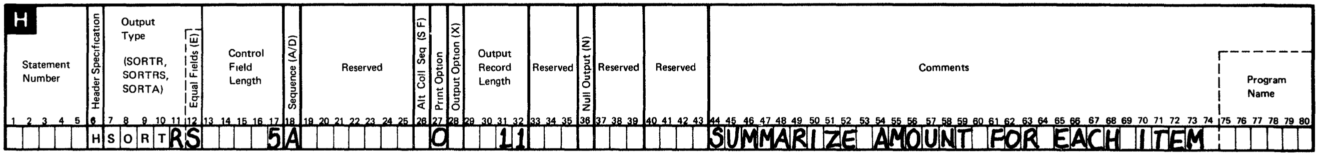
Data Types	**Maximum Field Length**
C (character)	256 characters
D (digit)	16 characters
Z (zone)	1 character
U (unpacked)	16 characters
P (packed)	8 characters

Suppose for example, you have a file of item numbers and other data that looks as follows:



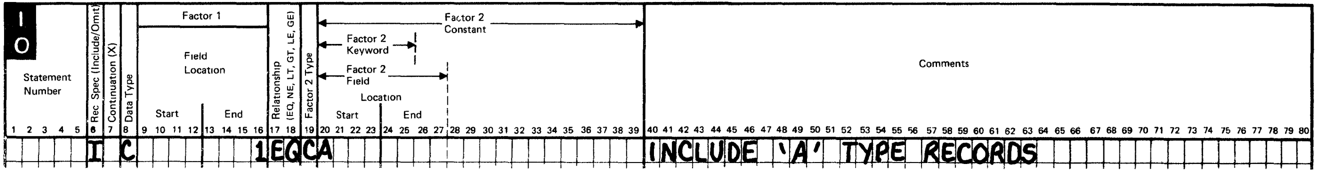
You want to define the sort job so that it groups all the item numbers and adds the amount for each group. Assume the total amount for a given item number will never exceed $9999.99. You could define the following sort specifications to do this job:

Header



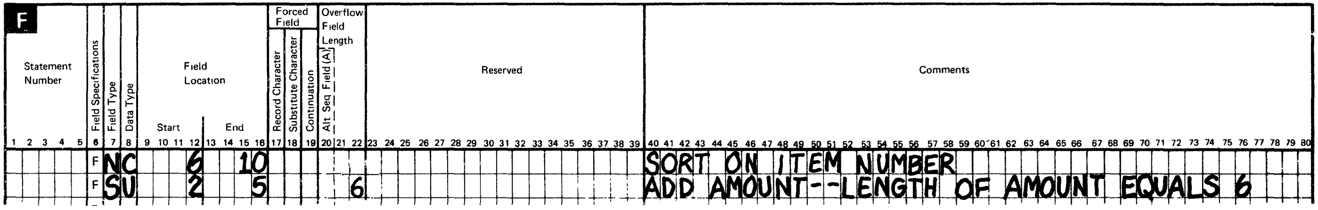
```
     HSORTRS    5A         O  11           SUMMARIZE AMOUNT FOR EACH ITEM
```

Record Selection



```
     I C       1EQCA                   INCLUDE 'A' TYPE RECORDS
```

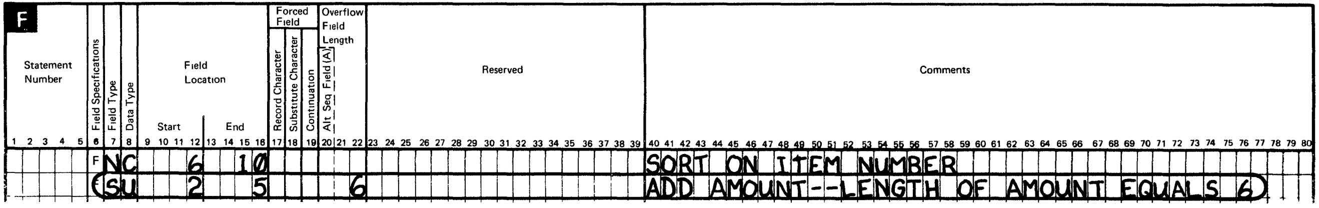
Field Selection



```
     FNC   6  10                       SORT ON ITEM NUMBER
     FSU   2   5     6                 ADD AMOUNT--LENGTH OF AMOUNT EQUALS 6
```

In this example, the records are sorted by item number (in positions 6 through 10), and selling price (in positions 2 through 5) is the summary data field. When the selling prices are added, their totals will exceed the 4-position field length. Therefore, to prevent data from overflowing and to prevent a loss of data, a larger field length is specified.

Field Selection



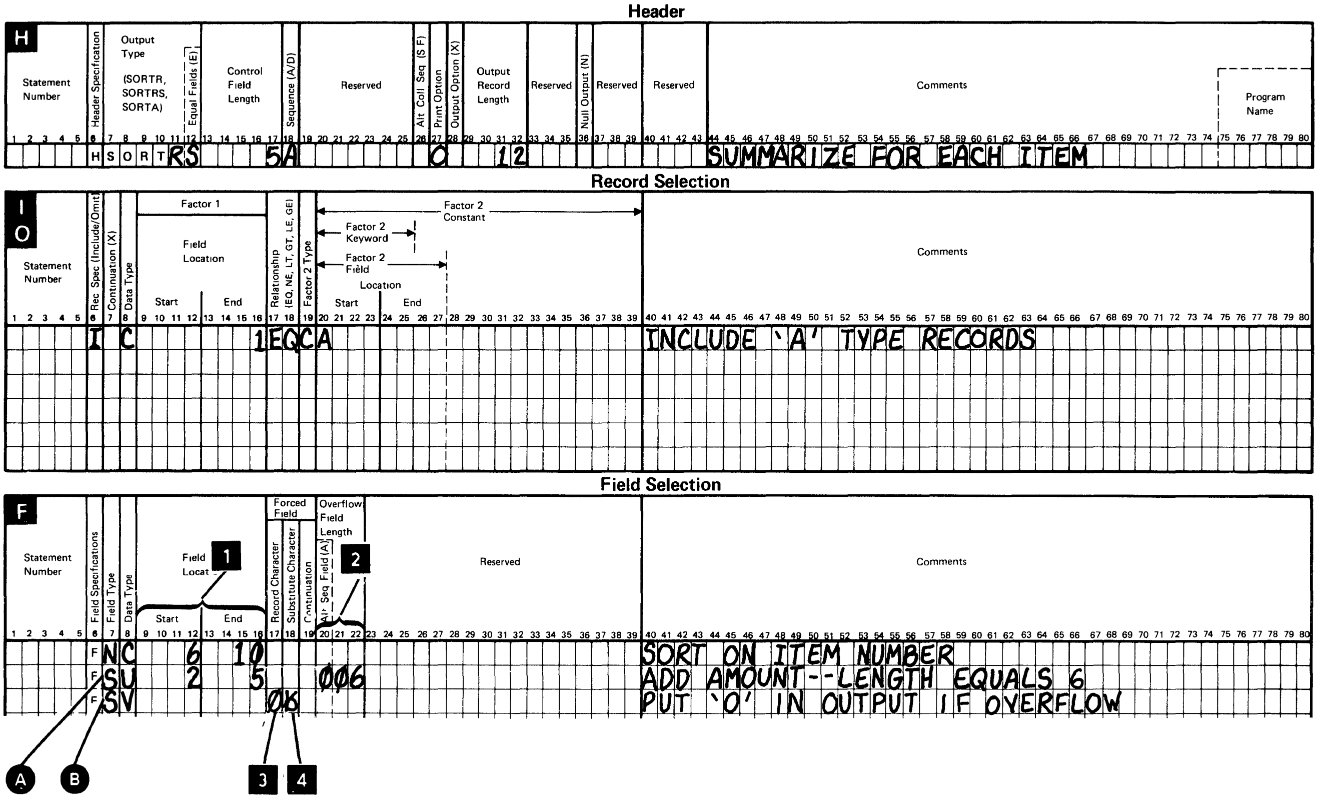
```
     FNC   6  10                       SORT ON ITEM NUMBER
     FSU   2   5     6                 ADD AMOUNT--LENGTH OF AMOUNT EQUALS 6
```

The overflow field length is defined in positions 2 through 5 and redefined as a 6-position field in overflow field length columns (20 through 22).

Detecting Summary Data Overflow

There may be times when you do not want to specify a summary overflow field length for a summary field, or you do not know how large a summary overflow field length should be specified. In these cases sort allows you to force a character constant into your output to indicate when overflow occurs. Subsequent programs that read your sort output file can then test for the presence of this character constant and know if overflow occurred for the control field.

Suppose you want to use the specifications from the previous example, but you want to ensure that no totals greater than $9999.99 are actually generated. You could define the sort specifications as follows:



Header

Statement Number (1-5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7-12)	Control Field Length (13-17)	Sequence (A/D) (18)	Reserved (19-25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29-32)	Reserved (33-35)	Null Output (N) (36)	Reserved (37-39)	Reserved (40-43)	Comments (44-74)	Program Name (75-80)
	H	SORTRS	5	A			O		12					SUMMARIZE FOR EACH ITEM	

Record Selection

Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ, NE, LT, GT, LE, GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location (20-39)	Comments (40-80)
	I		C		1	EQ	C	A	INCLUDE 'A' TYPE RECORDS

Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	6	10						SORT ON ITEM NUMBER
	F	S	U	2	5				ØØ6		ADD AMOUNT--LENGTH EQUALS 6
	F	S	V			Ø	6				PUT 'O' IN OUTPUT IF OVERFLOW

Statement Ⓐ is a summary data field. Columns 9 through 16 1 indicate the input field location and length, and columns 20 through 22 2 indicate the amount of space to be reserved for the output total. Also, in this example, the output record length in the header specification must be long enough to hold the extra positions reserved for any overflow totals:

Item number	= 5 positions
Overflow length	= 6 positions
Total output length	= 11 positions

In this example, an additional specification is required. Statement Ⓑ tells the sort program to put O 3 in the next output position if overflow occurs (while adding the amount in its 6-position output area). If overflow does not occur for any control field value, a blank 4 will be placed in the overflow indicator position.

Also in the preceding example, the output length entered in the header specification is increased by 1 (to 12) to allow for the overflow indicator position.

Only one summary overflow indicator can be specified per record type. If multiple summary overflow indicators are specified (one for each record type), only the final one specified will have any effect on the output record.

Steps to Follow When Specifying Overflow Indicators

Use the following steps to specify an overflow indicator field:

1. Fill in columns 1 through 6 on the field specifications as you would for any control field, with the page and line numbers, if necessary.

2. Enter an S (for summary) in column 7 of the field specifications.

3. Put a V (for forced field) in column 8 of the field specifications.

4. Leave columns 9 through 16 of the field specifications blank.

5. Enter a character in column 17 to which the overflow indicator field will be set if overflow occurs in any summary data field.

   If overflow occurs in any summary data fields, the character in column 17 will be written into the overflow indicator field.

   If overflow occurs and no overflow character is identified (that is, column 17 contains a blank) the system will set the overflow indicator field to an asterisk (*).

6. Enter a character in column 18 to which the overflow indicator field will be set if an overflow does not occur in any summary data fields.

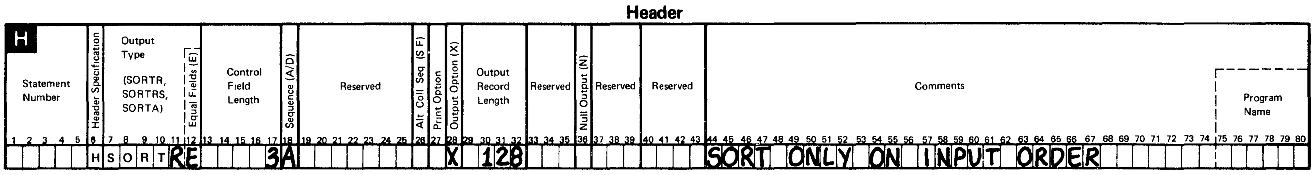
   If none of the summary data fields overflow, the overflow indicator field will contain the character specified in column 18. This character is forced into the overflow indicator field. If column 18 contains a blank, the overflow indicator field will be set to a blank.

Forced Data Fields

When running a regular sort (SORTR) or a summary sort (SORTRS), you can force a character constant at selected locations in the output data.

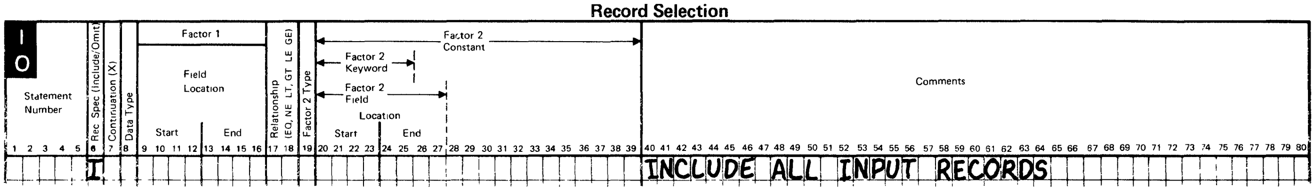
Assume, for example, that you have a file with two types of records, and each type of record has 128 characters. An active record is indicated by an A in position one and a deleted record is indicated by a D in position one. Suppose you have the job of resetting the entire file to all active records.

You could use the following sort specifications to do this job:



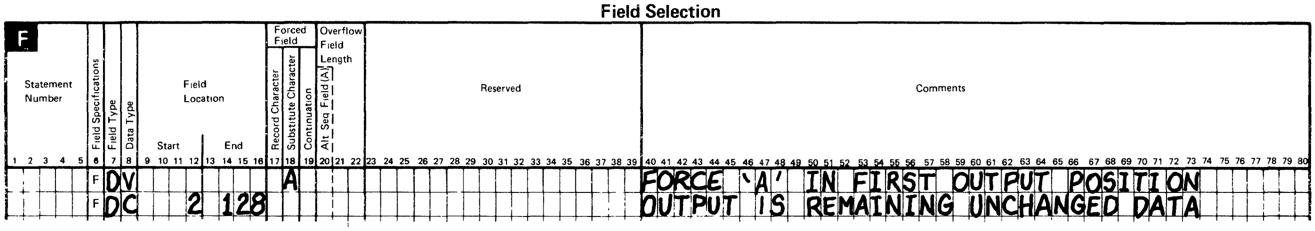
Header

Header Specification (6)	Output Type (7-11)	Equal Fields (E) (12)	Control Field Length (13-17)	Sequence (A/D) (18)	Output Option (X) (28)	Output Record Length (29-32)	Comments (40-74)
H	SORTR	E	3	A	X	128	SORT ONLY ON INPUT ORDER



Record Selection

Rec Spec (Include/Omit) (6)	Comments (40-74)
I	INCLUDE ALL INPUT RECORDS



Field Selection

Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Comments (40-74)
F	D	V			A	FORCE 'A' IN FIRST OUTPUT POSITION
F	D	C	2	128		OUTPUT IS REMAINING UNCHANGED DATA

In the preceding example, equal control field ordering was specified in the header specification. By setting the total control field length to 3, you tell the sort program that the input relative record number is the only control field. The field specifications immediately define data field specification that forces the character constant A into the first output position. The remainder of the output comes' ' from positions 2 through 128 of the input record.

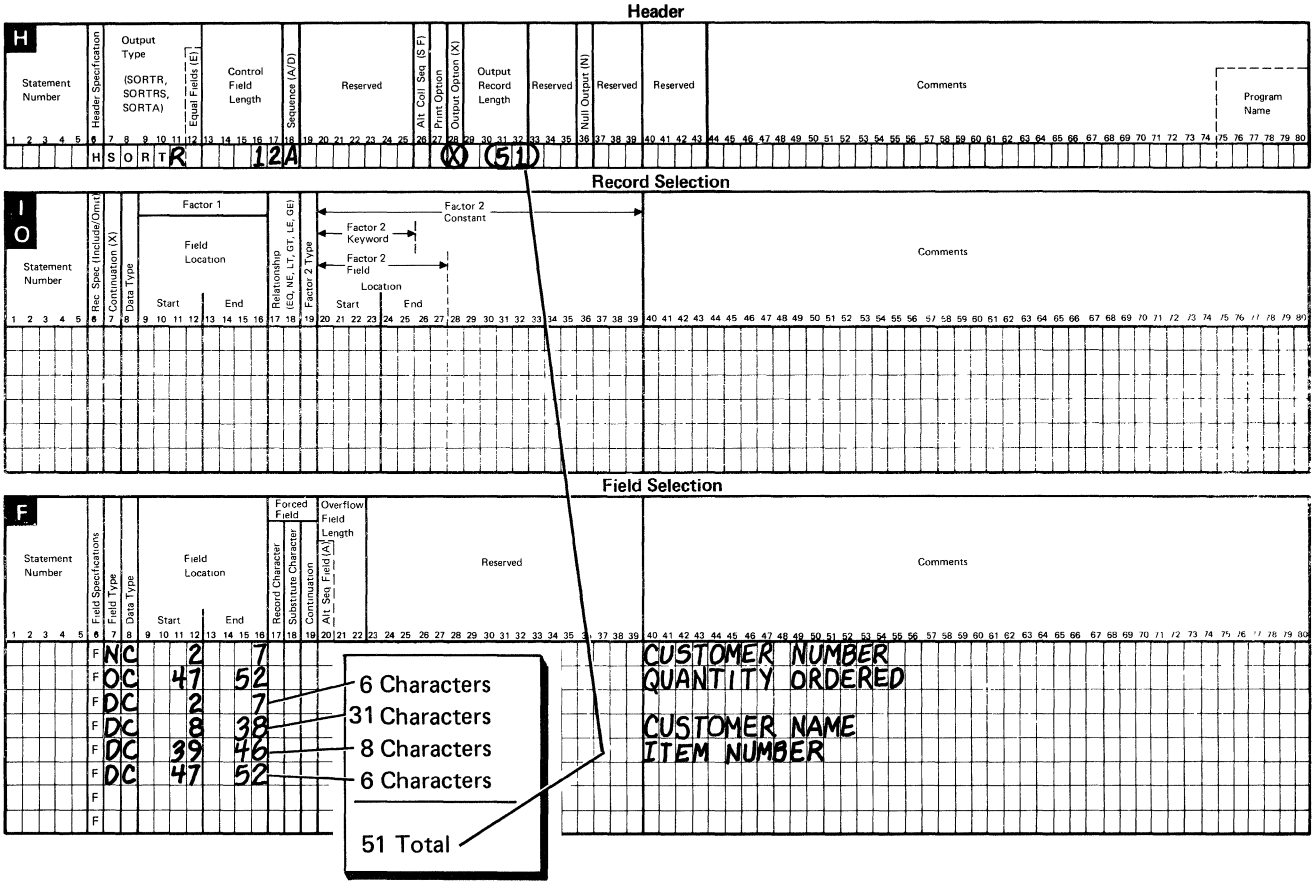
Determining the Output Record Length for Sort Jobs

You must determine the length of the output records and specify this length to the sort program. Output record lengths apply to SORTR and SORTRS jobs only.

The length of the output record is determined by the data fields you specify in the field specifications and is influenced by whether or not you write the control field into the output file.

If you do not write the control field into the output file, the output record length will include only the data fields. The following example shows the control field specified as a data field.

Assume, in this example, you are taking inventory and you want to find out which items and how many were ordered by your customer. You could do this by sorting an input file by customer number (in ascending order) and quantity ordered (in descending order). You also want to include the customer numbers, customer names, item numbers, and the quantities ordered in the output file in this order.



Header

Header Specification	Output Type (SORTR, SORTRS, SORTA)	Control Field Length	Sequence (A/D)	Print Option	Output Record Length
H	SORTR	12	A	X	51

Record Selection

Field Selection

Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Comments
F	N	C	2	7	CUSTOMER NUMBER
F	O	C	47	52	QUANTITY ORDERED
F	D	C	2	7	
F	D	C	8	38	CUSTOMER NAME
F	D	C	39	46	ITEM NUMBER
F	D	C	47	52	
F					
F					

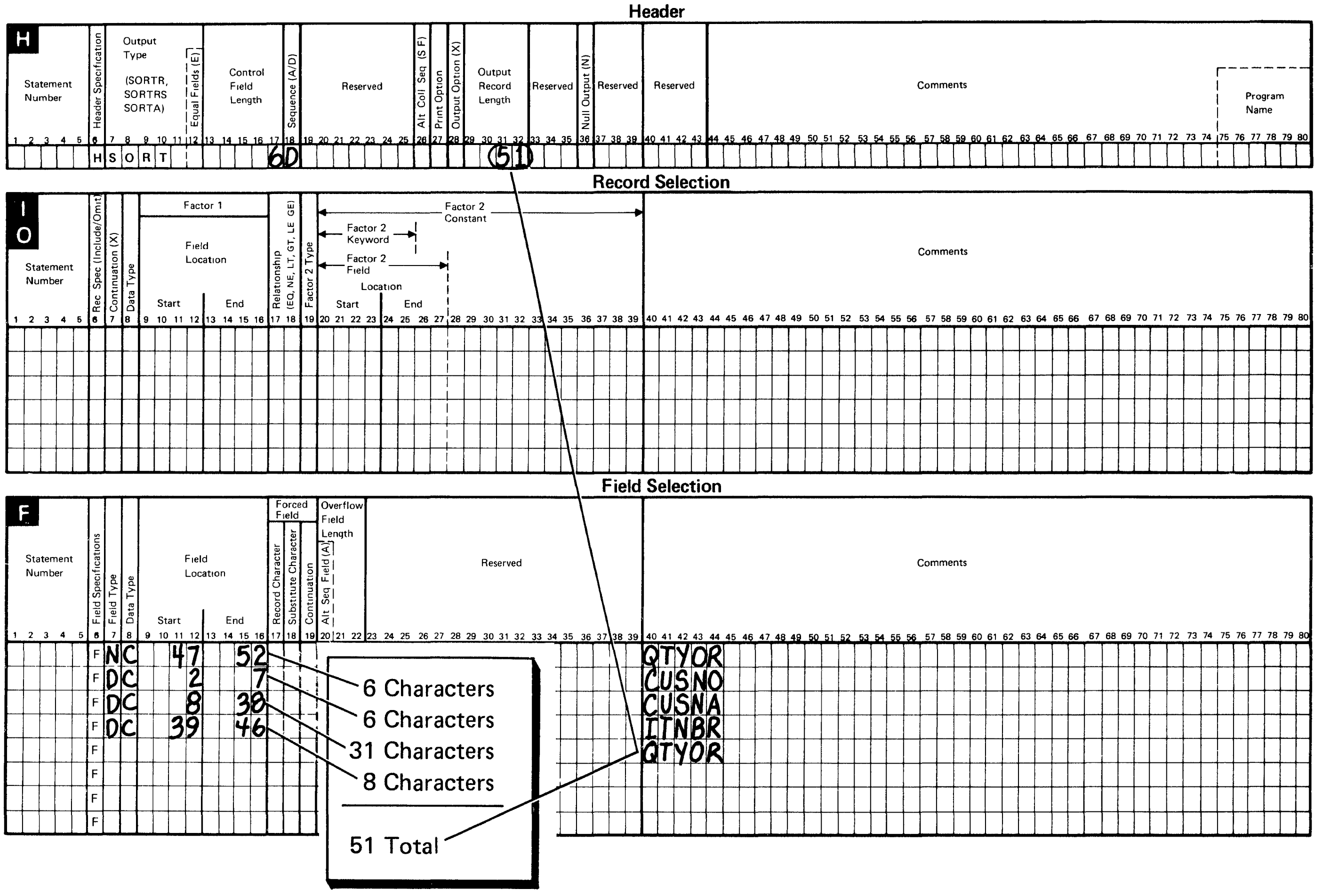
6 Characters
31 Characters
8 Characters
6 Characters

51 Total

The X in the output option column of the header specification indicates that the control field should be dropped. This entry forces you to redefine the opposite control field as a data field. Otherwise, its contents would appear in the output file in a form different from the input.

To calculate the output record length, just add the lengths of all the data fields (6, 31, 8, 6) and enter the sum (51) into the output record length columns of the header specification as shown.

If you have the control field data written into the output file, (you leave the output option column of the header specification blank), the output record length will include the control field(s) and data fields, as shown in the following example:


Header
Statement Number
Header Specification
Output Type (SORTR, SORTRS SORTA)
Equal Fields (E)
Control Field Length
Sequence (A/D)
Reserved
Alt Coll Seq (S F)
Print Option
Output Option (X)
Output Record Length
Null Output (N)
Comments
Program Name
Record Selection
Rec Spec (Include/Omit)
Continuation (X)
Data Type
Factor 1
Field Location
Start
End
Relationship (EQ, NE, LT, GT, LE GE)
Factor 2 Type
Factor 2 Constant
Factor 2 Keyword
Factor 2 Field
Location
Field Selection
Field Specifications
Field Type
Forced Field
Record Character
Substitute Character
Continuation
Overflow Field Length
Alt Seq Field (A)

H SORT 6 D 51

Field Specifications	Field Type	Data Type	Start	End	Comments
F	N	C	47	52	QTYOR
F	D	C	2	7	CUSNO
F	D	C	8	38	CUSNA
F	D	C	39	46	ITNBR
F					QTYOR
F					
F					
F					

6 Characters
6 Characters
31 Characters
8 Characters

51 Total

Commenting on Your Sort Job

You can describe and comment on your sort job in the comment columns of the sort specifications. Any characters can go in these columns, except a plus sign (+) cannot be the last character in any comment if the specifications are in a procedure. If it is, the sort program may interpret the plus sign as a continuation character and replace it with the next sort specification statement.

Comments are a good way to document information about your sort jobs. These comments can be printed and kept as a record along with your sort specifications. If you specify that a sort job print the sort specifications and you made comments about the job, both your comments and the sort specifications are printed. The comments have no effect on how the program works.

Compiling Sort Specifications

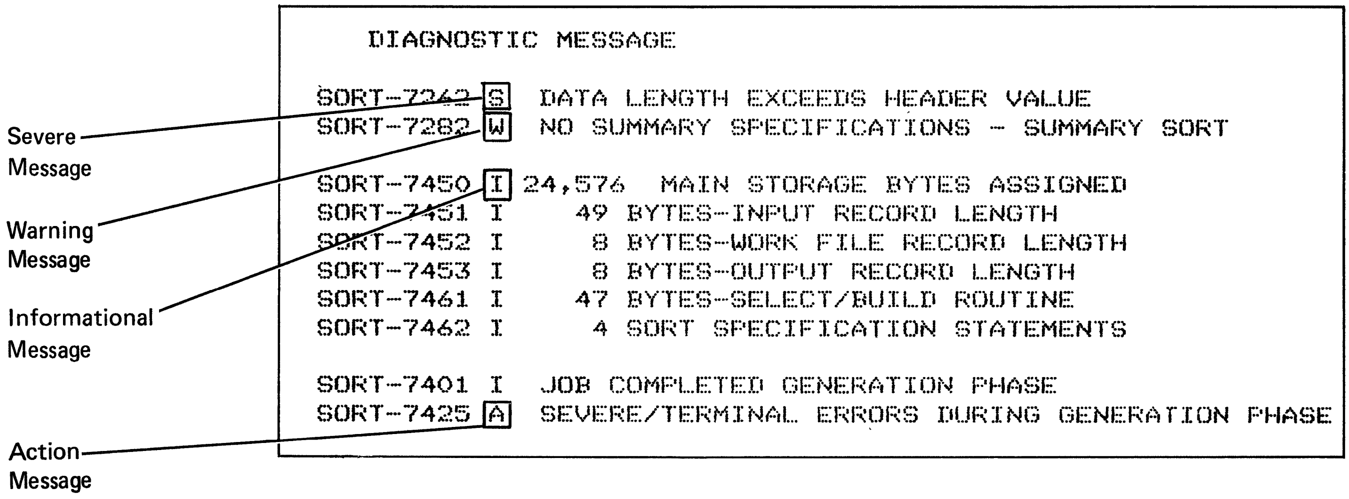
The system treats sort specifications like source data. Your sort specifications are considered to be source data to the system. This sort source is compiled when you run a sort job. The sort specifications are processed as they are encountered by the sort program.

What If You Get an Error During a Compile?

Sometimes during a sort run, the sort program detects an error in the sort specifications. If you get an error during a sort run, you should refer to the appropriate section of the *Messages Guide* for more information.

Sort Messages

The following types of messages might be generated by the sort program.



```
   DIAGNOSTIC MESSAGE

SORT-7242 S  DATA LENGTH EXCEEDS HEADER VALUE
SORT-7282 W  NO SUMMARY SPECIFICATIONS - SUMMARY SORT

SORT-7450 I 24,576  MAIN STORAGE BYTES ASSIGNED
SORT-7451 I     49 BYTES-INPUT RECORD LENGTH
SORT-7452 I      8 BYTES-WORK FILE RECORD LENGTH
SORT-7453 I      8 BYTES-OUTPUT RECORD LENGTH
SORT-7461 I     47 BYTES-SELECT/BUILD ROUTINE
SORT-7462 I      4 SORT SPECIFICATION STATEMENTS

SORT-7401 I  JOB COMPLETED GENERATION PHASE
SORT-7425 A  SEVERE/TERMINAL ERRORS DURING GENERATION PHASE
```

Chapter 3. Entering and Storing Sort Specifications in System/36

After you design a sort job and define the specification statements to generate the kind of sorted output you want, you then enter the specification statements into the system. Use the Source Entry Utility (SEU), which is described in detail in the *Source Entry Utility Guide*, to enter your specification statements into the system.

Before you enter the sort specifications into the system, be sure they are in the correct order.

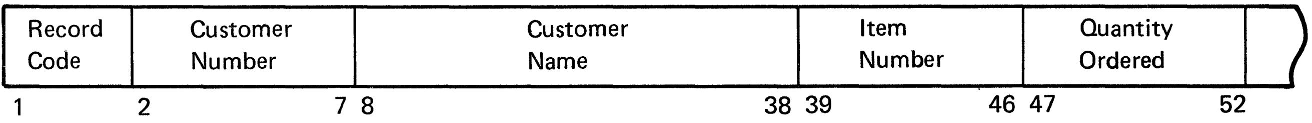
Order of Entering the Sort Specifications into the System

Ordinarily, you enter the sort specifications into the system in their order of appearance on the sort specifications coding form. The header specification must be present and placed first in the source member, next the record selection specifications, then the field selection specifications. A source member is an area of storage in a library on the system disk.

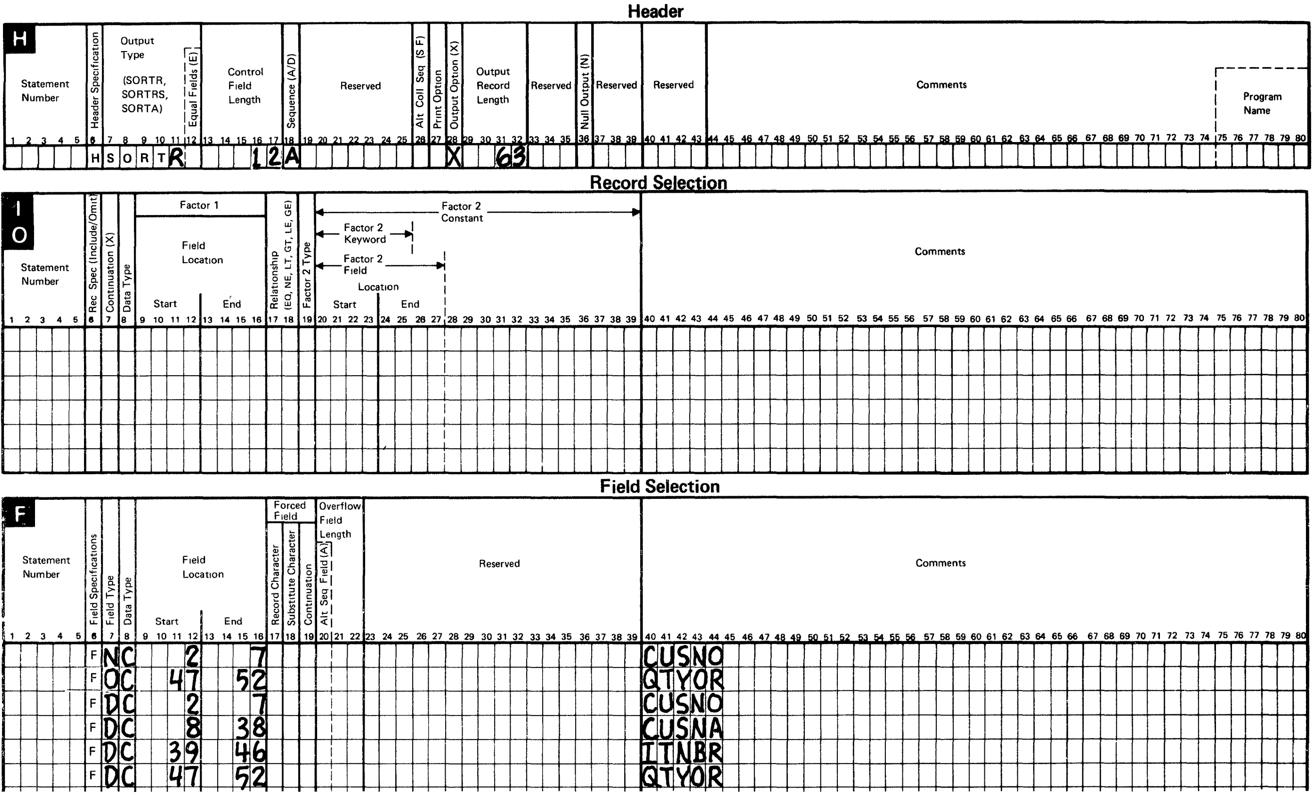
The order of the record and field selection specifications can vary, however. You may not have to fill out all three types of specifications depending on the number of records you want to sort and the format of those records. (Format refers to the locations, lengths, and types of fields in a record.)

For example, you might want to sort all the records in a file, all of which have the same format (are of one type) **1**. In this case, you need not fill out record selection specifications. You fill out the header **2** and field **3** selection specifications only as shown in the following example. In this example, the header specification identifies the job to the system and the field specifications identify the fields in the input record that will be written into the output file. Assume all the records in your input file have the following format:

1



Record Code	Customer Number	Customer Name	Item Number	Quantity Ordered
1	2 7	8 38	39 46	47 52



Header

2 **H**

Statement Number (1–5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7–11)	Equal Fields (E) (12)	Control Field Length (13–17)	Sequence (A/D) (18)	Reserved (19–25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29–32)	Reserved (33–35)	Null Output (N) (36)	Reserved (37–39)	Reserved (40–43)	Comments (44–74)	Program Name (75–80)
	H	SORTR		12	A				X	63						

Record Selection

I **O**

Statement Number (1–5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9–12)	Factor 1 Field Location End (13–16)	Relationship (EQ, NE, LT, GT, LE, GE) (17–18)	Factor 2 Type (19)	Factor 2 Constant / Factor 2 Keyword / Factor 2 Field Location Start (20–23)	Factor 2 Field Location End (24–27)	(28–39)	Comments (40–80)

Field Selection

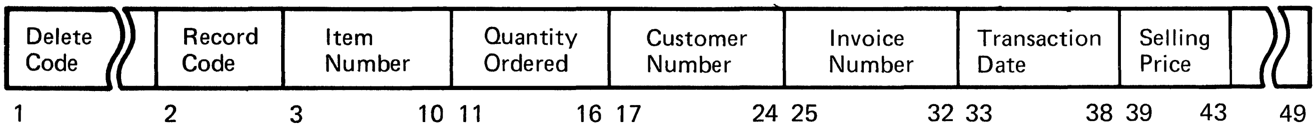
3 **F**

Statement Number (1–5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9–12)	Field Location End (13–16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt Seq Field (A) (20–22)	Reserved (23–39)	Comments (40–80)
	F	N	C	2	7						CUSNO
	F	O	C	47	52						QTYOR
	F	D	C	2	7						CUSNO
	F	D	C	8	38						CUSNA
	F	D	C	39	46						ITNBR
	F	D	C	47	52						QTYOR

You may want to sort a file 1 with several different types of records. In this case, for the first type of record you should define and enter all the specifications (the header 2, record selection 3, and field selection specifications 4).

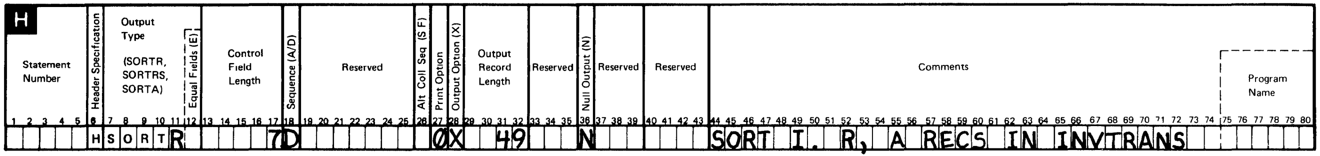
I-type records contain delivery information.

1



2

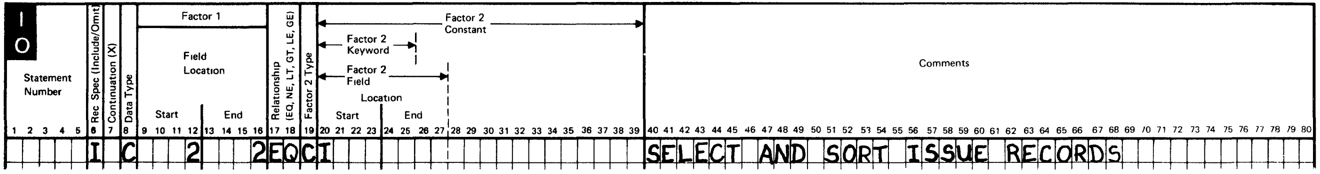
Header


H
Statement Number
Header Specification
Output Type (SORTR, SORTRS, SORTA)
Equal Fields (E)
Control Field Length
Sequence (A/D)
Reserved
Alt Coll Seq (S F)
Print Option
Output Option (X)
Output Record Length
Reserved
Null Output (N)
Reserved
Reserved
Comments
Program Name

```
     HSORTR     7D         ØX 49   N       SORT I, R, A RECS IN INVTRANS
```

3

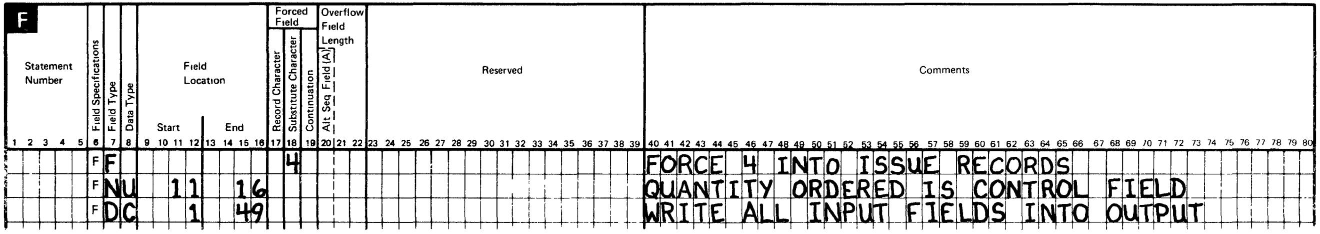
Record Selection


I
O
Statement Number
Rec Spec (Include/Omit)
Continuation (X)
Data Type
Factor 1
Field Location
Start
End
Relationship (EQ, NE, LT, GT, LE, GE)
Factor 2 Type
Factor 2 Constant
Factor 2 Keyword
Factor 2 Field Location
Start
End
Comments

```
     I C   2   2EQCI                   SELECT AND SORT ISSUE RECORDS
```

4

Field Selection


F
Statement Number
Field Specifications
Field Type
Data Type
Field Location
Start
End
Forced Field
Record Character
Substitute Character
Continuation
Alt Seq Field (A)
Overflow Field Length
Reserved
Comments

```
     FF          4                     FORCE 4 INTO ISSUE RECORDS
     FNU  11  16                       QUANTITY ORDERED IS CONTROL FIELD
     FDC   1  49                       WRITE ALL INPUT FIELDS INTO OUTPUT
```

Then define and enter only the record 6 and field selection specifications 7 for the next type of records 5.

R-type records 5 contain order receipt data.

5

Delete Code	Record Code	Item Class	Item Number	Selling Price	Purchase Order Number	Quantity Ordered	Transaction Date	Blank
1	2	3 4	5 12	13 17	18 21	22 27	28 33	49

6

Record Selection

```
         1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
     I C   2   2EQCR                   SELECT AND SORT RECEIPT RECORDS
```

7

Field Selection

```
         1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
     FF           3                     FORCE 3 INTO RECEIPT RECORDS
     FNU  22  27                       QUANTITY ORDERED IS CONTROL FIELD
     FDC   1  49                       WRITE DATA INTO POS 1-49 IN OUTPUT FILE
```

Then define and enter the record 9 and field selection specifications 10 for the third type of records 8, and so on.

A-type records 8 contain adjustment information.

8

Delete Code	Record Code	Item Number	Adjustment Code	Quantity Ordered	Transaction Date	Warehouse	Blank
1	2	3 10	11	12 17	18 23	24	25 49

9

Record Selection

```
         1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
     I C   2   2EQCA                   SELECT ADJUSTMENT RECORDS ONLY
```

10

Field Selection

```
         1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
     FFC  11  112                      FORCE 2 FOR ADDITIONS TO STOCK
     FFC  11  1121X                    FORCE 1 FOR SHORT ITEMS IN STOCK
     FNU  12  17                       QUANTITY ORDERED IS CONTROL FIELD
     FDC   1  49                       WRITE DATA IN POS 1-49 IN OUTPUT FILE
```

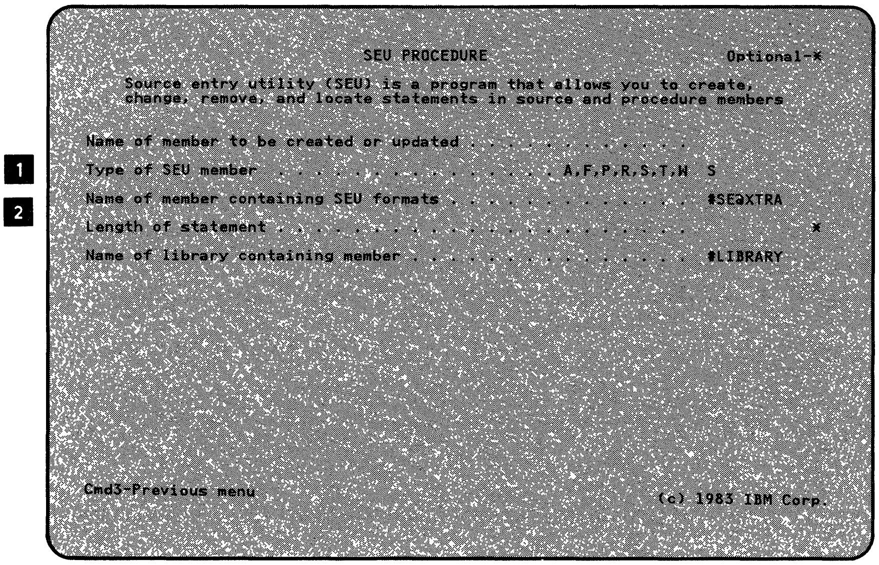
For multiple record-type sorts, you should remember that each sort step orders the records included in that step by the order specified in that step. Once the records have been sorted, their order in the output file remains constant; however, subsequent sort steps may change their overall position within the output file.

Note: The multiple record-type sorting technique should only be used on files with multiple record types. If a record type is included in more than one sort step, an unexpected output may occur.

Entering the Sort Specifications

Once you establish the order in which you should enter your sort specifications into the system, use the Source Entry Utility (SEU) and place the specifications into a source member. Before starting, you should become familiar with SEU or have a copy of the *Source Entry Utility Guide* at hand as you enter the specifications.

In response to the SEU command procedure prompt:



```
                         SEU PROCEDURE                          Optional-*

     Source entry utility (SEU) is a program that allows you to create,
     change, remove, and locate statements in source and procedure members

  Name of member to be created or updated . . . . . . . . . . .
  Type of SEU member  . . . . . . . . . . . . . . . . A,F,P,R,S,T,W  S
  Name of member containing SEU formats . . . . . . . . . . . .  #SE@XTRA
  Length of statement . . . . . . . . . . . . . . . . . . . . .         *
  Name of library containing member . . . . . . . . . . . . . .  #LIBRARY




  Cmd3-Previous menu                                      (c) 1983 IBM Corp.
```

you must enter:

1 The name of the member to be created

This member name can be the same name you specified in the program name columns of the sort specifications coding form.

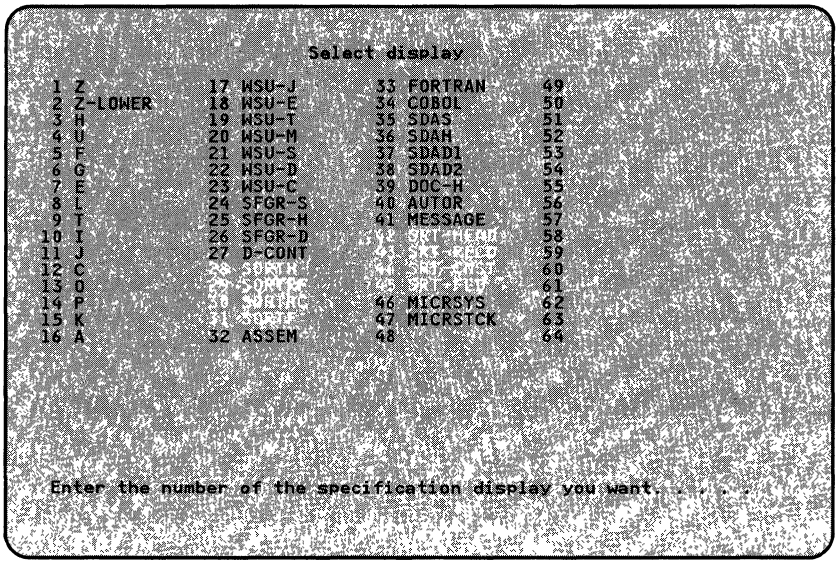
2 The type of member you are creating

Whenever you enter just the sort specifications, you must specify S (the member is a **source member**).

Whenever you enter both the sort specifications and the OCL statements into a library member, or just the OCL statements, you must specify P (the member is a **procedure member**).

The other options in the display have defaults and do not require an entry.

SEU provides the following Select Display menu from which you can choose the sort displays. See the *Source Entry Utility Guide* for more information about this menu and how to use this utility.

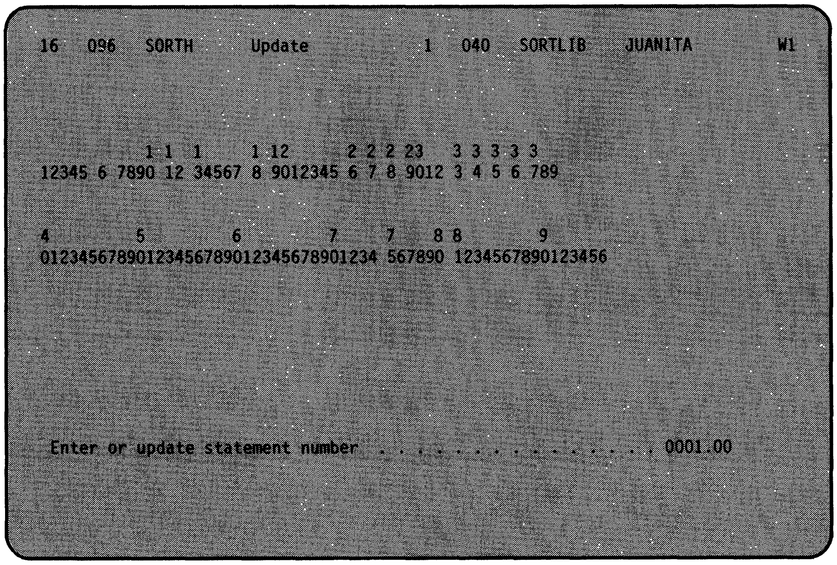


```
                       Select display
  1 Z             17 WSU-J         33 FORTRAN      49
  2 Z-LOWER       18 WSU-E         34 COBOL        50
  3 H             19 WSU-T         35 SDAS         51
  4 U             20 WSU-M         36 SDAH         52
  5 F             21 WSU-S         37 SDAD1        53
  6 G             22 WSU-D         38 SDAD2        54
  7 E             23 WSU-C         39 DOC-H        55
  8 L             24 SFGR-S        40 AUTOR        56
  9 T             25 SFGR-H        41 MESSAGE      57
 10 I             26 SFGR-D        42 SRT-HEAD     58
 11 J             27 D-CONT        43 SRT-RECD     59
 12 C             28 SORTH         44 SRT-CNST     60
 13 O             29 SORTRF        45 SRT-FLD      61
 14 P             30 SORTRC        46 MICRSYS      62
 15 K             31 SORTF         47 MICRSTCK     63
 16 A             32 ASSEM         48              64

 Enter the number of the specification display you want. . . .
```

You can display:

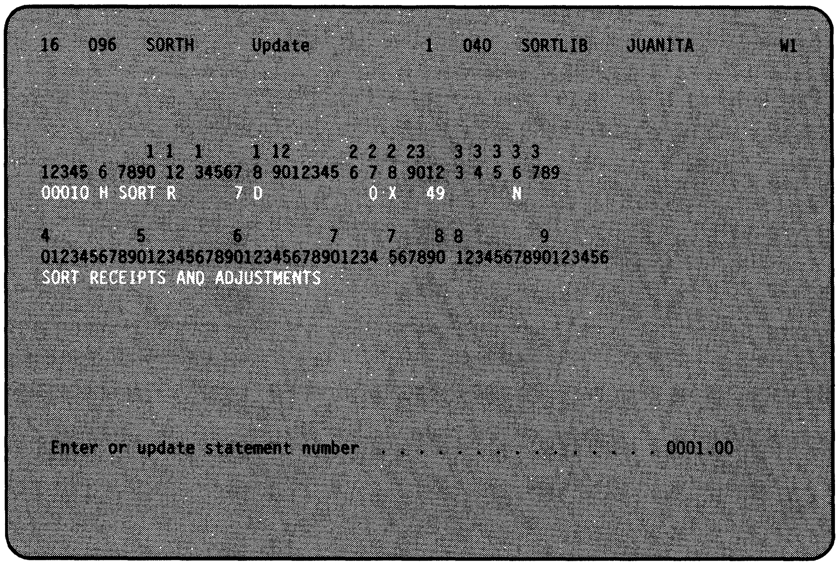
SORTH and SRT-HEAD	for the header specification
SORTRF and SRT-RECD	for record selection specifications that compare a field to another field in the record
SORTRC and SRT-CNST	for record selection specifications that compare fields to a constant or date keyword
SORTF and SRT-FLD	for field specifications

Displays 28 through 31 (of the SEU Select display) provide 96 columns for the sort specification entries and comments, as shown in the following example:



```
16   096   SORTH      Update          1   040   SORTLIB    JUANITA        W1



           1 1  1     1 12      2 2 2 23   3 3 3 3 3
12345 6 7890 12 34567 8 9012345 6 7 8 9012 3 4 5 6 789

4         5         6         7     7    8 8        9
01234567890123456789012345678901234 567890 1234567890123456




 Enter or update statement number . . . . . . . . . . . . . . 0001.00
```

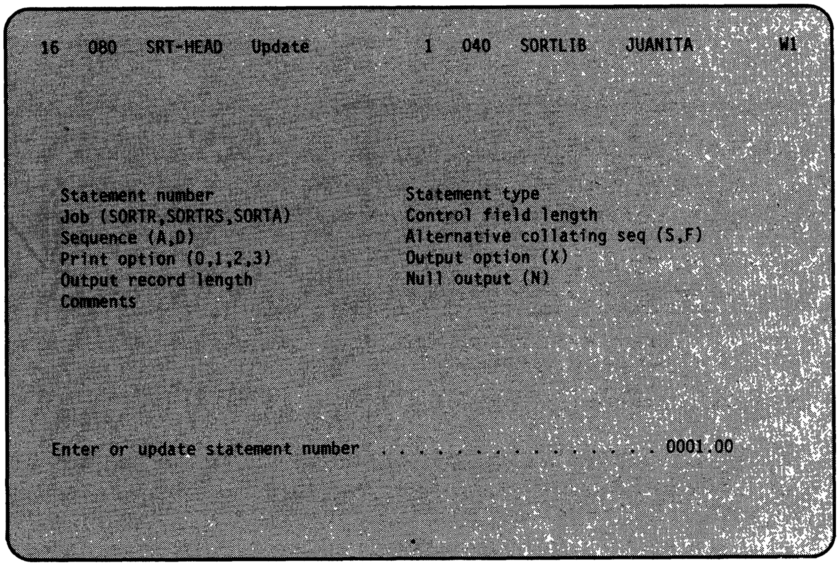
When filled out, the SORTH specification looks as follows:



```
16   096   SORTH      Update          1   040   SORTLIB    JUANITA        W1



           1 1  1     1 12      2 2 2 23   3 3 3 3 3
12345 6 7890 12 34567 8 9012345 6 7 8 9012 3 4 5 6 789
00010 H SORT R      7 D           O X   49      N

4         5         6         7     7    8 8        9
01234567890123456789012345678901234 567890 1234567890123456
SORT RECEIPTS AND ADJUSTMENTS



 Enter or update statement number . . . . . . . . . . . . . . 0001.00
```

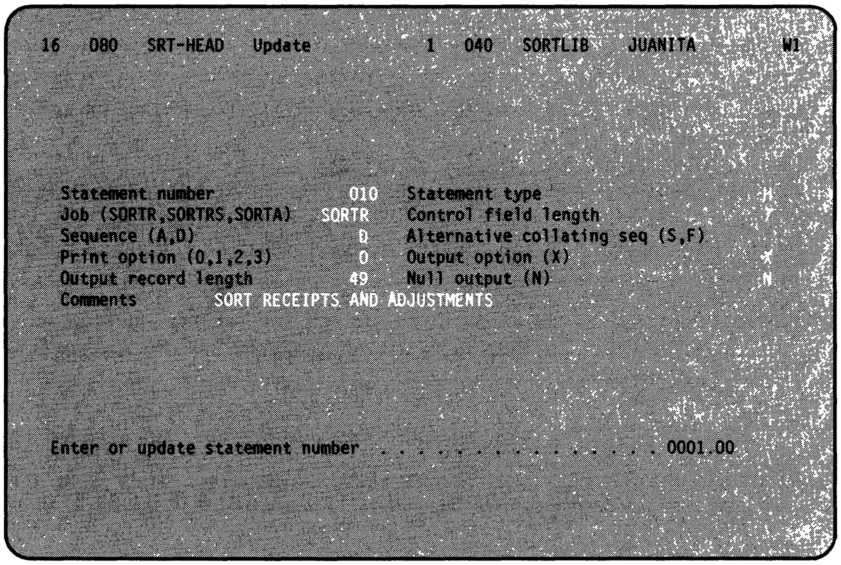
Displays 41 through 44 (of the SEU Select display) contain prompts to which you respond. Each prompt is associated with the column headings on the sort specification coding form.

The following example shows the header (41 SRT-HEAD) specification display you can select from this menu:



```
16  080  SRT-HEAD  Update           1  040  SORTLIB   JUANITA        W1



 Statement number                   Statement type
 Job (SORTR,SORTRS,SORTA)           Control field length
 Sequence (A,D)                     Alternative collating seq (S,F)
 Print option (0,1,2,3)             Output option (X)
 Output record length               Null output (N)
 Comments




Enter or update statement number . . . . . . . . . . . . . . 0001.00
```

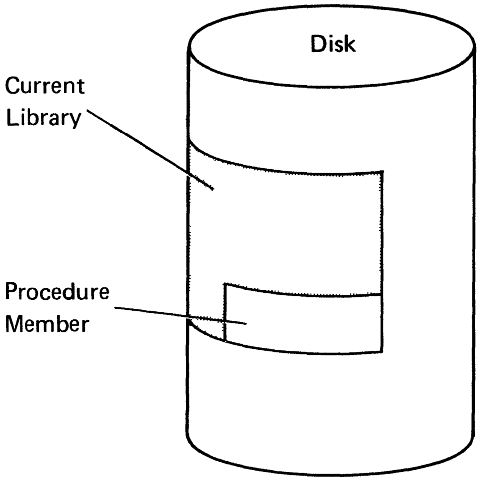
When filled out, the header SRT-HEAD specification display looks as follows:



```
16  080  SRT-HEAD  Update           1  040  SORTLIB   JUANITA        W1



 Statement number              010  Statement type                   H
 Job (SORTR,SORTRS,SORTA)    SORTR  Control field length             7
 Sequence (A,D)                  D  Alternative collating seq (S,F)
 Print option (0,1,2,3)          0  Output option (X)                X
 Output record length           49  Null output (N)                  N
 Comments         SORT RECEIPTS AND ADJUSTMENTS




Enter or update statement number . . . . . . . . . . . . . . 0001.00
```

Entering the Sort Specifications and the OCL Statements into a Library Member

You can enter the sort specifications and the OCL statements into the same library member, a procedure member. A sort procedure member contains both the OCL statements and the sort specifications needed to run a sort job. As shown in the following illustration, a procedure member is stored in a library on disk:



If you enter the sort specifications and the OCL statements into the same library member, you must enter them into a procedure member as follows:

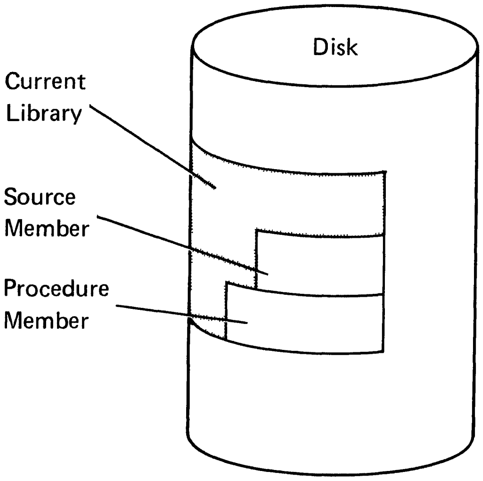
```
1// LOAD #GSORT
2// FILE NAME-INPUT,LABEL-ITEMBALN
3// FILE NAME-WORK,LABEL-SORTWORK,BLOCKS-20,RETAIN-S
4// FILE NAME-OUTPUT,LABEL-MIKE,BLOCKS-10,RETAIN-T
5// RUN
   Ⓐ HSORTR      8A         OX  24   N
   Ⓑ I P   19  22LEF  23  26                AVAIL FLD LESS OR EQUAL TO REORD
   Ⓒ FNP   19  22                           AVAIL NUMBER OF UNITS AVAILABLE
   Ⓓ FNP   23  26                           REORD REORDER POINT
   Ⓔ FDC    2   9                           ITNBR ITEM NUMBER
6// END
```

The statements numbered 1 through 6 are the OCL statements required to create a procedure that will run a sort job. The statements labeled Ⓐ through Ⓔ are the sort specifications that describe the sort job.

Do not let a plus (+) sign be the last nonblank character in a sort specification statement contained in a procedure. If necessary, add a comment after the plus (+) sign. Otherwise, the program will assume the next line is a continuation of the line with the plus (+) sign, and try to treat it that way.

Entering the Sort Specifications and OCL Statements into Separate Library Members

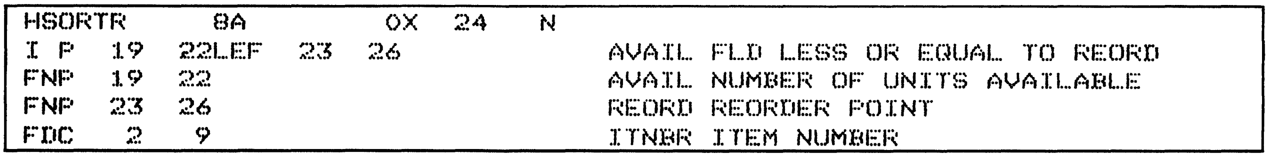
You can enter the sort specifications and the OCL statements into two different library members as shown in the following illustration:



If you enter the sort specifications and the OCL statements into separate library members, you enter the sort specifications into a **source member** and the OCL statements into a separate member, a **procedure member**.

Note: When you put sort specifications into a procedure member, be sure that none of the sort specifications contain a plus (+) sign as the last nonblank character in the specifications statement. If it is, when the statement is read from the procedure member, the plus sign is treated as a continuation sign and the statement following the plus sign will be read and chained to the preceding statement.

The following is an example of a source member containing sort specifications only. Use SEU to enter them:



```
HSORTR     8A         OX  24   N
I P  19   22LEF  23  26             AVAIL FLD LESS OR EQUAL TO REORD
FNP  19   22                        AVAIL NUMBER OF UNITS AVAILABLE
FNP  23   26                        REORD REORDER POINT
FDC   2    9                        ITNBR ITEM NUMBER
```

The following example shows a procedure member that will call a source member to run a sort job:

```
// LOAD #GSORT
// FILE NAME-INPUT,LABEL-ITEMBALN
// FILE NAME-WORK,LABEL-SORTWORK,BLOCKS-20,RETAIN-S
// FILE NAME-OUTPUT,LABEL-MIKE,BLOCKS-10,RETAIN-T
// RUN
// SOURCE REORDER,JMGLIB
```

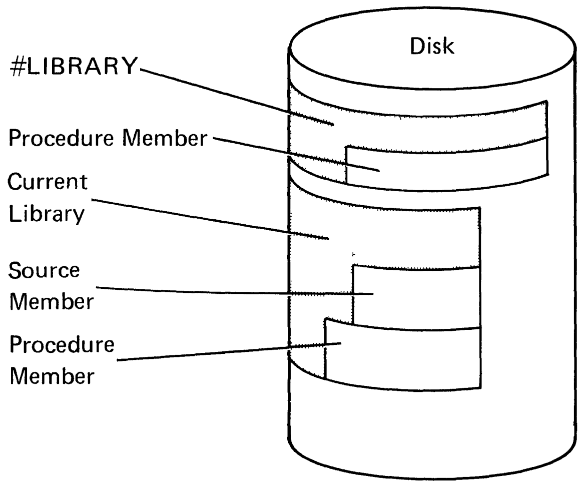
In this example, the sort specifications are stored in a source member in a library. The source member name is REORDER, it is stored in a user library call JMGLIB.

When the procedure is run, the sort program reads the sort specifications from REORDER.

You can enter the procedure OCL that will run the sort job directly through a display station keyboard, that is, without storing the statements in a procedure member. If you use this method of entry, after you run the job, the sort procedure statements are lost and must be entered again for the next sort job. For this reason, it is a good idea to enter and store them in a procedure or source member.

Storing Sort Specifications and OCL Statements in the System

After you enter your sort specifications and/or OCL statements into the system you can store them in a library. If you do not specify a library, the system will store the statements in the current library. The current library can be either a library you created, or the system library (#LIBRARY) as shown in the illustration:



It is important to note that if you store the sort specifications in a library you created, sort requires that you name this library in the // SOURCE statement. See the *System Reference* manual for more information about storing source statements and creating libraries on disk.

Chapter 4. How to Run Sort

To run a sort job, you can do one of the following:

- Use the SORT procedure command.
- Use your own sort procedure (created with OCL statements).
- Call sort from an assembler or a COBOL program. (This method of running sort is discussed in detail in Appendix C.)

Using the SORT Procedure Command to Run Sort

The SORT procedure command permits you to run some sort jobs without your having to write procedures.

Considerations and Requirements for Using the SORT Procedure Command

When you use the SORT procedure command, you must store the sort specifications in a source member (library member) before you run the job. Also, the input and output files must meet certain requirements, as discussed in the next two topics.

Input File Requirements

- Use only one file, local or remote, as input to the sort program.
- Make sure only one file on the disk has the input file label. If you use the SORT procedure command and the sort program encounters duplicate file labels, it will sort the file with the most recent date. See the *System Reference* manual for information about the system date.

Output File Requirements

Make sure the output file, local or remote, does not currently exist. Unlike the sorted file produced when you run a job using an OCL procedure, the sorted file produced by the SORT command cannot be written over any existing file on the disk (including the input file).

The sort program automatically designates the output file as a **resident file** that is not delete capable and whose size cannot be increased.

Additional File Considerations

If the sort procedure is only one step in a procedure that contains several job steps, and you want to release the display station from the job step once the sort job is initiated, a // END statement must follow the SORT command statement in the procedure. (In this case, the sort job is considered one part of a much larger job having several parts.) Otherwise, you might produce unpredictable results in your sorted output. (See the // ATTR OCL statement in the *System Reference* manual for information about releasing display stations from job steps.)

Entering the SORT Procedure Command

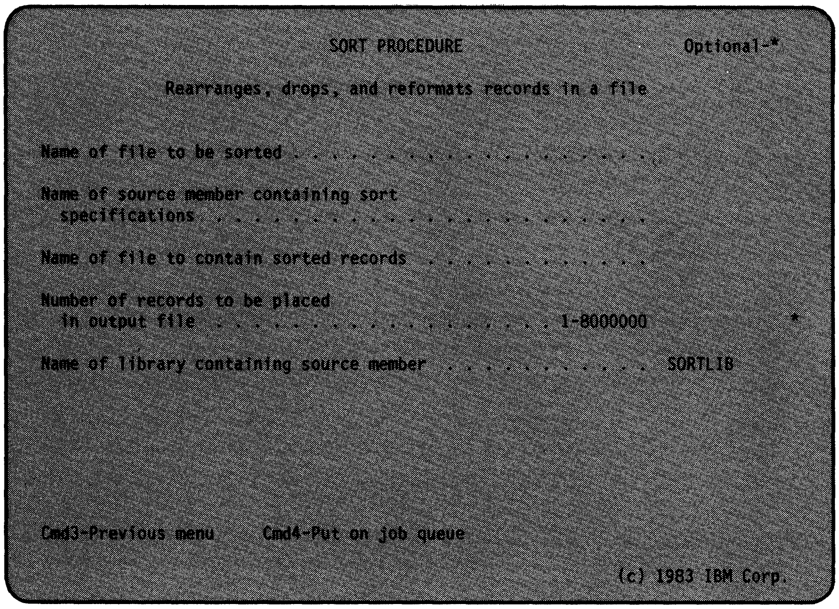
You can enter the SORT procedure command into the system in two ways:

- You can enter only the word SORT and be prompted for the SORT procedure parameters.
- You can enter the entire procedure command statement.

  When all required parameters are specified, the sort job will run without additional prompting for parameters.

Entering Only the Word SORT

If you enter only the word SORT, the following display appears:



```
                         SORT PROCEDURE                     Optional-*

               Rearranges, drops, and reformats records in a file

Name of file to be sorted . . . . . . . . . . . . . . . . . . .

Name of source member containing sort
  specifications . . . . . . . . . . . . . . . . . . . . . . .

Name of file to contain sorted records  . . . . . . . . . . . .

Number of records to be placed
  in output file  . . . . . . . . . . . . . . . . . . 1-8000000              *

Name of library containing source member  . . . . . . . . . .  SORTLIB





Cmd3-Previous menu    Cmd4-Put on job queue

                                                             (c) 1983 IBM Corp.
```

The first three items in this display are required parameters. If you do not enter any or all of them, the system will display the prompt again with a message that a required parameter is missing.

Entering the SORT Procedure Command Statement

Rather than entering the word SORT, you can enter the SORT procedure command statement, as shown in this example:

```
SORT ITEMBALN,EXAMP1,EX1OUT,1000,LIBR1,Y
        1        2       3      4     5   6
```

The SORT procedure command requires the same information as the display that appears when you enter only the word SORT. In the preceding example, the following parameters are present:

1 The name of the file to be sorted

2 The source member containing sort specifications

3 The output file name

4 Number of records the output file will contain

5 Library containing sort specifications

6 Place the sort job on the job queue.

Also, the first three parameters are required. If you do not enter them, the same prompt that appears when you enter only the word sort is displayed with a message that a required parameter is missing.

After you enter the SORT procedure command in either of the preceding ways, the sort program checks all the parameters to ensure their accuracy. If an error is found in a parameter, a message is displayed on the system display screen. If none of the parameters have an error, the system processes the SORT procedure command, which then processes the sort specifications.

SORT Procedure Command Parameters

The SORT procedure command statement consists of items of information called **parameters**, as shown in the following diagram:

```
SORT     input file name,source member name,output file name,

         number of records, ┌                     ┐   ┌ ┐
                            │source member library│ , │N│
                            │current library      │   │Y│
                            └                     ┘   └ ┘
```

Note: The underlined parameters are assumed by the sort program if you do not specify them. For example, if you do not specify the name of a library containing the specifications, the sort program assumes they are in the current library.

The SORT procedure command parameters are:

Input file name: Identifies the file on disk to be sorted. The file may be a remote file if the Distributed Data Management feature is in use. This is a required parameter.

Source member name: Identifies the library source member containing the sort specifications to be used to sort the input file. The source member name is a required parameter.

Output file name: Identifies the file, local or remote, that will contain the sorted records. This name must not be an existing file name. If the Distributed Data Management (DDM) feature is in use, the output file may be a remote file. The output file name is also a required parameter.

Number of records: Indicates the number of records the new output file will contain. This number can be up to 8 000 000. If this parameter is not specified, the SORT procedure command will assume that the output file will contain the same number of records as the input file.

Source member library: Specifies the name of the library containing the source member. If a user library is not specified, the current library is assumed. (See the *System Reference* manual for more information about source members and user libraries.)

N: Specifies that the sort job should not be placed on the job queue. The job will be run from the display station.

Y: Specifies that the sort job should be run from the job queue.

Using Procedures to Run a Sort Job

Using the SORT procedure command is one way to run a sort job. Using your own procedures is another way. A procedure is a group of statements that the system needs to run a program, a job, or a job step. If you're using an OCL procedure rather than the SORT procedure command, your OCL procedure will contain similar information required by the SORT procedure command parameters.

If you want to use a procedure to run a sort job, you must know the OCL statements required to run a sort job. Then you can enter and store the OCL statements in a procedure member on disk. If you use a procedure that is stored on disk, the sort specifications can be contained in either a source member or within the same sort procedure. The primary source of information for the OCL statements is the *System Reference* manual; however, the guide you are reading now contains the general information you will need to write a sort procedure.

Creating a Procedure Using Operation Control Language Statements

You can create a procedure that will run a sort job using operation control language (OCL) statements: LOAD, FILE, RUN, END, and the SOURCE (utility control) statement. If you intend to sort multiple files, you must use OCL statements as the SORT procedure lets you sort only one file at a time.

The following examples show OCL procedures (created using OCL statements) that can run a sort job. The first OCL procedure calls your sort specifications from a source member. The second OCL procedure contains both the OCL statements and the sort specifications.

```
1    // LOAD #GSORT

2    // FILE NAME-INPUT,LABEL-ITEMBALN

3    // FILE NAME-WORK,LABEL-BALANCE,BLOCKS-50,RETAIN-S

4    // FILE
     NAME-OUTPUT,LABEL-BALOUT,RETAIN-T,DISP-NEW,RECORDS-300

5    // RUN

6    // SOURCE ITEMSORT,ITEMLIB

7    // END
```

[1]

LOAD statement 1 tells the system to load an IBM-supplied program called #GSORT into the system. The #GSORT program is the system's sort program.

FILE statement 2 tells the sort program that the name of your input file to be sorted is ITEMBALN.

FILE statement 3 tells the sort program that a 50-block area (500 disk sectors) with the label of BALANCE will be used as the sort work area. RETAIN-S means that the work file is a scratch file and will be deleted after the sort is completed.

FILE statement 4 identifies the output file as a new resident file named BALOUT. When this statement is processed, the system reserves disk space for 300 output records.

RUN statement 5 instructs the system to run the #GSORT program.

[1] If the sort is a job step in a procedure containing other sort job steps, and the sort does not contain the // ATTR statement, do not use // END in the sort procedure. See the *System Support Reference Manual* for more information about the ATTR and END OCL statements.

SOURCE statement 6 identifies to the sort program the name of the source member containing the sort specifications for that sort job. This statement indicates that the sort specifications are contained in a source member named ITEMSORT. The ITEMSORT source member is stored in a library named ITEMLIB.

END statement 7 indicates the end of the procedure.

This example OCL procedure contains both the OCL statements and the sort specifications. All procedures are stored in library procedure members.

```
// LOAD #GSORT
// FILE NAME-INPUT,LABEL-ITEMBALN
// FILE NAME-WORK,LABEL-BALANCE,BLOCKS-50,RETAIN-S
// FILE NAME-OUTPUT,LABEL-BALOUT,RETAIN-T,DISP-NEW,RECORDS-300
// RUN
   HSORTR     8A        0    24    N
   I P  19   22LEF   23   26          AVAIL FLD LESS OR EQUAL TO REORD
   FNP  19   22                       AVAIL NUMBER OF UNITS AVAILABLE
   FNP  23   26                       REORD REORDER POINT
   FDC   2    9                       ITNBR ITEM NUMBER
// END
```

The following topics contain more information about some of the OCL statements.

File Information Needed When Writing Your Own OCL Procedures

When you use your own OCL procedure to run a sort job, you must supply the sort program with the following information about the input, work, and output files. You can use the OCL FILE statement to provide this information to the sort program. (This section contains only general file information that is needed to write a sort procedure. More detailed information about the FILE statement is provided in the *System Reference* manual).

The format of the FILE statement is:

```
// FILE    NAME-file name[,UNIT-F1] [,LABEL-file label] [,RECORDS-records]
                                                        [,BLOCKS-blocks  ]


          [,RETAIN-{T}] [,DATE-{mmddyy}] [,DISP-{SHR  }]
          [        {J}] [      {ddmmyy}] [      {SHRMM}]
          [        {S}] [      {yymmdd}] [      {SHRMR}]
                                         [      {SHRRM}]
                                         [      {SHRRR}]
                                         [      {NEW  }]
                                         [      {OLD  }]
```

Note: The FILE statement contains other parameters; however, the parameters shown are the primary ones required for a sort procedure. The DBLOCK-parameter should not be used.

FILE Statement for the Input File: The following is an example of a FILE statement for an input file:

```
// FILE NAME-INPUT,LABEL-ITEMBALN
```

- **NAME-file name:** Specifies the name the program uses to refer to the file. For sort, the file name must be one of the following:

  - INPUT or INPUT1 (but not both)
  - INPUT2
  - INPUT3
  - INPUT4
  - INPUT5
  - INPUT6
  - INPUT7
  - INPUT8

Multiple Input Files

Up to eight files, local or remote, may be used as input to a sort program. The // FILE statements may be entered in any sequence. The input files are processed serially regardless of the order of the // FILE statements. A particular sort could have the // FILE statements in the order of INPUT8, INPUT3, and INPUT5. The order of the sort would be INPUT3, INPUT5, and INPUT8. The lowest numbered file is fully processed first, then the next lowest numbered, and so on to the highest numbered file, until each input file has been processed individually.

Equal control field ordering (H-specification column 12) lets you merge two or more input files. Chapter 7 has more information on the H-specification.

The restrictions for multiple input files are:

- OCL must be used. The SORT procedure can not accept multiple input file information.

- SORTA (addrout sort) is not allowed.

- The lengths of the records in each input file must be the same.

- **LABEL-file label:** Specifies the actual name by which a file is identified on the disk. This is the name by which you identify your input file.

If the LABEL parameter is omitted from a disk FILE statement, the file name from the NAME parameter is used. For example, if you omit the label of your input file and you used INPUT1 as the file name, the sort program will also use INPUT1 as the file LABEL.

Once you name the input file, you must use the LABEL parameter to give the sort program the label of (your name for) your input file. For example, if you want to sort an input file you named ITEMBALN, use the following format:

```
// FILE NAME-INPUT,LABEL-ITEMBALN...
```

FILE statement for the Work File (Optional): The work file statement is optional in many sort jobs and can sometimes be omitted from the OCL procedure; however, you must specify a FILE statement for the sort work file in the following cases:

- If the input file is a shared file to which records are being added

- If the input file is a shared file to which records have been added but which has not been closed.

The work file *must* be a local file. A remote work work file is not allowed.

The following is an example of a FILE statement for a work file:

```
// FILE NAME-WORK,LABEL-ITEMOUT,RECORDS-500
```

- **NAME-file name:** Specifies the name of the sort work file as WORK. The work file name must be WORK.

- **LABEL-file label:** Identifies a name you assign to the work file. For example, your work file statement in your OCL procedure might look like this:

  ```
  // FILE NAME-WORK,LABEL-BALANCE,RECORDS-500
  ```

  Note: Remote work file is not supported if DDM feature is in use.

- **RECORDS or BLOCKS:** Indicates the total number of blocks or records in the work file. One block contains 2560 bytes. The smallest disk file unit that can be reserved is one block. Either RECORDS or BLOCKS, but not both, can appear in the FILE OCL statement. The RECORDS or BLOCKS parameter must be used for a new file. See Appendix A for information about calculating the number of BLOCKS and RECORDS in a file.

- **RETAIN:** Classifies the disk file as a resident (T), scratch (S), or a job (J) file when it is created.

  - **T:** Specifies a **resident** file. A resident file remains on the disk when the job ends. If the RETAIN parameter is omitted from the FILE statement when the file is created, the file is assumed to be a resident file.
  - **S:** Specifies a **scratch** file. A scratch file can be used only by the job step creating it and does not exist after the job step has ended.
  - **J:** Specifies a **job** file. After a job file is created, it can be used by any remaining job steps in a multiple-step procedure. A job file is defined only within the job and does not exist after the job ends.

The work file is usually a scratch file because you usually do not need its information after the sort job (or sort job step in a multiple-step procedure) has been run.

FILE Statement for the Output File: The following is an example of a FILE statement for an output file:

```
// FILE NAME-OUTPUT,LABEL-BALOUT,RECORDS-500,RETAIN-T,DISP-NEW
```

- **NAME-file name:** Indicates the name of the output file. This name must be OUTPUT.

- **LABEL-file label:** Identifies your name for this file to the sort program. In the example, BALOUT is the label for the output file.

- **RECORDS or BLOCKS:** Indicates the total number of blocks or records in the output file. See Appendix A for information on how to calculate the number of BLOCKS and RECORDS for a file.

- **RETAIN:** Indicates output file status. A file built during the sort job can have any of these parameters:

  - **J** (scratch at end of job)
  - **S** (scratch at end of job step)
  - **T** (resident).

  If the work or output file you want to use is an existing resident (RETAIN-T) file, you must either supply the location where the file starts and its originally allocated size, or you must indicate that the file is an existing file by specifying DISP-OLD in your work or output file FILE statement. The output file may be remote.

- **DISP:** (Disposition) specifies that the file is a new file or an old file, or that the file can be shared by other jobs running on the system. The DISP parameter is not allowed if RETAIN-J is specified. If DISP is not specified, the system determines whether a file is new or old based upon whether the file is in the disk volume table of contents.

  - **SHR:** Specifies that the file already exists and can be shared by other programs running on the system. Read, update, delete, and add operations can be performed on the file. SHR is the same as SHRMM.
  - **SHRMM:** Specifies that the program using the file can modify the file (that is, records can be read, updated, deleted, or added). Other programs that are sharing the file can also modify the file. SHRMM is the same as SHR.
  - **SHRMR:** Specifies that only the program using the file can modify the file (that is, records can be read, updated, deleted, or added). Other programs that are sharing the file can only read records from the file.
  - **SHRRM:** Specifies that the program using the file only needs to perform read operations on the file (that is, no records will be updated, deleted, or added). Other programs that are sharing the file can modify the file (that is, they can read, update, delete, or add records to the file).
  - **SHRRR:** Specifies that the program using the file only needs to perform read operations on the file (that is, no records will be updated, deleted, or added). Other programs that are sharing the file can also only read records from the file.
  - **NEW:** Specifies that the file is new. If a file already exists with the same label and creation date as the new file, an error message is displayed. The new file can be created using any disk file organization. The file cannot be shared by any other programs until the program that created it ends.
  - **OLD:** Specifies that the file already exists, and is not to be shared until the program that is using it ends. If the file does not exist, an error message is displayed. DISP-OLD allows you to process an existing file as an output file without having to specify the RECORDS, BLOCKS, or LOCATION parameter. When an existing file is overlaid (that is, the file contains totally new information), the creation date is changed to the job step date; also, DISP-OLD must be specified.

Using the SOURCE Statement in Your Sort Procedure

The // SOURCE statement identifies the location of the source member containing the sort specifications.

The format of the SOURCE statement is:

```
// SOURCE source member name [,user library name]
```

The source member can exist in either the system library or any user library. If the source member is in a user library, the name of that library must be placed in the // SOURCE statement. For example, assume you entered and stored sort specifications in a library you created, one named ITEMLIB. The // SOURCE statement would identify the library like the following:

```
// SOURCE PAYR1,ITEMLIB
```

In the preceding example, PAYR1 is the name of the source member containing the sort specifications and ITEMLIB is the user library where the source member is stored.

If a user library is specified in the // SOURCE statement but is not found on disk, the sort program will search the system library for the source member.

If the user library is not on disk and the source member in the system library is not the one you want, other sort specifications that are not intended to be used for this job might be used by the sort program.

Using the END statement in Your Sort Procedure

Always use an // END OCL statement after the last sort specifications statements in your OCL procedure. Otherwise, undesirable results like the following might occur:

- OCL statements following the last sort specifications statement will be processed as more sort specifications statements.

- The sort program will not print or display any error messages if the // END statement is keyed in at the display station keyboard after the procedure that is running is finished.

If you use the // SOURCE statement to enter your sort specifications into the system, normally a // END statement should not be placed after the // SOURCE statement.

If you want to run sort as one step of a multiple-step procedure and you want to release the display station to continue processing the procedure (using the ATTR OCL statement), after starting the sort job, you must use the // END OCL statement after the // SOURCE statement. You should release the display station. That way, if any errors occur and you choose an option that cancels this job step (usually a 2 or 3), you will not cancel an entire multi-step procedure. See the *System Reference* manual for more information about the // ATTR OCL statement and the // END OCL statement.

Running OCL Procedures Stored on Disk

If you use an OCL procedure that is stored on disk, the sort specifications can be contained in either a source (S) member or within a procedure (P) member on disk.

To run a procedure that is stored on disk, you can enter either of the following:

- The name of the procedure (which is in effect a procedure command)

  For example: *ITEMPROC*

- An // INCLUDE statement:

  ```
  // INCLUDE  procedure name[,library name]
  ```

  For example: *// INCLUDE ITEMPROC,ITEMLIB*

  In the example, **ITEMPROC** is the name of the procedure that will run the sort job and **ITEMLIB** is the name of the library where the procedure is stored.

Chapter 5. Sort Performance Considerations

The amount of time required to run a sort job can vary greatly depending on the following factors:

- Number of input files
- Number of records in the input file(s)
- Order of records in the input file(s)
- Size of the work records
- Number of sort specifications for the sort job
- Location of the input, work, and output files on disk
- Whether or not you use an alternative collating sequence
- System environment in which a sort job is running
- Region size.

The following text describes how each factor affects the amount of time required to run a sort job.

Number of Input Files

The more input files you use in a sort job, the longer it takes to run the job. This occurs because sort requires more time to read numerous files and write their data into the work file, then write the data into the output file, than if you were sorting a fewer number of files. This is especially true if all the files contain a great deal of data. On the other hand, sort will not require as much time to process numerous files with relatively small amounts of data.

Number of Records in the Input File

The number of input records you sort also has a significant impact on the amount of time it takes to run a sort job. For example, it would not take as long to sort a file containing a few input records as it would take to sort a file containing several hundred records. If possible, select only those records that must be sorted.

Therefore, for a more efficient sort run time, omit records from a sort job whenever possible. If there are certain records in a file that you do not want written into the output file, use an omit statement so the sort program will not use part of its run time to check these records. This way only the records to be sorted will be read into the work file and the output file and the job will complete its run within a shorter period of time.

Order of Records in the Input File

The order of the records in the input file can affect how long it takes a sort job to run. Suppose that two files have about the same number of records to be sorted. Suppose also that the two files contain the same information; however, the first file has more records in the sequence you want than the second file. The records in the first file can usually be sorted faster than those in the second file if the same sort specifications are used to sort both files. This additional time is needed because it takes longer for the sort program to change the order of the additional data, then write it into the output file.

Work Record Size

The larger the input records, the more data the sort program must move into the work area when processing the records. Therefore, the more data sort moves into the work record, the longer it takes to sort a file. The sort program builds a work record for each input record included in the job. Therefore, if program run time is important to you, do not include unnecessary fields in your sort job. Every field you do not include decreases the size of the record by the length of that field.

For example, assume you want to sort a file containing records that are 156 characters long. These records contain information such as customer names, customer numbers, customer addresses, customer orders, descriptions of items ordered, and other information. You want to produce an alphabetized list of customer names and addresses for address labels. You would only need the customer names and customer addresses, not all the other data. Assuming the customer name field contains 30 characters and the address field contains another 30 characters, the sort program only needs to move a small amount of data from the input records to the work records. Each work record would only contain 60 characters of data, a reduction of 94 characters of data per record. To go even further, if you multiplied 94 times the number of input records, assume 250 input records, (94 X 250 = 22 500 characters), the amount of time saved would be very significant.

Number of Sort Specifications

The more sort specifications you use, the less main storage space will be available for records. The sort specifications are source data you must enter into the system for processing. This source data uses main storage space as it is processed during a sort job. Because the sorting process also uses main storage space, it is important to allow as much main storage space as possible for the sorting process. Therefore, the fewer sort specifications you use in a sort job, the more main storage space will be available for the sorting process. Consider using sort specifications to exclude records you do not want to sort.

Disk Location of the Sort Files

When you run a sort job at the same time you run other jobs on the system, the performance effect of the disk location of the files used in the sort job is difficult to predict. However, if you run a sort job and it is the only job running on the system, you can specify the placement of the input, work, and output files such that they are sorted faster. To specify file placement on disk, use the FILE statement. See the *System Reference* manual for more information about file placement on disk.

For the best performance, place the sort work file on a different disk than the one containing the input and output files. If you cannot place the files on different disks, allocate the sort work file as close to the input and output files as possible to prevent the system from doing an extensive search for the files.

Using remote files will increase the time used to run the sort program. The input and output files may reside on a remote system. Sort ensures the work files are local files. See the *Distributed Data Management Guide*, SC21-8011, for more information on remote files.

Alternative Collating Sequence

Using an alternative collating sequence increases the time it takes to run a sort job (see *Specifying an Alternative Collating Sequence* in Appendix B for more information). The sort program uses an additional 375 bytes of main storage space for the sort job when you use an alternative collating sequence. The size of the input file and work file buffers is thus shortened accordingly. Also, when you use an alternative collating sequence, the sort program must perform more data movements to compare the records. Therefore, it takes longer to run the job.

System Environment in Which a Sort Job Executes

Two identical sort jobs operating in equal size regions may have different run times, depending on their respective system environments. A sort job that competes for system resources in a multiprogramming environment usually requires more running time than the same sort job would require if it were the only job running on the system. A multiprogramming environment is one in which more than one job at a time is running.

Region Size

You can increase the size of the region that the sort program uses whenever you have enough main storage available to do so. A region is an amount of storage space on the system that is used for running programs and jobs. The larger regions usually allow the sort program to process more records per disk access, which usually reduces the time required to sort records in a file. The program requires a minimum of 18 K bytes to sort a file. For information about how to increase the region size, see the REGION statement in the *System Reference* manual.

In a multiprogramming environment, increasing the region size may cause more swapping of programs to occur, which increases disk accesses. Therefore, it may be difficult to determine if increasing the region size will improve performance.

Chapter 6. Sort Job Examples

This chapter contains sort job examples that show how you can use and define a variety of functions provided by the sort program. Each example includes:

- An explanation of the purpose of the job
- An example input file
- Defined specification statements
- A discussion of the specification statement entries
- An example output file.

Example 1 of a Sort Job. Sort the Records in ITEMMSTR File in Ascending Order

Assume your company is taking inventory and you are asked to submit a report listing all the items ordered by customers and a description of each item. First you want to sort the records by the number of each item. Then you want to print the report using a high-level programming language, such as RPG, or use the Data File Utility (DFU). (In some of the following examples, the record code field is shown in the record format description only, not in the examples of input and output records.)

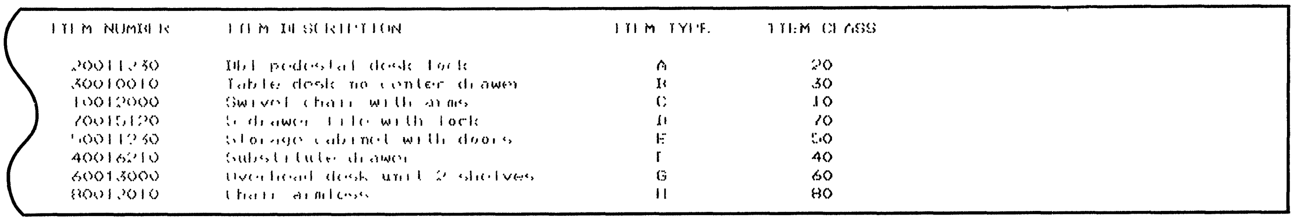
This example shows how to sort an input file containing the customer orders to generate regularly sorted output.

The input file used, ITEMMSTR, is sorted in ascending order (low to high) with the lowest item numbers written into the output file first. The items are sorted using item number (in positions 2 through 9 of the input records) as the control field. The records in the input file all have the same format, and all the data in the input file will be written into the output file.

In this example, although the entire sample job is defined on one sort specifications form, the form is divided into its three parts to allow for explanations of some entries.

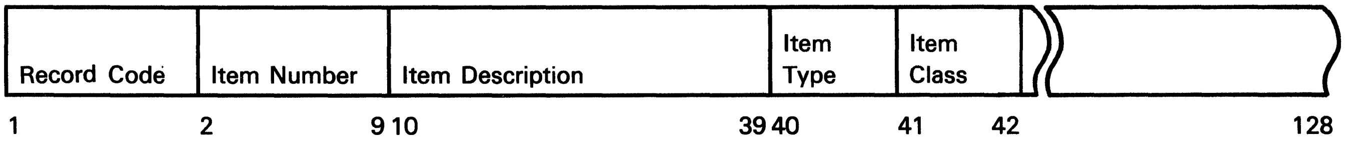
Input

The following file was used as input to the job:



```
ITEM NUMBER    ITEM DESCRIPTION                  ITEM TYPE    ITEM CLASS

  20011230     Dbl pedestal desk lock                A            20
  30010010     Table desk no center drawer           B            30
  10012000     Swivel chair with arms                C            10
  70015120     5 drawer file with lock               D            70
  50011260     Storage cabinet with doors            E            50
  40016210     Substitute drawer                     F            40
  60013000     Overhead desk unit 2 shelves          G            60
  80012010     Chair armless                         H            80
```

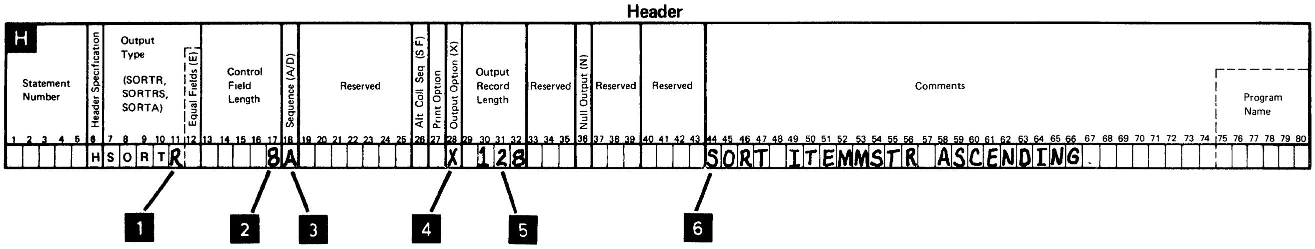
The records in the input file have the following format:



Header Specification Entries

To sort ITEMMSTR as previously described, the sort header specification is defined as follows:

Header



Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
1 2 3 4 5	6	7 8 9 10 11	12	13 14 15 16 17	18	19 20 21 22 23 24 25	26	27	28	29 30 31 32	33 34 35	36	37 38 39	40 41 42 43	44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	75 76 77 78 79 80
	H	SORTR		8	A				X	128					SORT ITEMMSTR ASCENDING	

1 Identifies the job to the system sort program as a regular sort job.

2 Specifies the length of the control field. In this example, item number, which is in positions 2 through 9 of the input records, contains 8 characters.

3 Specifies the order in which the records will be sorted. In this example, the item numbers are sorted in ascending (A) order.

4 Specifies that the information in the control fields should be dropped, not written into the output file. You can, however, specify that the data in the control field be written into the output file by specifying the control field as a data field in a field specifications statement.

5 Identifies the number of characters the output records will contain in the output file. The output records will be the same length as the input records, 128 characters long.

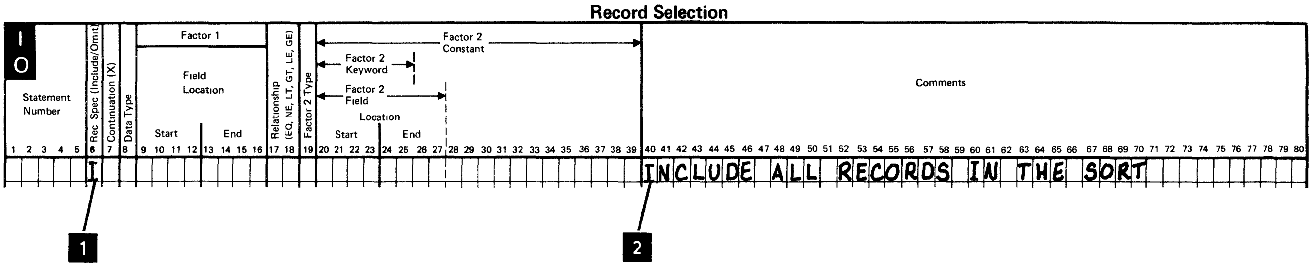
Although the control field is dropped in column 28, its contents are specified as a data field in *Statement 2* of the field specifications statements. Therefore, its contents are written into the output file.

6 Provides a record of why and how the input file is sorted. This comment states that the input file, ITEMMSTR, is used in this job, and that the data will be sorted in ascending order, then written into the output file. You can use any characters in comments except the plus sign (+) cannot be the last character in any comments.

Record Selection Entries

This job, which does not require record selection specifications, is sometimes called an implied include-all, because all records in the input file will be included in the sort.

You can either omit the record selection specifications or define one like the following:



I INCLUDE ALL RECORDS IN THE SORT

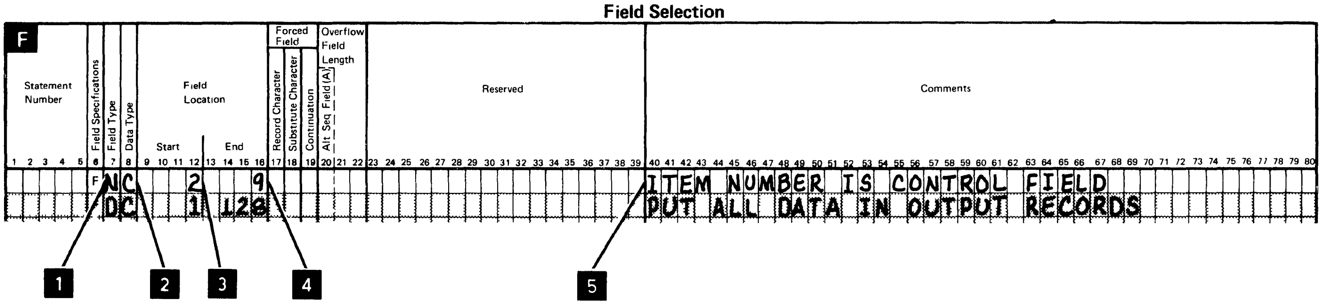
1 Specifies that the sort program should sort all the records in the input file.

2 Is only a comment that has no effect on the outcome of the sort job, unless a plus sign (+) is the last character in a specification statement. 'Comments are simply a method of documenting your job. You can use any characters in the comments fields to document your sort job, except that if the plus sign is the last character in any specification statement, unpredictable results will occur.

Field Selection Entries

There are two field specifications in this sort job. The first is designated as Statement 1, the second as Statement 2.

Field Selection



Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Alt Seq Field (A)	Overflow Field Length	Reserved	Comments
1 2 3 4 5	6	7	8	9 10 11 12	13 14 15 16	17	18	19	20	21 22	23–39	40–80
	F	N	C	2	9							ITEM NUMBER IS CONTROL FIELD
		D	C	1	128							PUT ALL DATA IN OUTPUT RECORDS

Statement 1

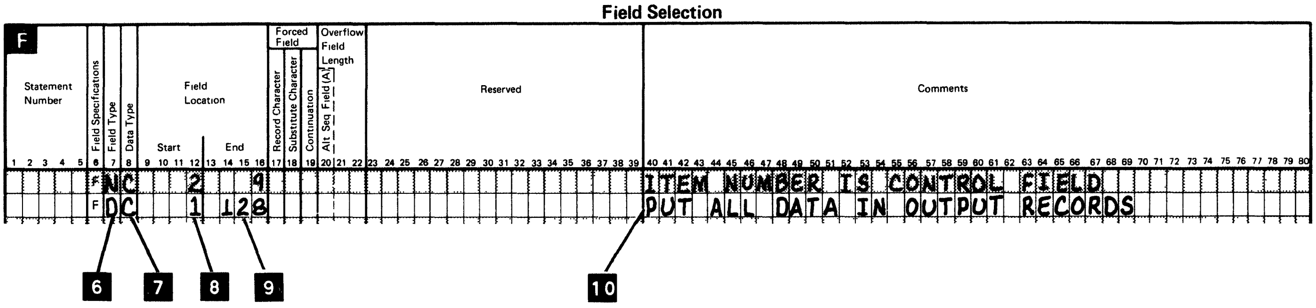
1 Identifies the field as a control field. In this example, the control field is a normal control field.

2 Specifies that the sort program should interpret the data in the control field as character data where both the zone and digit portions are compared.

3, **4** Identify the location of the control field. The starting position in the input record is 2 and the ending position of the control field in the input record is 9. The sort program will compare the zone and digit portions of each character in positions 2 through 9 to determine where its order in the output file should be.

5 Are comments about the sort job. Comments do not affect the outcome of the job, but only serve as reminders. You can use any characters in comments, except that a plus sign (+) cannot be the last character in any comments.

Statement 2



```
FNC   2   9                         ITEM NUMBER IS CONTROL FIELD
FDC   1 128                         PUT ALL DATA IN OUTPUT RECORDS
```

6 Specifies that the fields in positions 1 through 128 should be data fields in the output file.

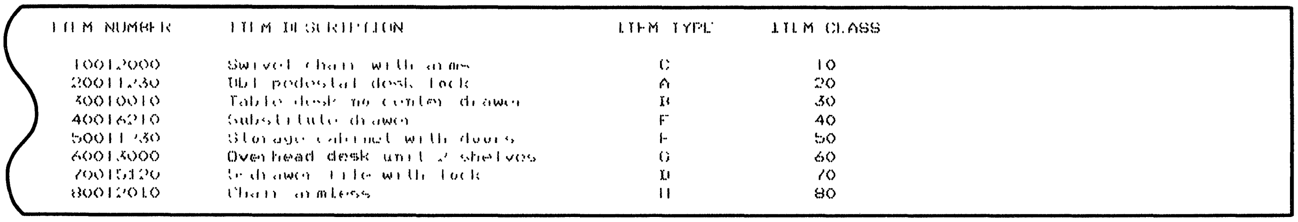
7 Specifies that the data field will contain character data or both the zone and digit portions of characters.

8 **9** Identify the location of the fields in the input record that will be considered data fields in the output file. All the data in the input records will be written into the output file.

10 Is simply a statement that documents facts about the sort job. They have no effect on the outcome of the sort job, unless the plus sign (+) is the last character in a specifications statement. If it is, unpredictable results will occur.

Output

The sorted file would look like the following:



```
 ITEM NUMBER     ITEM DESCRIPTION                     ITEM TYPE    ITEM CLASS

   10012000      Swivel chair with arms                   C            10
   20011230      Dbl pedestal desk lock                   A            20
   30010010      Table desk no center drawer              B            30
   40016210      Substitute drawer                        F            40
   50011230      Storage cabinet with doors               E            50
   60013000      Overhead desk unit 2 shelves             G            60
   70015120      5 drawer file with lock                  D            70
   80012010      Chair armless                            H            80
```

Example 2 of a Sort Job. Sort a File and Put Selected Fields in the Output Records

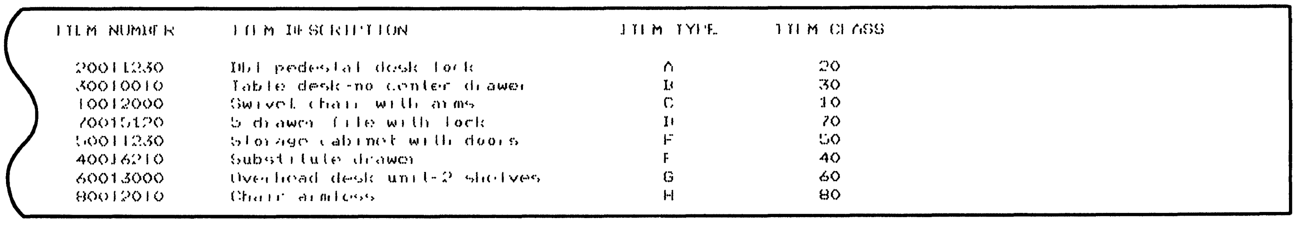
Assume in this example that your business sells office furniture. During the year, one of your suppliers reclassified some items and changed the descriptions of others. You must update the master file containing this information.

Before you start updating the file, you want to sort it and print a report showing only the number of each item, its description, and its class. Example 2 explains how the master file will be sorted.

In example 2, the same file, ITEMMSTR, is sorted in ascending order using item number (ITNBR) as the control field. Because this job will generate an output file containing only specified fields of data, the selected fields must be described in the field specifications.

Input

The following input file was used (item number is an 8-digit field located in positions 2 through 9 of the input records).



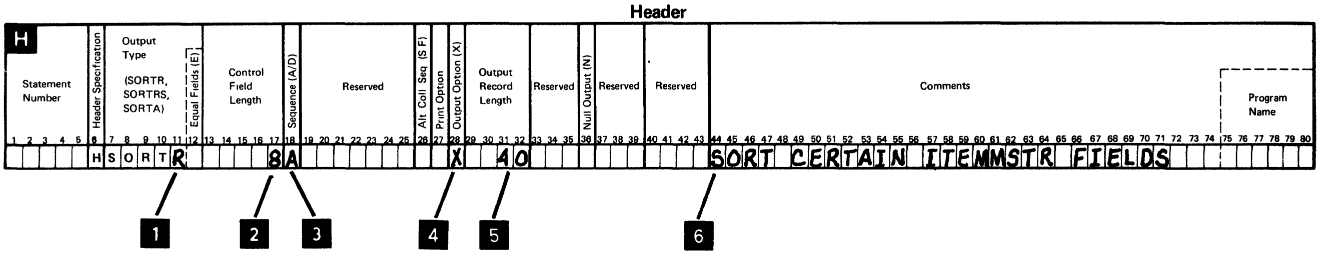
```
ITEM NUMBER     ITEM DESCRIPTION                    ITEM TYPE    ITEM CLASS

  20011230      Dbl pedestal desk lock                  A            20
  30010010      Table desk-no center drawer             B            30
  10012000      Swivel chair with arms                  C            10
  70015120      5 drawer file with lock                 D            70
  50011230      Storage cabinet with doors              E            50
  40016210      Substitute drawer                       F            40
  60013000      Overhead desk unit-2 shelves            G            60
  80012010      Chair armless                           H            80
```

The records in the input file have the following format:

Record Code	Item Number	Item Description	Item Type	Item Class	Warehouse Stock Location	
1	2 9	10 39	40 41	42	43 47	48 128

Header Specification Entries

To sort this file, the header specification is defined as follows:



Header

Statement Number (1-5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7-11)	Equal Fields (E) (12)	Control Field Length (13-17)	Sequence (A/D) (18)	Reserved (19-25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29-32)	Reserved (33-35)	Null Output (N) (36)	Reserved (37-39)	Reserved (40-43)	Comments (44-74)	Program Name (75-80)
	H	SORTR		8	A				X	40					SORT CERTAIN ITEMMSTR FIELDS	

1 Identifies the job to the system sort program as a regular sort job.

2 Specifies the length of the control field as containing 8 characters.

3 Indicates that the input records will be put in ascending order when they are written into the work and output files.

4 Specifies that the information in the control fields should be dropped, not written into the output file.

When you keep the control field in a sort job, the data in the control field is written into the output file.

5 Indicates the length of each record written into the output file. Although the input records are 128 bytes long, the output record length is only 40 characters long because only specified fields are selected and included in the output record.

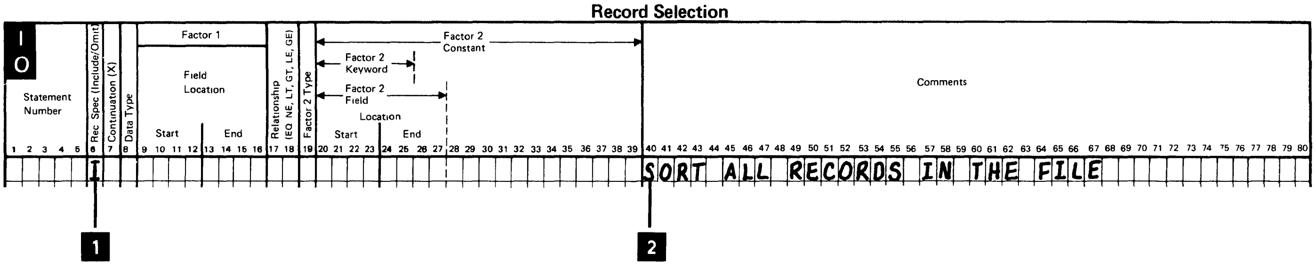
6 Provides documentation about your sort job. Comments have no effect on the outcome of the job, except the plus sign (+) cannot be the last character in any specifications statement. If it is, unpredictable results will occur.

Record Selection Entries

This job, which does not require record selection specifications, is sometimes called an implied include-all, because all records in the input file will be included in the sort.

You can either omit the record selection specifications or define one as follows:

Record Selection



I

SORT ALL RECORDS IN THE FILE

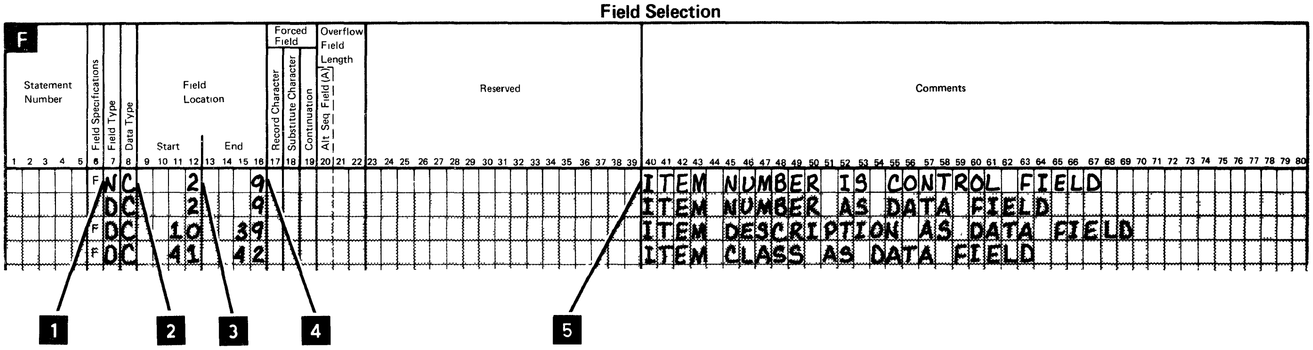
1 Specifies that all records should be included in the sorted output.

2 Indicates that all the records in the input file will be included in the sort.

Field Selection Entries

This job contains four field specifications.

Statement 1



Field Selection

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Record Character	Substitute Character	Continuation	Overflow Field Length	Reserved	Comments
	F	N	C	2	9						ITEM NUMBER IS CONTROL FIELD
		D	C	2	9						ITEM NUMBER AS DATA FIELD
	F	D	C	10	39						ITEM DESCRIPTION AS DATA FIELD
	F	D	C	41	42						ITEM CLASS AS DATA FIELD

F
Forced Field
Overflow Field Length
Alt Seq Field (A)
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
1
2
3
4
5

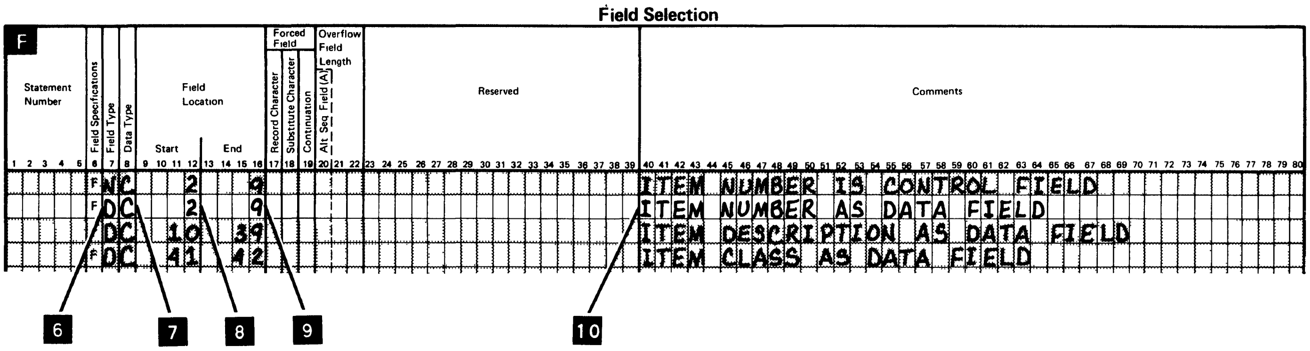
1 Identifies the field as a normal control field.

2 Identifies the data in the control field as character data where both the zone and digit portions will be compared.

3 **4** Specifies the location of the control field within the input records as starting in position 2 and ending in position 9.

5 Documents your sort job. Any characters can be used in comments, except the plus sign (+) cannot be the last character in any specifications statement. If it is, unpredictable results will occur.

Statement 2


Field Selection
F
Statement Number
Field Specifications
Field Type
Data Type
Field Location
Start
End
Forced Field
Record Character
Substitute Character
Continuation
Overflow Field Length
Alt Seq Field (A)
Reserved
Comments
6
7
8
9
10

Statement Number (1–5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9–12)	Field Location End (13–16)	Comments (40–80)
	F	N	C	2	9	ITEM NUMBER IS CONTROL FIELD
	F	D	C	2	9	ITEM NUMBER AS DATA FIELD
	F	D	C	10	39	ITEM DESCRIPTION AS DATA FIELD
	F	D	C	41	42	ITEM CLASS AS DATA FIELD

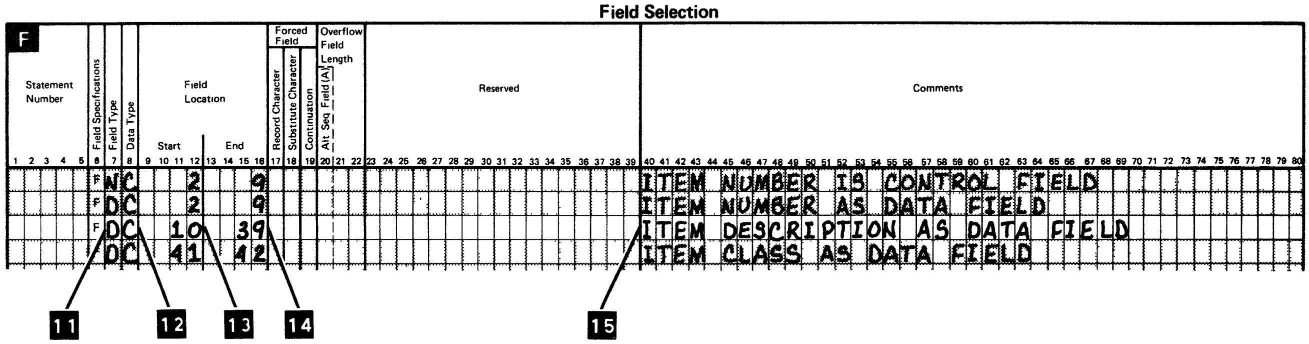
6. Identifies the field as a data field.

7 Specifies that the sort program should interpret the data in this field as character data where both the zone and digit portions are included in the output record.

8, 9 Identify the location of the data field within the input records. Because this is the first data field described in the field specifications (and because the control field data will be dropped, X in column 28 of the header specification), this will be the first field written into the output file (in positions 1 through 8).

10 Identifies comments about the job.

Statement 3


F
11
12
13
14
15

Field Selection

Statement Number (1–5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9–12)	Field Location End (13–16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Alt Seq Field (A) (20)	Overflow Field Length (21–22)	Reserved (23–39)	Comments (40–80)
	F	N	C	2	9							ITEM NUMBER IS CONTROL FIELD
	F	D	C	2	9							ITEM NUMBER AS DATA FIELD
	F	D	C	10	39							ITEM DESCRIPTION AS DATA FIELD
		D	C	41	42							ITEM CLASS AS DATA FIELD

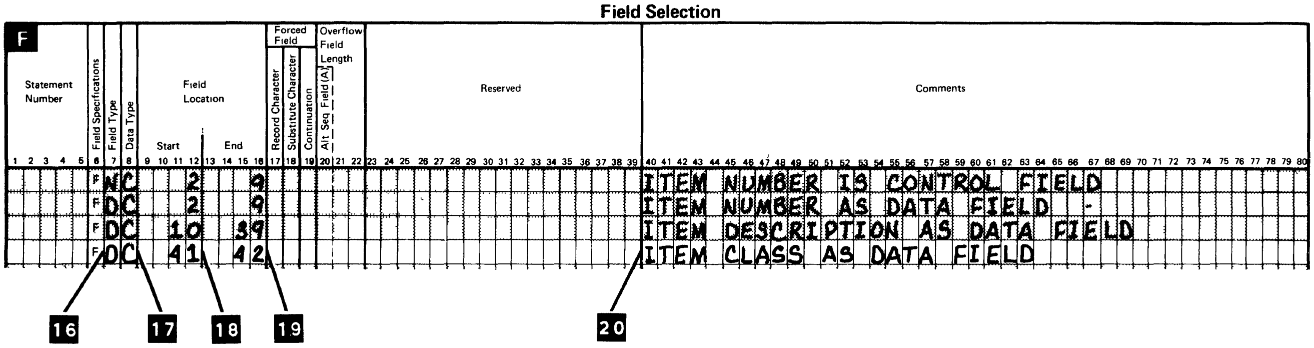
11 Identifies the field as a data field.

12 Specifies that the sort program should interpret the data in this field as character data where both the zone and digit portions are included in the output record.

13 **14** Identify the location of the data field within the input records. The sort program will take the data in positions 10 through 39 of the input records and write it into the output records. The order in which the data fields are described determines their order in the output records. In this example, the data in positions 2 through 9, 10 through 39, and 41, and 42 will be written into the output records in the same order in which they are described in the field specifications.

15 Documents the output for the sort job and explains the abbreviation.

Statement 4



Field Selection

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length Alt Seq Field (A)	Reserved	Comments
	F	N	C	2	9						ITEM NUMBER IS CONTROL FIELD
	F	D	C	2	9						ITEM NUMBER AS DATA FIELD -
	F	D	C	10	39						ITEM DESCRIPTION AS DATA FIELD
	F	D	C	41	42						ITEM CLASS AS DATA FIELD

16 Specifies that the field should be a data field in the output file.

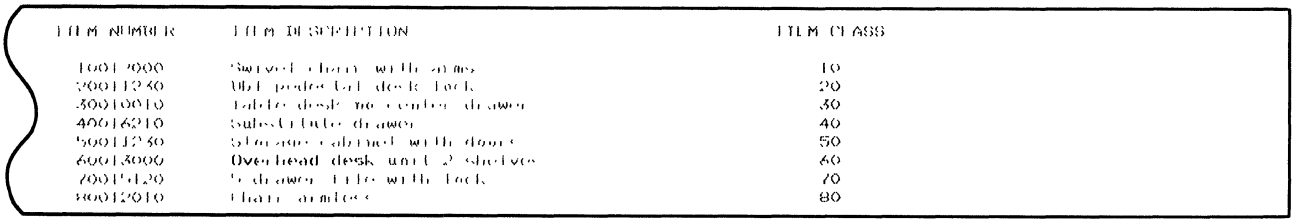
17 Specifies that the data field will contain character data, or both the zone and digit portions of characters.

18 **19** Identify the location of the field in the input record that will be considered a data field in the output file. The data also in positions 41 and 42 of the input records, the item class, will be written into the output file after the item description.

20 Is simply an explanatory statement about the sort job. Comments have no effect on the outcome of the sort job, unless the plus sign (+) is the last character in a specifications statement. If it is, unpredictable results will occur.

Output

The sorted file would look like the following:



```
ITEM NUMBER    ITEM DESCRIPTION                          ITEM CLASS

 10012000      Swivel chair with arms                       10
 20011230      Dbl pedestal desk lock                       20
 30010010      Table desk no center drawer                  30
 40016210      Substitute drawer                            40
 50011230      Storage cabinet with doors                   50
 60013000      Overhead desk unit 2 shelves                 60
 70015120      5 drawer file with lock                      70
 80012010      Chair armless                                80
```

In the output records, item numbers will be placed in positions 1 through 8, descriptions are placed in positions 9 through 38, and item classes are placed in positions 39 and 40.

Example 3 of a Sort Job. Sort a File and Reformat the Output

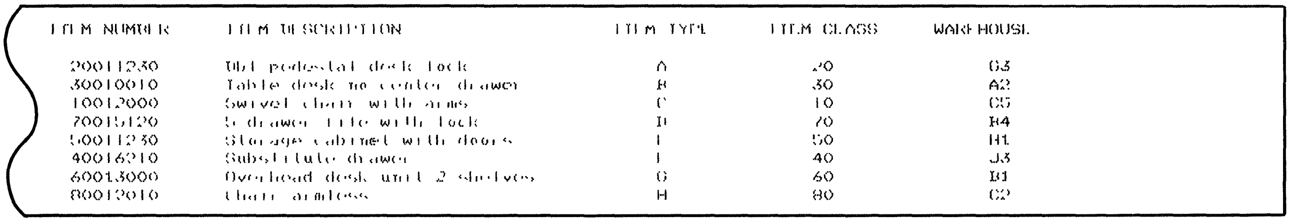
Assume in this example that your business sells office furniture. During the year, one of your suppliers reclassified some items and changed the descriptions of other items. Your job is to update the master file containing this customer order information.

Before updating the file, you want to sort it, reformat the data so the descriptions of the items become the first field written into the output file (and ultimately a printed report listing the sorted records). Example 3 shows how you can do this.

Example 3 shows how to sort an input file and generate an output file containing reformatted data. Included in the output records are the descriptions of each item, its type, its class, and its number.

Input

The following input records are sorted in ascending order with certain data fields reformatted in the output records:



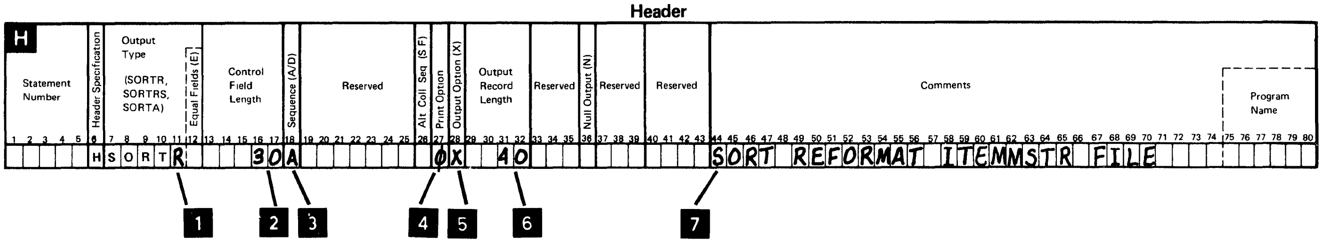
```
  ITEM NUMBER    ITEM DESCRIPTION                  ITEM TYPE    ITEM CLASS    WAREHOUSE

   20011230      Dbl pedestal desk lock                A            20            G3
   30010010      Table desk no center drawer           B            30            A2
   10012000      Swivel chair with arms                C            10            C5
   70015120      5 drawer file with lock               D            70            B4
   50011230      Storage cabinet with doors            E            50            H1
   40016210      Substitute drawer                     F            40            J3
   60013000      Overhead desk unit 2 shelves          G            60            B1
   80012010      Chair armless                         H            80            C2
```

The records in the input file have the following format:

Record Code	Item Number	Item Description	Item Type	Item Class	Warehouse Stock Location	
1	2 9	10 39	40 41	42	43 47	48 128

Header Specification Entries

To sort the input records as previously described, define the sort specifications as follows:



Header

H Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
1 2 3 4 5	6	7 8 9 10 11	12	13 14 15 16 17	18	19 20 21 22 23 24 25	26	27	28	29 30 31 32	33 34 35	36	37 38 39	40 41 42 43	44–74	75 76 77 78 79 80
	H	SORTR		30	A			0	X	40					SORT REFORMAT ITEMMSTR FILE	

1 Indicates that a regularly sorted output file will be generated.

2 Specifies the number of characters in the field in the input records to use to sort the records. In this example, item description, which is in positions 10 through 39 of the input records, contains 30 characters.

3 Specifies the order in which the records will be sorted. In this example, the item descriptions are sorted in ascending (A) order.

4 Specifies that the print option will be a zero. When the print option is a zero, the sort program prints the following information about the job:

- Sort specification statements
- Diagnostic messages indicating any errors encountered in sort specifications
- Program status messages identifying various stages of the job
- Action messages (followed by displayed messages) identifying circumstances requiring attention before the job can continue
- Displayed messages that appear on the display screen.

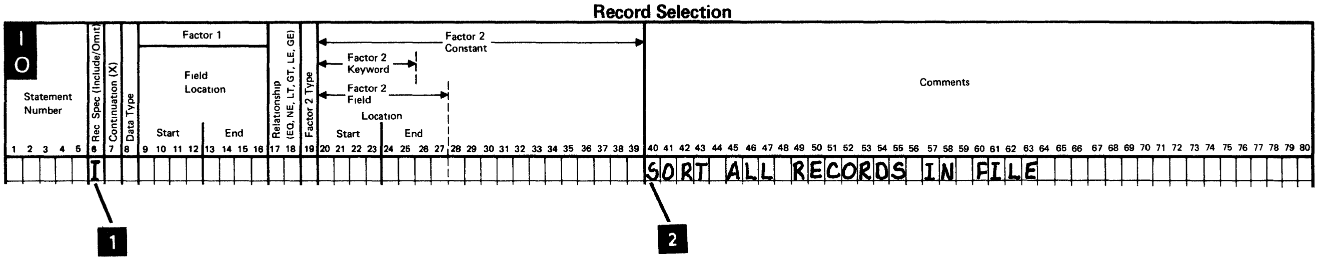
5 Specifies that the information in the control fields will be dropped, not written into the output file.

However, because you want the data in the control field to be written into the output file, you must specify the field again, as a data field on the field specifications (columns 7 through 16). Otherwise, the data in the control field will be written into the output file in its hexadecimal form.

6 Identifies the number of characters the output records will contain in the output file. The output records will contain only specified data fields, therefore the length of the output records will be 40 characters, not 128 characters.

7 Provides a record of why and how the input file is sorted. This comment states that the input file, ITEMMSTR, is used in this job, and that it should be sorted and the data reformatted as specified. You can use any characters in comments, except the plus sign (+) cannot be the last character in any comments.

Record Selection Entries


Record Selection
I
O
Statement Number
Rec Spec (Include/Omit)
Continuation (X)
Data Type
Factor 1
Field Location
Start
End
Relationship (EQ, NE, LT, GT, LE, GE)
Factor 2 Type
Factor 2 Constant
Factor 2 Keyword
Factor 2 Field
Location
Start
End
Comments
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
1
2

I

SORT ALL RECORDS IN FILE

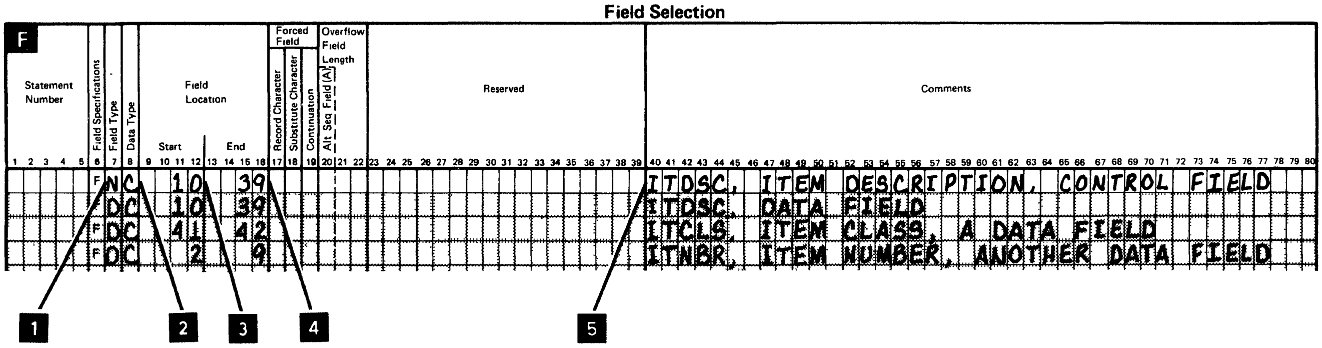
1 Specifies that the sort program should sort all the records in the input file. No other entries are required on the record specification.

2 Is only a comment to help you document your job. You can use any characters in comments, except that the plus sign (+) cannot be the last character in any specifications statement, because it will produce unpredictable results.

Field Selection Entries

There are four field selection statements in this sort job.

Statement 1



Field Selection

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Record Character	Substitute Character	Continuation	Alt Seq Field (A) / Overflow Field Length	Reserved	Comments
	F	N	C	10	39						ITDSC, ITEM DESCRIPTION, CONTROL FIELD
	F	D	C	10	39						ITDSC, DATA FIELD
	F	D	C	41	42						ITCLS, ITEM CLASS, A DATA FIELD
	F	D	C	2	9						ITNBR, ITEM NUMBER, ANOTHER DATA FIELD

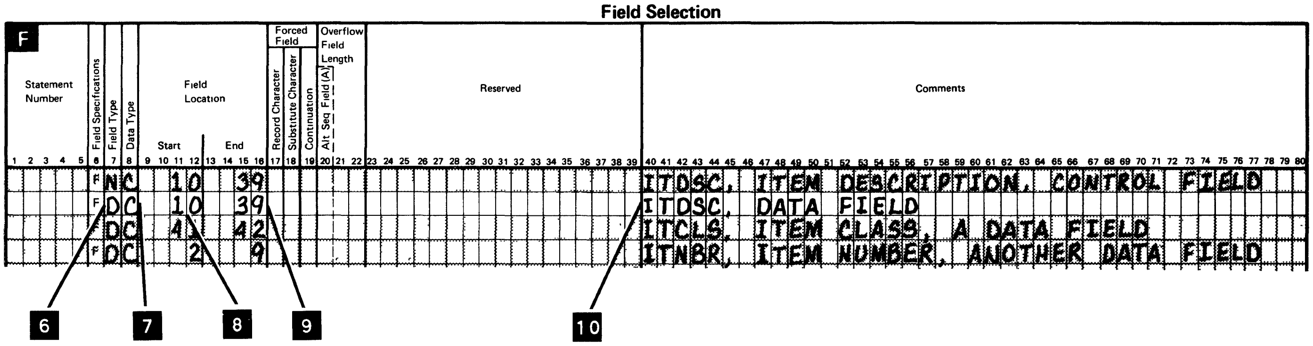
1 Identifies the field as a normal control field. Because column 28 of the header specification contains an X, meaning drop the control field, the description of each item will be specified (as a control first) again as a data field in order for its contents to be written into the output records.

2 Specifies that the sort program should interpret the data in the control field as character data where both the zone and digit portions are compared.

3, **4** Identify the location of the control field. The starting position in the input record is 10 and the ending position of the control field in the input record is 39. The sort program will compare the zone and digit portions of each character in positions 10 through 39 to determine the order of each record in the output file.

5 Is only a comment about the sort job.

Statement 2



Field Selection

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length / Alt Seq Field (A)	Reserved	Comments
	F	N	C	10	39						ITDSC, ITEM DESCRIPTION, CONTROL FIELD
	F	D	C	10	39						ITDSC, DATA FIELD
	F	D	C	41	42						ITCLS, ITEM CLASS, A DATA FIELD
	F	D	C	2	9						ITNBR, ITEM NUMBER, ANOTHER DATA FIELD

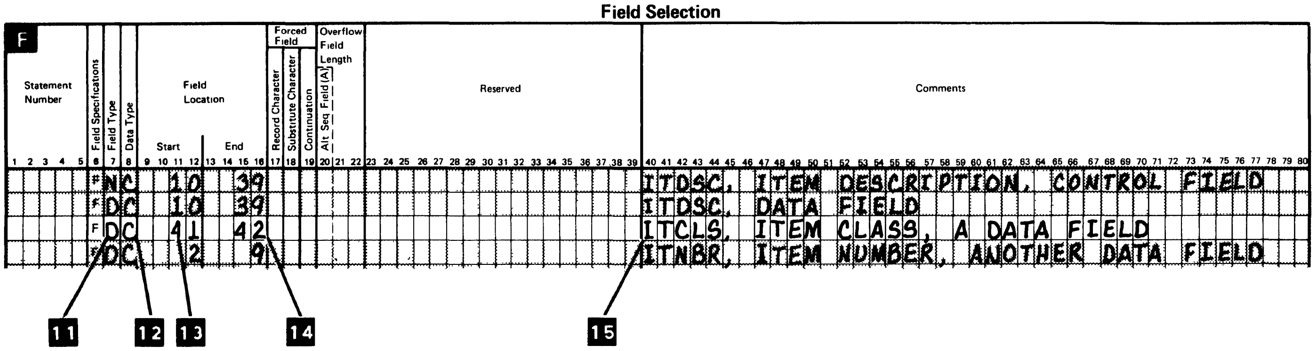
6 Identifies the field as a data field.

7 Specifies that the data field will contain character data, or both the zone and digit portions of characters.

8, 9 Identify the location of the fields in the input record that will be considered data fields in the output file. Position 10 identifies the starting location of the item description and position 39 identifies the ending location of the description.

10 Is only a comment about the sort job.

Statement 3



Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	10	39						ITDSC, ITEM DESCRIPTION, CONTROL FIELD
	F	D	C	10	39						ITDSC, DATA FIELD
	F	D	C	41	42						ITCLS, ITEM CLASS, A DATA FIELD
	F	D	C	2	9						ITNBR, ITEM NUMBER, ANOTHER DATA FIELD

11 Identifies the field as a data field that will be written into the output file.

12 Specifies that the data field will contain character data, or both the zone and digit portions of characters.

13, **14** Identify the location of the fields in the input record that will become data fields in the output file. The data in positions 41 and 42 of the input records, the item class, will be written into the output file after the item description because it is specified after the item description in the field specifications statements.

15 Is simply a statement about the sort job. Comments have no effect on the outcome of the sort job, unless a plus sign (+) appears as the last character in a specification statement.

Statement 4

Field Selection

Statement Number (1-5)	F (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Substitute Character (18)	Continuation (19)	Overflow Field Length Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	10	39						ITDSC, ITEM DESCRIPTION, CONTROL FIELD
	F	D	C	10	39						ITDSC, DATA FIELD
	F	D	C	41	42						ITCLS, ITEM CLASS, A DATA FIELD
	F	D	C	2	9						ITNBR, ITEM NUMBER, ANOTHER DATA FIELD

16 17 18 19 20

16 Specifies that the field should be a data field in the output file.

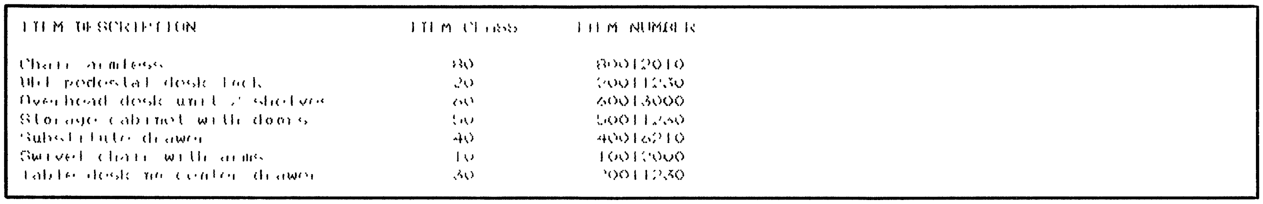
17 Specifies that the data field will contain character data, or both the zone and digit portions of characters.

18, **19** Identify the location of the field in the input record that will be considered a data field in the output file. The data in positions 2 through 9 of the input records, the item numbers, will be written into the output file after the item class, for a total of three data fields in the output records.

20 Is a comment that provides documentation about the sort job.

Output

The sorted output looks as follows:



```
ITEM DESCRIPTION                     ITEM CLASS     ITEM NUMBER

Chair armless                           80           80012010
Wet pedestal desk lock                  20           20011230
Overhead desk unit 2 shelves            60           60013000
Storage cabinet with doors              50           50011260
Substitute drawer                       40           40016210
Swivel chair with arms                  10           10012000
Table desk no center drawer             30           30011230
```

Example 4 of a Sort Job. Sort a File That Will Contain Selected Information

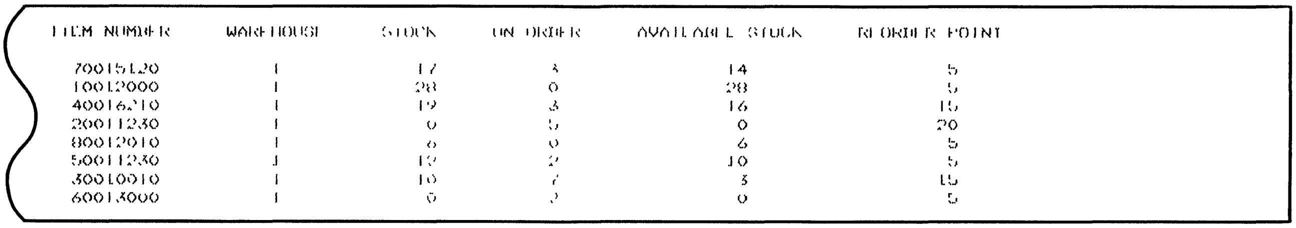
Assume in this example that after taking inventory, you discover you have run out of some items and that you are very low on other items. You need a file containing items requiring reordering. This sort job selects only inventory for stock items that need reordering. The items are items for which the quantity on hand (AVAIL field) is less than or equal to the reorder point (REORD field). The reorder point is the number of items necessary to warrant reordering a supply.

This example shows how to sort an input file containing inventory information to generate an output file containing stock information.

The input file used, ITEMBALN, is sorted in ascending order (with the lowest values written into the output file first). The items are sorted using item number (in positions 2 through 9 of the input records) as the control field. The output file will contain a list of items whose available quantity in stock is equal to or less than the reorder point.

Input

The following file was used as input to the job:



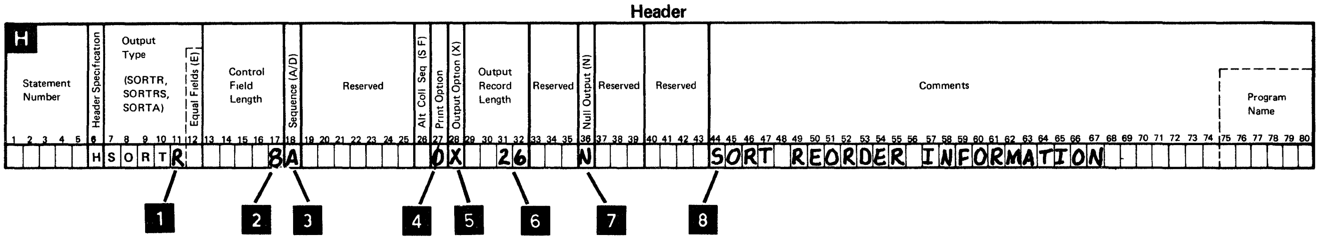
```
ITEM NUMBER   WAREHOUSE   STOCK   ON ORDER   AVAILABLE STOCK   REORDER POINT

 70015120         1         17        3             14                5
 10012000         1         28        0             28                5
 40016210         1         19        3             16               15
 20011230         1          0        5              0               20
 80012010         1          6        0              6                5
 50011230         1         12        2             10                5
 30010010         1         10        7              3               15
 60013000         1          0        2              0                5
```

All the records in the input file have the same format.

Record Code	Item Number	Warehouse	Stock	On Order	Available	Reorder Point
1	2 9	10	11 14	15 18	19 22	23 26

Header Specification Entries

To sort ITEMBALN as previously described, the sort header specification was defined as follows:



Header

H Statement Number (1–5)	Header Specification (6)	Output Type (SORTR, SORTRS, SORTA) (7–12)	Equal Fields (E) (12)	Control Field Length (13–17)	Sequence (A/D) (18)	Reserved (19–25)	Alt Coll Seq (S F) (26)	Print Option (27)	Output Option (X) (28)	Output Record Length (29–32)	Reserved (33–35)	Null Output (N) (36)	Reserved (37–39)	Reserved (40–43)	Comments (44–74)	Program Name (75–80)
	H	SORTR		8	A			0	X	26		N			SORT REORDER INFORMATION	

1 Indicates that a regularly sorted output file will be generated.

2 Specifies the number of characters in the field in the input records used to sort the records. In this example, item number, which is in positions 2 through 9 of the input records, contains 8 characters.

3 Specifies the order in which the records will be sorted. In this example, A means ascending sequence.

4 Specifies that the print option indicator should be set to zero. A zero in column 27 specifies that the sort program will print the following information about the sort job:

- Sort specification statements
- Diagnostic messages indicating any errors encountered in sort specifications
- Program status messages identifying various stages of the job
- Action messages (followed by displayed messages) identifying circumstances requiring attention before the job can continue
- Displayed messages that appear on the display screen.

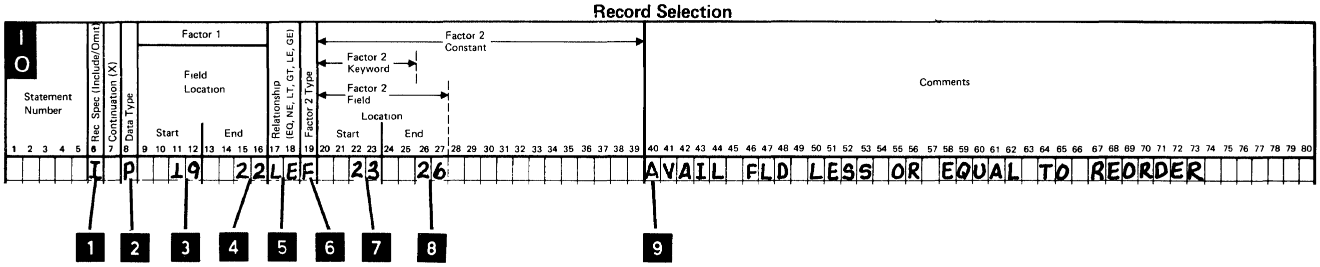
5 Specifies that the information in the control fields should be dropped. The control field is not included as additional data in the output record.

6 Identifies the number of characters the output records will contain in the output file.

7 Specifies that the sort program should create an empty output file without issuing message SORT-7724, *No input records included.* The sort program takes this option only *if none of the input records meet the conditions specified in the record selection statement(s).*

8 Provides a record of why and how the input file is sorted.

Record Selection Entries



```
     I P  19  22LEF  23  26            AVAIL FLD LESS OR EQUAL TO REORDER
```

1 Specifies that the sort program should include only those records whose contents in positions 19 through 22 are less than or equal to the contents of the field in positions 23 through 26, the factor 2 field.

2 Identifies the data in positions 19 through 22 and 23 through 26 as packed data. See *How Sort Compares Characters* earlier in this chapter for more information about packed data.

3, **4** Identify the starting position of the field in the input record to which data will be compared as position 19; the ending position of the field is in position 22 of the input records. In this example, this field is the *available* field.

5 Indicates the relationship that should exist between the data being compared. In this example, the data in the factor 1 location must compare as less than or equal to the data in the factor 2 location.

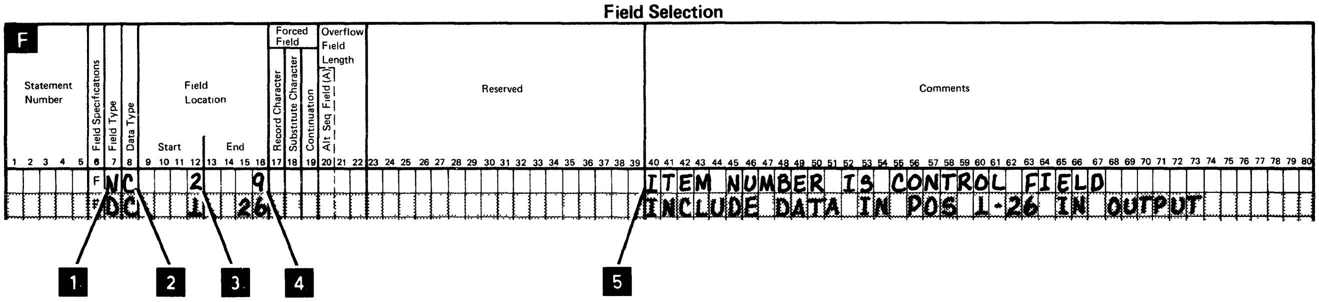
6 Identifies factor 2 as a field. Factor 2 can be another field in the same input record (as in this example), a constant, or a keyword. See *Factor 1 and Factor 2* earlier in this chapter for more information.

7, 8 Identify the location of the factor 2 field in the input record as starting in position 23 and ending in position 26. In this example, the *available* field is compared to the reorder point to determine their relationship.

9 You can use any characters in comments (except that the plus sign (+) cannot be the last character in a specification statement) to document your sort job.

Field Selection Entries

Statement 1



Field Selection

Statement Number (1–5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9–12)	Field Location End (13–16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt Seq Field (A) (20–22)	Reserved (23–39)	Comments (40–80)
	F	N	C	2	9						ITEM NUMBER IS CONTROL FIELD
	F	D	C	1	26						INCLUDE DATA IN POS 1-26 IN OUTPUT

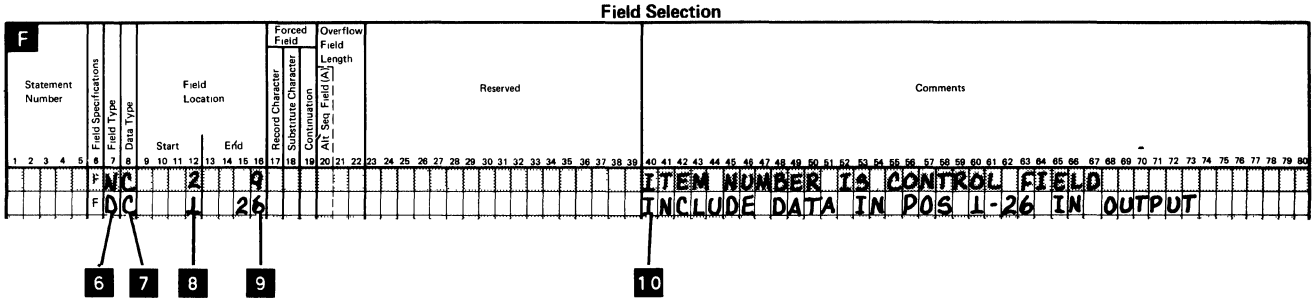
1 Identifies the field as a control field. In this example, the control field is a normal control field.

2 Specifies that the sort program should interpret the data in the control field as character data where both the zone and digit portions are compared.

3, 4 Identify the location of the control field. The starting position in the input record is 2 and the ending position of the control field in the input record is 9. The sort program will compare the zone and digit portions of each character in positions 2 through 9 to determine its order in the output file.

5 Is the documentation for the sort job. Comments do not affect the outcome of the job unless a plus sign (+) appears as the last character in a specification statement, but serve only as reminders.

Statement 2



Field Selection

F Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length Alt Seq Field (A)	Reserved	Comments
	F	N	C	2	9						ITEM NUMBER IS CONTROL FIELD
	F	D	C	1	26						INCLUDE DATA IN POS 1-26 IN OUTPUT

6 Specifies that the field should be a data field in the output file.

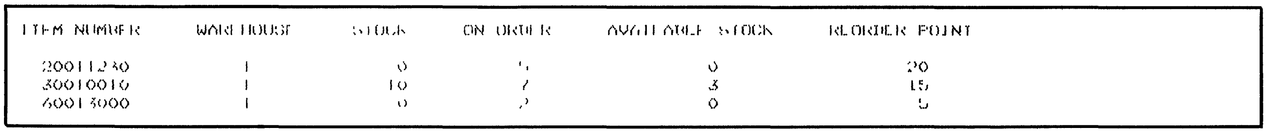
7 Specifies that the data field will contain character data, or both the zone and digit portions of characters.

8, **9** Identify the location of the field in the input record that will be considered a data field in the output file.

10 Are simply statements that document facts about the sort job. They have no effect on the outcome of the job, unless a plus sign (+) is the last character in any specifications statement. If it is, unpredictable results will occur.

Output

Once the records are sorted, the output file would contain the following:



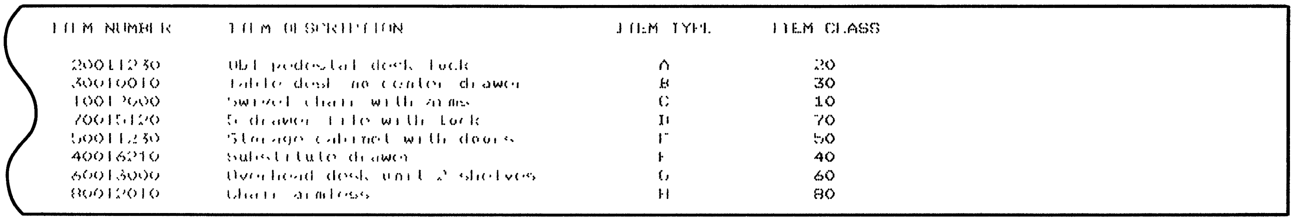
```
ITEM NUMBER    WAREHOUSE    STOCK    ON ORDER    AVAILABLE STOCK    REORDER POINT

  20011230         1           0         5              0                 20
  30010010         1          10         7              3                 15
  60013000         1           0         2              0                  5
```

Example 5 of a Sort Job. Sort a File by Record Address

As in example 3, this example sorts the ITEMMSTR file in ascending order, however, this example uses item class (ITCLS) as the control field. Because this job will generate an output file containing only record addresses, no data fields are specified in the field specifications, only the control field. The advantage of using a record address sort is that you do not have to duplicate all the data in the input file to generate the sorted data. The record addresses only indicates where the real data in the file is located.

Input

Item class is a 2-digit field in positions 41 and 42 of the input records as shown in the following input file:



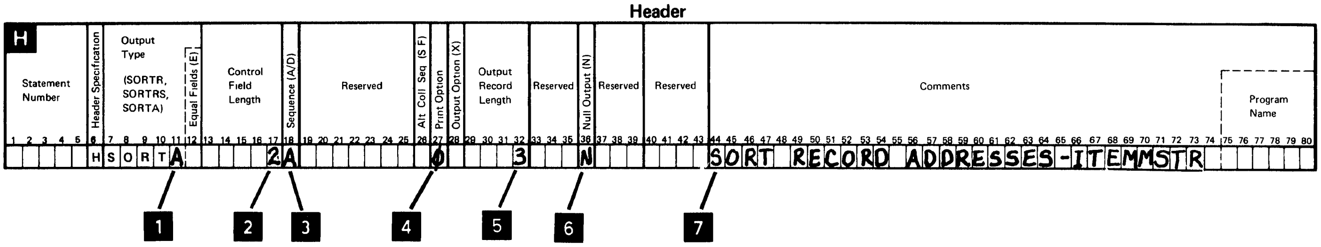
```
ITEM NUMBER     ITEM DESCRIPTION                  ITEM TYPE    ITEM CLASS

 20011230       Dbl pedestal desk lock                A            20
 30010010       Table desk no center drawer           B            30
 10012000       Swivel chair with arms                C            10
 70015120       5 drawer file with lock               D            70
 50011230       Storage cabinet with doors            E            50
 40016210       Substitute drawer                     F            40
 60013000       Overhead desk unit 2 shelves          G            60
 80012010       Chair armless                         H            80
```

The records in the input file have the following format:

Record Code	Item Number	Item Description	Item Type	Item Class	Warehouse Stock Location	
1	2 9	10 39	40	41 42	43 47	48 128

To sort this file, the following sort specifications were defined.

Header Specification Entries



Header

Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
1 2 3 4 5	6	7 8 9 10 11	12	13 14 15 16 17	18	19 20 21 22 23 24 25	26	27	28	29 30 31 32	33 34 35	36	37 38 39	40 41 42 43	44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	75 76 77 78 79 80
	H	SORTA		2	A			Ø		3		N			SORT RECORD ADDRESSES-ITEMSTR	

1 Identifies the job to the system sort program as a record address sort job.

2 Specifies the control field as 2 positions long.

3 Indicates that the input records will be put in ascending order when they are written into the work and output files.

4 Specifies that the print option indicator should be turned on. A zero in column 27 specifies that the sort program will print the following information about the sort job:

- Sort specification statements
- Diagnostic messages indicating any errors encountered in sort specifications
- Program status messages identifying various stages of the job
- Action messages (followed by displayed messages) identifying circumstances requiring attention before the job can continue
- Displayed messages that appear on the display screen.

5 Specifies that this output file will contain only relative record numbers that are 3 characters long.

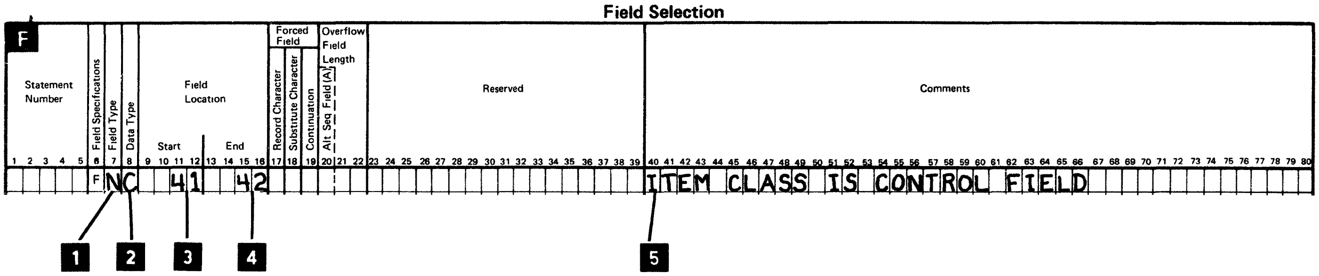
6 Specifies that the sort program should create an empty output file without issuing message SORT-7724, *No input records included.* This option becomes effective only *if no input records meet the conditions specified in the record selection specification statement.*

7 Provides documentation about your sort job.

This job, which does not require record selection specifications, is sometimes called an implied include-all. All records in the input file will be included in the sort.

The job does, however, require field specification statements, which follow.

Field Selection Entries



1 Identifies the field as a normal control field.

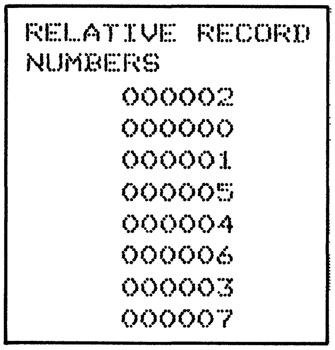
2 Identifies the data in the control field as character data where both the zone and digit portions will be compared.

3, **4** Specify the location of the control field within the input records as starting in position 41 and ending in position 42.

5 Documents your sort job.

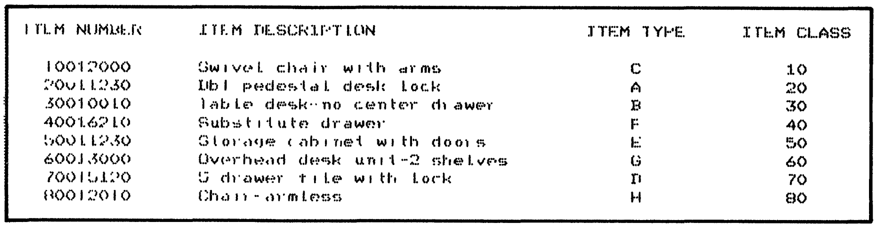
Output

The sorted output looks as follows:



```
RELATIVE RECORD
NUMBERS
          000002
          000000
          000001
          000005
          000004
          000006
          000003
          000007
```

Output



```
ITEM NUMBER   ITEM DESCRIPTION                  ITEM TYPE   ITEM CLASS

 10012000     Swivel chair with arms                C           10
 20011230     Dbl pedestal desk lock                A           20
 30010010     Table desk-no center drawer           B           30
 40016210     Substitute drawer                     F           40
 50011230     Storage cabinet with doors            E           50
 60013000     Overhead desk unit-2 shelves          G           60
 70015120     5 drawer file with lock               D           70
 80012010     Chair-armless                         H           80
```

Note: This is the order of record after using the record address output (SORTA) option.

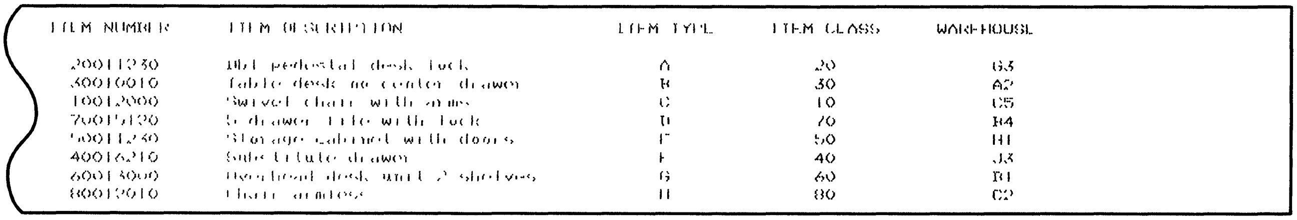
Example 6 of a Sort Job. Generate an Output File Containing Items of Certain Classes

Assume in this example that your company sells office furniture. During the year, your company has decided to discontinue the sale of items in certain classes. You want a sorted file of active items with certain information, item numbers, and item descriptions.

Example 6 shows how to sort an input file and generate an output file containing information about certain items only. This example differs from example 2 in that not all records are sorted and only some fields in the records are written into the output file. Included in the output records are the numbers of each item, its description, and its class.

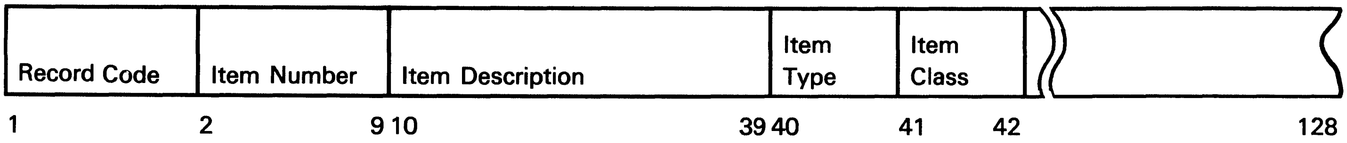
Input

The following input records are sorted in ascending order by item class.



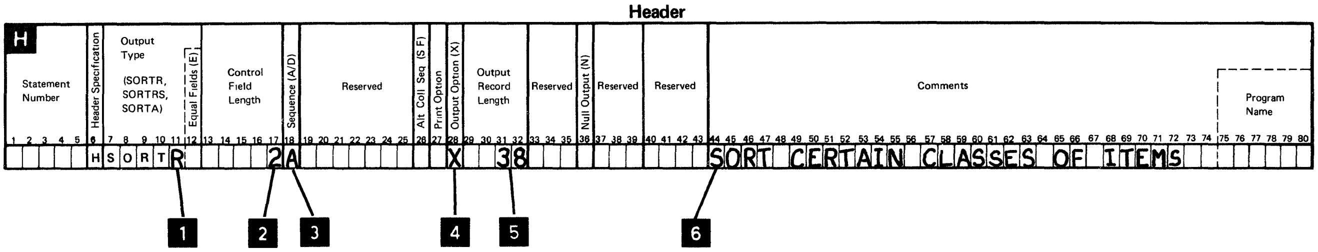
ITEM NUMBER	ITEM DESCRIPTION	ITEM TYPE	ITEM CLASS	WAREHOUSE
20011230	Dbl pedestal desk, lock	A	20	G3
30010010	Table desk no center drawer	B	30	A2
10012000	Swivel chair with arms	C	10	C5
70015120	5 drawer file with lock	D	70	B4
50011240	Storage cabinet with doors	E	50	H1
40016210	Side table drawer	F	40	J3
60013000	Overhead desk unit 2 shelves	G	60	B1
80012010	Chair armless	H	80	C2

The records in the input file have the following format:



Header Specification Entries

To sort the input records, define the sort specifications as follows:


Header
H
Statement Number
Header Specification
Output Type (SORTR, SORTRS, SORTA)
Equal Fields (E)
Control Field Length
Sequence (A/D)
Reserved
Alt Coll Seq (S F)
Print Option
Output Option (X)
Output Record Length
Reserved
Null Output (N)
Reserved
Reserved
Comments
Program Name
1
2
3
4
5
6

Statement Number (1-5)	Header Specification (6)	Output Type (7-11)	Equal Fields (12)	Control Field Length (13-17)	Sequence (18)	Output Option (28)	Output Record Length (29-32)	Comments (44-74)
	H	SORTR		2	A	X	38	SORT CERTAIN CLASSES OF ITEMS

1 Identifies the job to the system sort program as a regular sort job.

2 Specifies the length of the control field. In this example, item class, which is in positions 41 and 42 of the input record, contain only 2 characters.

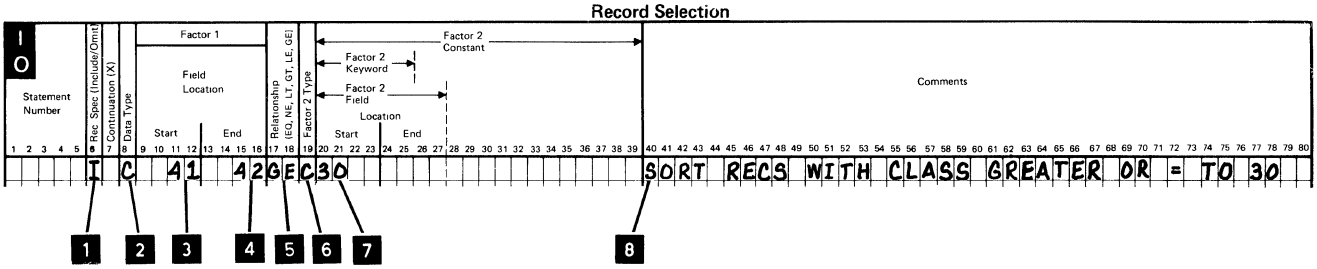
3 Indicates that the work and output records will be placed in ascending order.

4 Specifies that the information in the control fields should be dropped. Control fields are not included as additional data in the output records.

5 Specifies the length of the output records. This length does not include the data in the control field (item class — ITCLS). The entry in column 28 specifies that the control field will be dropped.

6 Is a comment indicating which file this job sorts, ITEMMSTR. This comment has no effect on the outcome of this sort job, unless you have the plus sign (+) as the last character in a specification statement, in which case unpredictable results will occur.

Record Selection Entries



```
     I C 41   42GEC30                SORT RECS WITH CLASS GREATER OR = TO 30
```

1 Specifies that the sort program should select and sort only those records whose contents in positions 41 and 42 are greater than or equal to the factor 2 constant, 30.

2 Identifies the data in positions 41 and 42 as character data whose zone and digit will be compared.

3, **4** Identify the starting position of the field in the input record to which data will be compared as position 41; the ending position of the field is in position 42 of the input records.

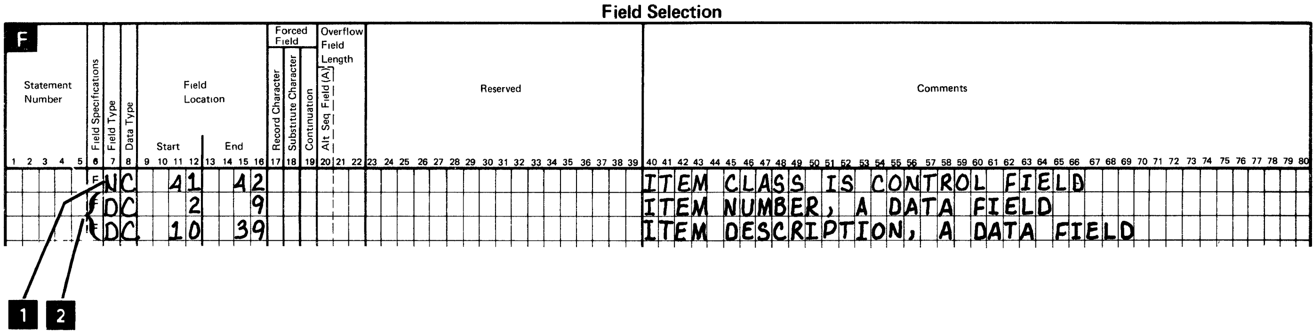
5 Indicates the relationship that should exist between the data being compared. In this example, the data in the factor 1 location must compare as greater than or equal to the constant in the factor 2 location.

6 Identifies factor 2 as a constant. Factor 2 can be another field in the same input record, a constant (as in this example), or a keyword. See *Factor 1 and Factor 2* in Chapter 2 for more information.

7 Identifies the constant data to be used in the compare as '30'.

8 Is a comment about the job. You can use any characters in comments (except that the plus sign (+) cannot be the last character in a specifications statement) to document your sort job.

Field Selection Entries



Field Selection

F

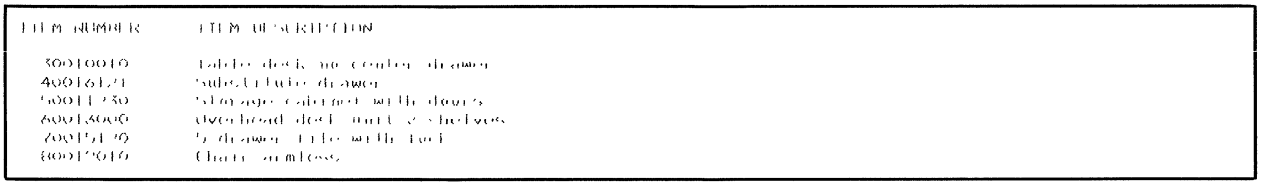
Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field: Record Character (17)	Forced Field: Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	N	C	41	42						ITEM CLASS IS CONTROL FIELD
	F	D	C	2	9						ITEM NUMBER, A DATA FIELD
	F	D	C	10	39						ITEM DESCRIPTION, A DATA FIELD

1 The control field is specified as a normal control field whose contents should be compared as whole characters, and is located in positions 41 and 42 of the input records. A comment in columns 40 through 74 indicates the name of the control field as ITCLS (item class).

2 Two data fields are included in the output records: ITNBR (item number) and ITDSC (item description). The field location columns indicate where in the input records each field is found. Their order on the field specifications indicate their placement in the output records. Item number will be written into the output file first, then the description of each item, and lastly the class of each item will be put in the output records.

Output

Once the output records are sorted, the output file would look as follows:



```
ITEM NUMBER      ITEM DESCRIPTION

 30010010        Table desk, no center drawer
 40016121        Substitute drawer
 50011230        Storage cabinet with doors
 60016000        Overhead desk unit 2 shelves
 70015190        5 drawer file with lock
 80015010        Chair armless
```

The item numbers will be written into positions 1 through 8 of the output file and the descriptions will be written into positions 9 through 38.

Note: The sort program does not print data. Therefore, to print any sorted records, you must use an RPG print program, the Data File Utility print option (LIST), the LISTDATA print procedure, or another print program.

Example 7 of a Sort Job. Sort a File Containing Multiple Record Types

In this example, assume you use an order entry application containiñg a transaction file with multiple record types. You want to sort all the records in the file in descending order by the quantities ordered and at the same time group the records by record type. You want to use the output records to produce a history of the activity of various items.

Example 7 shows how you can sort a file containing multiple record types and group each type of record in the output file. The input file contains three types of records:

- Delivery records containing data about which items were delivered to customers

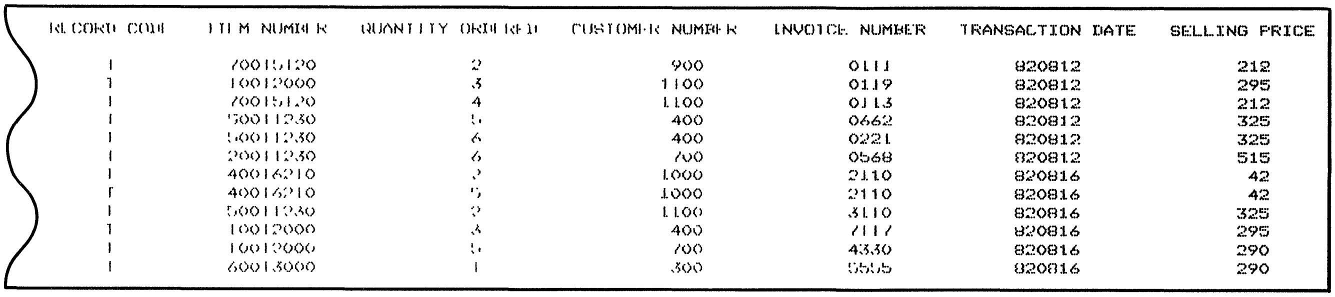
- Receipt records containing data about which items were received by customers

- Adjustment records containing data about which items required any type of adjustment, such as items with duplicate orders, or improperly delivered items.

  The input file contains two types of adjustment records: records containing short items in stock and records containing additions to stock. Any records identifying short items in stock have a 2 in position 11. Any records identifying additional items in stock have a 1 in position 11 of each record.

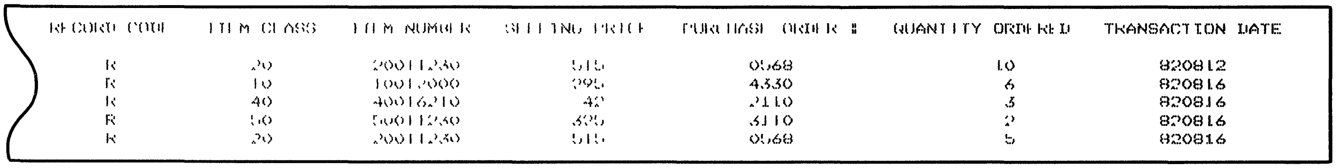
Input

The following input records were sorted so that all receipt, delivery, and adjustment records were grouped together in the output file. The records will be sorted in descending order by the quantity ordered. (Although delete codes are shown in the record format, they will not be shown in the input or output records.)



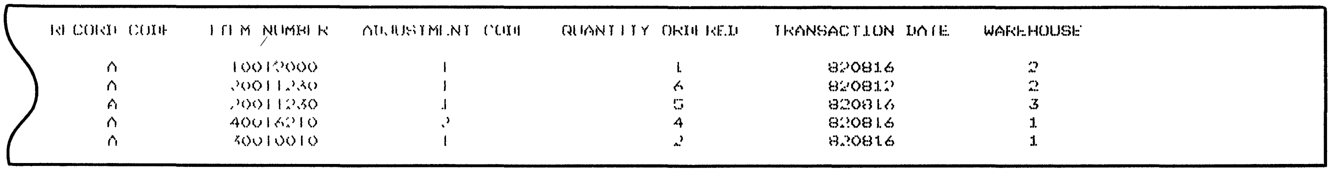
```
RECORD CODE   ITEM NUMBER   QUANTITY ORDERED   CUSTOMER NUMBER   INVOICE NUMBER   TRANSACTION DATE   SELLING PRICE

     I         70015120            2                  900             0111             820812               212
     I         10012000            3                 1100             0119             820812               295
     I         70015120            4                 1100             0113             820812               212
     I         50011230            5                  400             0662             820812               325
     I         50011230            6                  400             0221             820812               325
     I         20011230            6                  700             0568             820812               515
     I         40016210            2                 1000             2110             820816                42
     I         40016210            5                 1000             2110             820816                42
     I         50011230            2                 1100             3110             820816               325
     I         10012000            3                  400             7117             820816               295
     I         10012000            5                  700             4330             820816               290
     I         60013000            1                  300             5555             820816               290
```

Delete Code	Record Code	Item Number	Quantity Ordered	Customer Number	Invoice Number	Transaction Date	Selling Price		
1	2	3 10	11 16	17 24	25 32	33 38	39 43		49



```
RECORD CODE   ITEM CLASS   ITEM NUMBER   SELLING PRICE   PURCHASE ORDER #   QUANTITY ORDERED   TRANSACTION DATE

     R            20         20011230         515              0568                10               820812
     R            10         10012000         295              4330                 6               820816
     R            40         40016210          42              2110                 3               820816
     R            50         50011230         325              3110                 2               820816
     R            20         20011230         515              0568                 5               820816
```

Delete Code	Record Code	Item Class	Item Number	Selling Price	Purchase Order Number	Quantity Ordered	Transaction Date		Blank
1	2	3 4	5 12	13 17	18 21	22 27	28 33		49



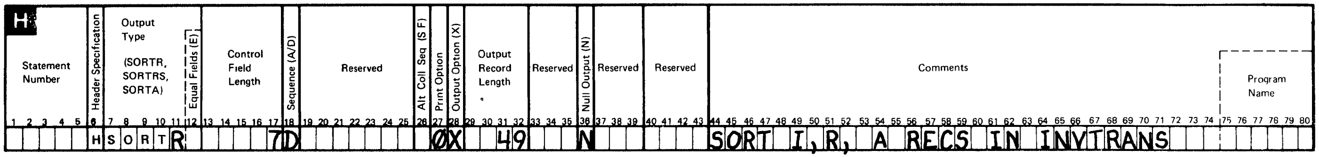
```
RECORD CODE   ITEM NUMBER   ADJUSTMENT CODE   QUANTITY ORDERED   TRANSACTION DATE   WAREHOUSE

     A         10012000            1                  1               820816             2
     A         20011230            1                  6               820812             2
     A         20011230            1                  5               820816             3
     A         40016210            2                  4               820816             1
     A         30010010            1                  2               820816             1
```

Delete Code	Record Code	Item Number	Adjustment Code	Quantity Ordered	Transaction Date	Warehouse	Blank
1	2	3 10	11	12 17	18 23	24	25 49

To sort the records as described, the following three sets of sort specifications were defined:

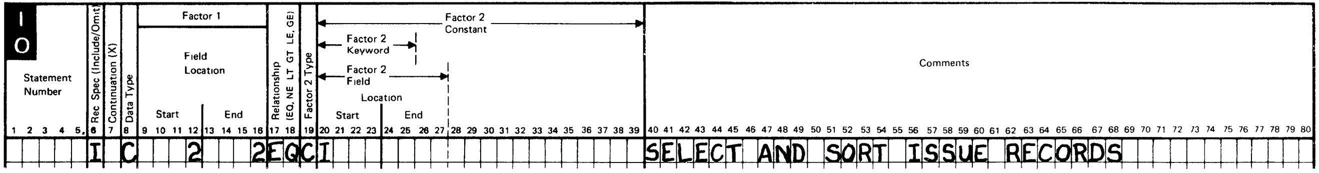
Set 1 describes how the issue records will be sorted.

Header



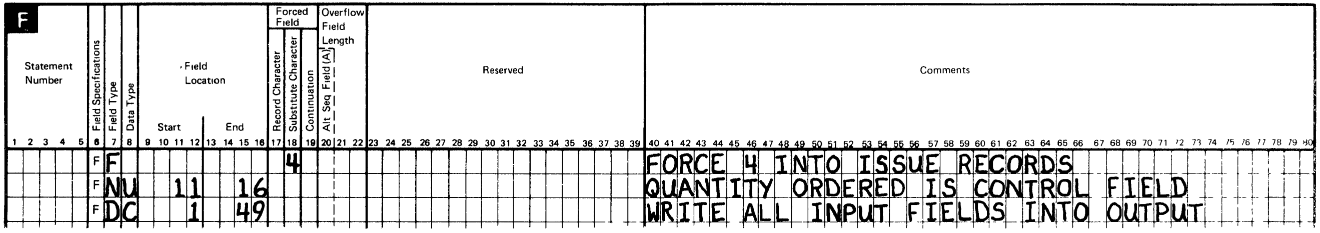
```
     HSORTR     7D        ØX  49   N       SORT I,R, A RECS IN INVTRANS
```

Record Selection



```
     I C   2   2EQCI                   SELECT AND SORT ISSUE RECORDS
```

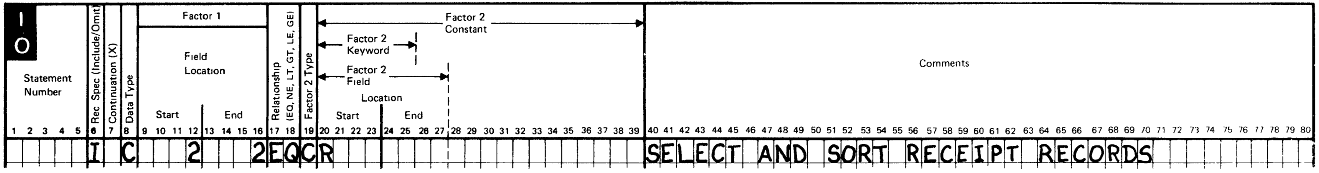
Field Selection



```
     FF          4                     FORCE 4 INTO ISSUE RECORDS
     FNU  11  16                       QUANTITY ORDERED IS CONTROL FIELD
     FDC   1  49                       WRITE ALL INPUT FIELDS INTO OUTPUT
```

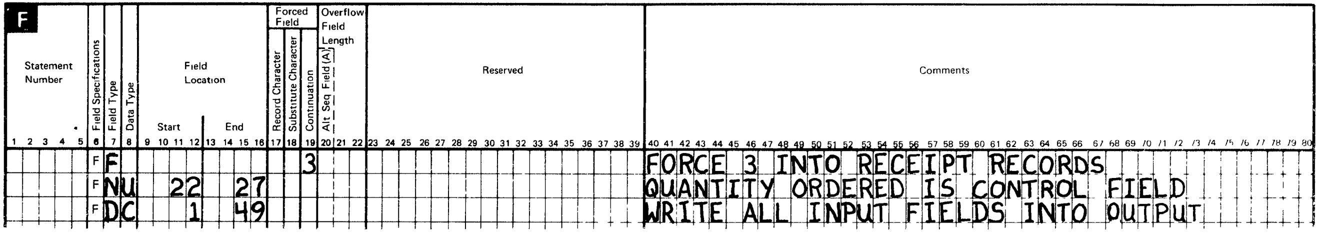
Set 2 describes how the receipt records are sorted.

Record Selection



```
     I C   2   2EQCR                   SELECT AND SORT RECEIPT RECORDS
```

Field Selection



```
     FF           3                    FORCE 3 INTO RECEIPT RECORDS
     FNU  22  27                       QUANTITY ORDERED IS CONTROL FIELD
     FDC   1  49                       WRITE ALL INPUT FIELDS INTO OUTPUT
```

Set 3 describes how the adjustment records are sorted.

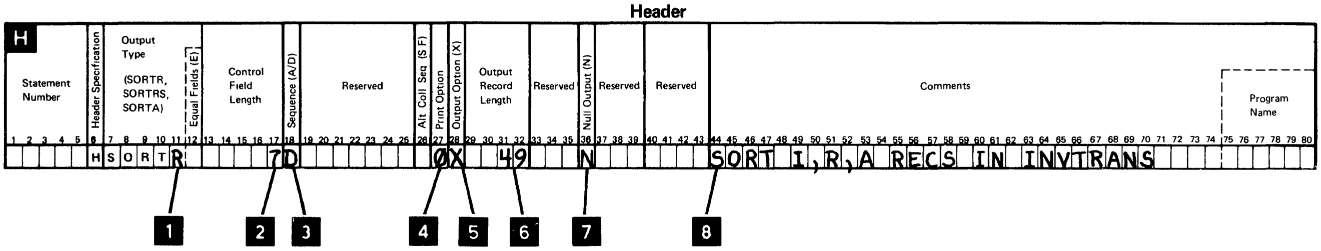
Record Selection

I O Statement Number 1 2 3 4 5	Rec Spec (Include/Omit) 6	Continuation (X) 7	Data Type 8	Factor 1 Field Location Start 9 10 11 12	Factor 1 Field Location End 13 14 15 16	Relationship (EQ, NE, LT, GT, LE, GE) 17 18	Factor 2 Type 19	Factor 2 Constant / Factor 2 Keyword / Factor 2 Field Location Start End 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39	Comments 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
	I		C	2	2	EQ	C	A	SELECT ADJUSTMENT RECORDS ONLY

Field Selection

F Statement Number 1 2 3 4 5	Field Specifications 6	Field Type 7	Data Type 8	Field Location Start 9 10 11 12	Field Location End 13 14 15 16	Forced Field Record Character 17	Forced Field Substitute Character 18	Continuation 19	Overflow Field Length Alt Seq Field (A) 20 21 22	Reserved 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39	Comments 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
	F	F	C	11	11	2					FORCE 2 FOR ADDITIONS TO STOCK
	F	F	C	11	11	2	1	X			FORCE 1 FOR SHORT ITEMS IN STOCK
	F	N	C	12	17						QUANTITY ORDERED IS CONTROL FIELD
	F	D	C	1	49						WRITE ALL INPUT FIELDS INTO OUTPUT

Set 1 Header Specification Entries



HSORTR 7D ØX 49 N SORT I,R,A RECS IN INVTRANS

1 Indicates that a regularly sorted output file will be generated.

2 Specifies the number of characters in the control field. In this example, quantity ordered, which contains 6 characters, is the control field. A single forced control field will also be used, so 1 must be added to the control field length, for a total of 7.

3 Specifies that the output records will be placed in descending order (the highest values will be written into the output file first).

4 Specifies that the print option is zero. A zero in column 27 specifies that the sort program will print the following information about the job:

- Sort specification statements
- Diagnostic messages indicating any errors encountered in sort specifications
- Program status messages identifying various stages of the job
- Action messages (followed by displayed messages) identifying circumstances requiring attention before the job can continue
- Displayed messages that appear on the display screen.

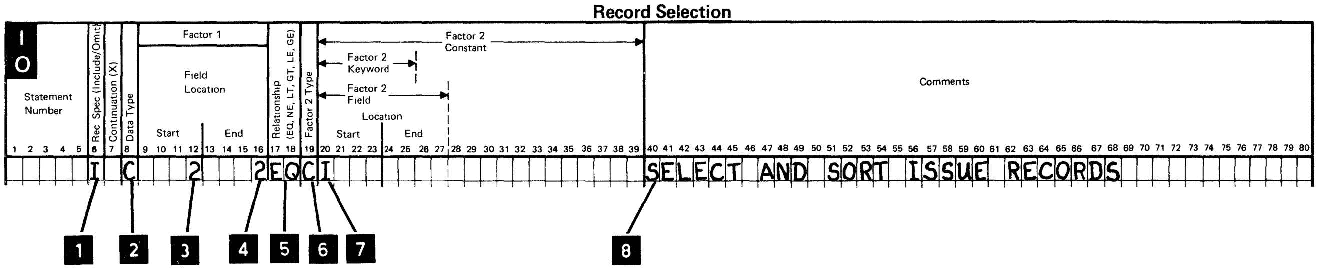
5 Specifies that the sort program will drop the control field data after the records are sorted. Consequently, the output records will contain only the fields specified as data fields.

6 Specifies that the output record will contain 49 characters from the input record.

7 Specifies that the sort program should create an empty output file without issuing message SORT-7724, *No input records included.* The sort program takes this option only *if input records do not meet the conditions specified in the record selection statements and are not selected for sorting.*

8 Are comments about the sort job.

Set 1 Record Selection Entries



Record Selection

I O Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ, NE, LT, GT, LE, GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location (20-39)	Comments (40-80)
	I		C	2	2	EQ	C	I	SELECT AND SORT ISSUE RECORDS

1 Specifies that only the records described in this statement should be included in this sort.

2 Identifies the data in position 2 as character data, with both the zone and digit portions of characters used in compare operations.

3, 4 Identify the location of factor 1, to which factor 2 (7) is compared.

5 Specifies that the zone and digit portions of the data in position 2 must equal that of the character I. Then the record will be included in the sort.

6 Specifies that the data from the input record will be compared to constant data.

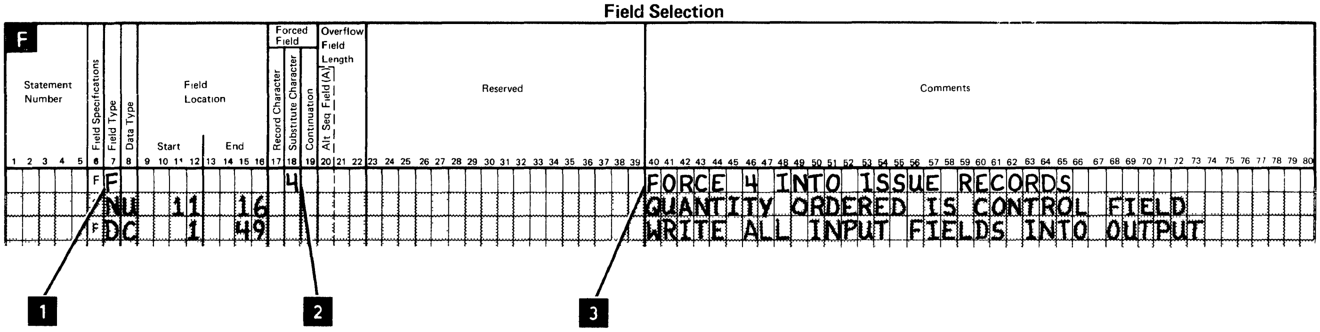
7 Identifies the factor 2 character constant, I. This factor 2 constant is compared with the data in position 2 (or factor 1) to determine their relationship. If factor 1 and factor 2 are equal, the records are included in the sorted output file.

8 Is a comment about this include statement.

Set 1 Field Selection Entries

Three field selection statements are defined for sorting the issue records.

Statement 1



Field Selection

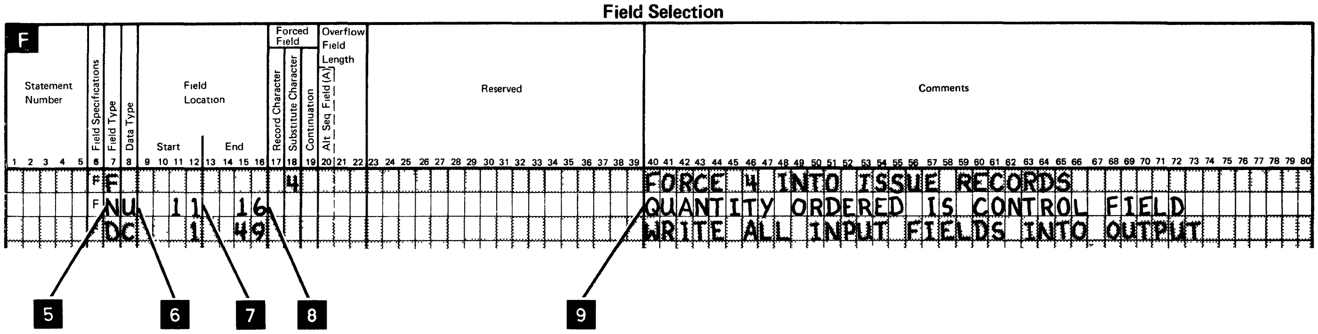
Statement Number 1-5	Field Specifications 6	Field Type 7	Data Type 8	Field Location Start 9-12	Field Location End 13-16	Forced Field Record Character 17	Forced Field Substitute Character 18	Continuation 19	Overflow Field Length / Alt Seq Field (A) 20-22	Reserved 23-39	Comments 40-80
	F	F					4				FORCE 4 INTO ISSUE RECORDS
		N	U	11	16						QUANTITY ORDERED IS CONTROL FIELD
	F	D	C	1	49						WRITE ALL INPUT FIELDS INTO OUTPUT

1 Identifies the field as a 1-character unconditional-forced control field. For this record type, the sort program will force a *4* into the first available position of the control field.

2 Is the character that will be placed in the first control field position. Notice that the forced character replacements are made before the records are sorted.

3 A comment about the forced control field.

Statement 2


Field Selection
F
5
6
7
8
9

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Alt Seq Field (A)	Overflow Field Length	Reserved	Comments
	F	F				4						FORCE 4 INTO ISSUE RECORDS
	F	N	U	11	16							QUANTITY ORDERED IS CONTROL FIELD
		D	C	1	49							WRITE ALL INPUT FIELDS INTO OUTPUT

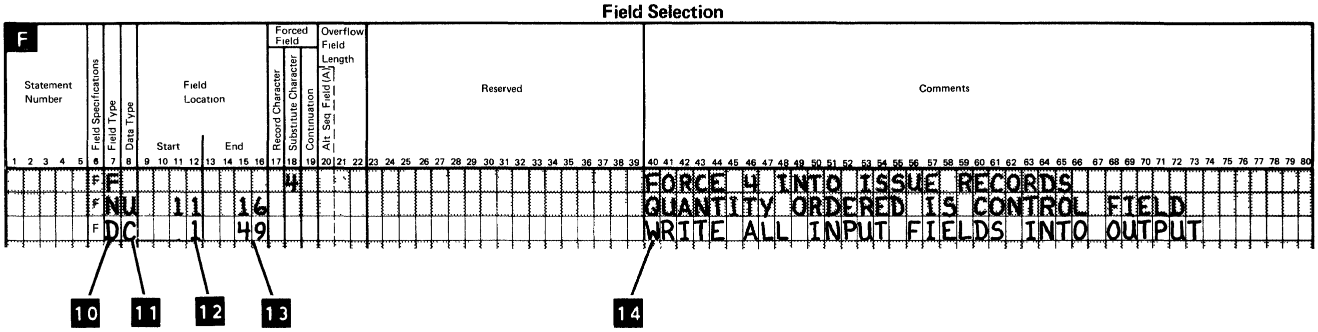
5 Identifies the control field as a normal one.

6 Identifies the type of data in the control field as signed unpacked (zoned) decimal data. The sign is to be moved to the work record. This entry enables the sort program to compare both the zone and digit portions of each character and the sign of each character.

7, **8** Specify the location of the control field in the input records as starting in position 11 and ending in position 16.

9 Is a comment about the field specification.

Statement 3


Field Selection
F
10
11
12
13
14

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Alt Seq Field (A)	Overflow Field Length	Reserved	Comments
	F	F				4						FORCE 4 INTO ISSUE RECORDS
	F	N	U	11	16							QUANTITY ORDERED IS CONTROL FIELD
	F	D	C	1	49							WRITE ALL INPUT FIELDS INTO OUTPUT

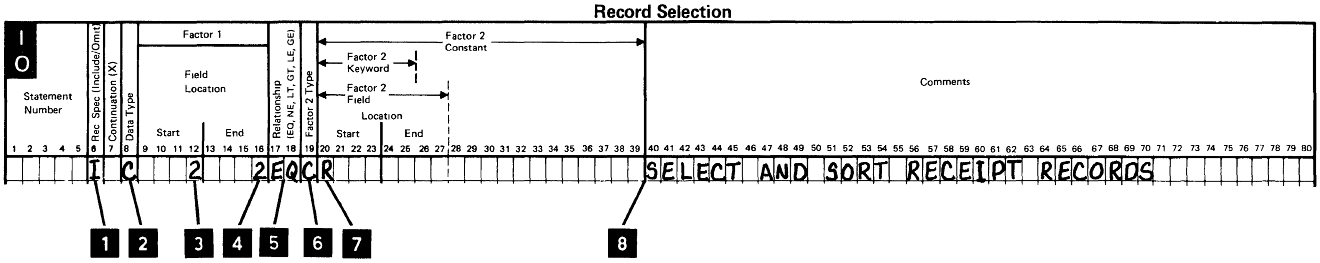
10 Identifies the field as a data field.

11 Specifies that the sort program should interpret the data in this data field as character data where both the zone and digit portions are included in the output record.

12, 13 Identify the location of the data field within the input records. The sort program will take the data in positions 1 through 49 of the input records and write it into the output records. The output data starts in position 1 of the input records and ends in position 49.

14 Is a comment about the field specification statement.

Set 2 Record Selection Entries



Statement Number	Rec Spec (Include/Omit)	Continuation (X)	Data Type	Factor 1 Field Location Start	Factor 1 Field Location End	Relationship (EQ, NE, LT, GT, LE, GE)	Factor 2 Type	Factor 2 Constant	Comments
	I		C	2	2	EQ	C	R	SELECT AND SORT RECEIPT RECORDS

1 Specifies that only the records described in this statement and meeting the stated conditions should be included in the sort.

2 Identifies the data in position 2 as character data. The sort program looks at both the zone and digit portions of character data.

3, 4 Identify the location of factor 1, to which factor 2 is compared.

5 Specifies that the data in position 2 of each record must equal the character R for the records to be included in the sort.

6 Specifies that the character R will be compared to constant data.

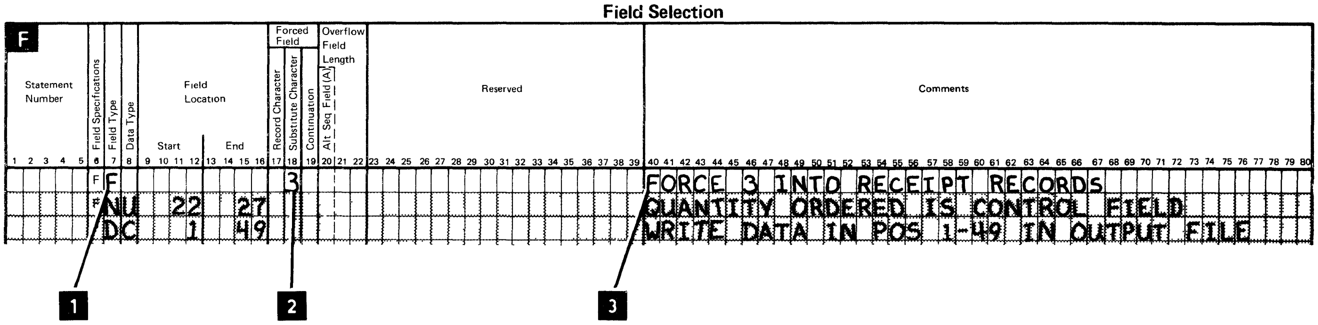
7 Identifies the factor 2 character constant. This character constant is compared to the contents of position 2, factor 1, to determine their relationship. If they compare as equals, the records are included in the sorted output file.

8 Is a comment about this include statement.

Set 2 Field Selection Entries

Three field selection statements are defined to sort the receipt records.

Statement 1



Field Selection

F

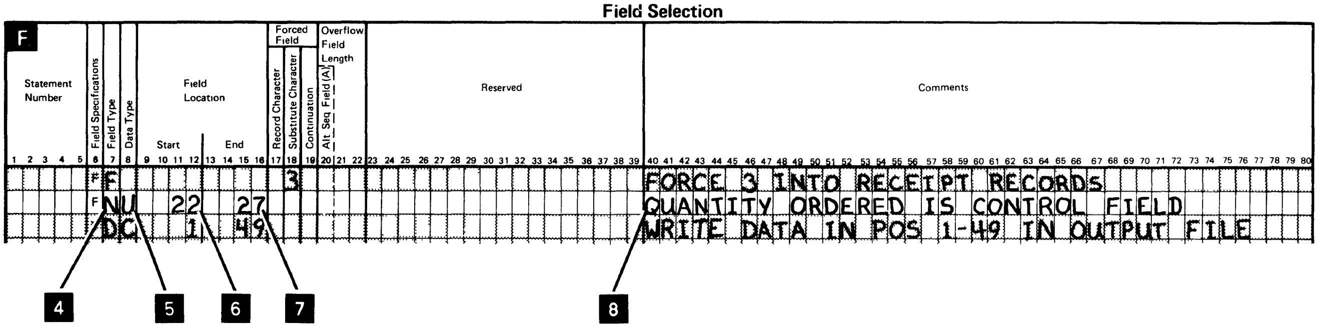
Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length Alt Seq Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	F					3				FORCE 3 INTO RECEIPT RECORDS
	F	NU		22	27						QUANTITY ORDERED IS CONTROL FIELD
		DC		1	49						WRITE DATA IN POS 1-49 IN OUTPUT FILE

1 Identifies the field as a 1-character unconditional-forced control field. For this record type, the sort program should force a 3 into into the first available position of the control field.

2 Identifies the character that will be placed in the first control field position. Notice that the forced character replacements are made before the records are sorted.

3 A comment about the forced control field.

Statement 2



Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt. Seq. Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	F				3					FORCE 3 INTO RECEIPT RECORDS
	F	N	U	22	27						QUANTITY ORDERED IS CONTROL FIELD
	F	D	C	1	49						WRITE DATA IN POS 1-49 IN OUTPUT FILE

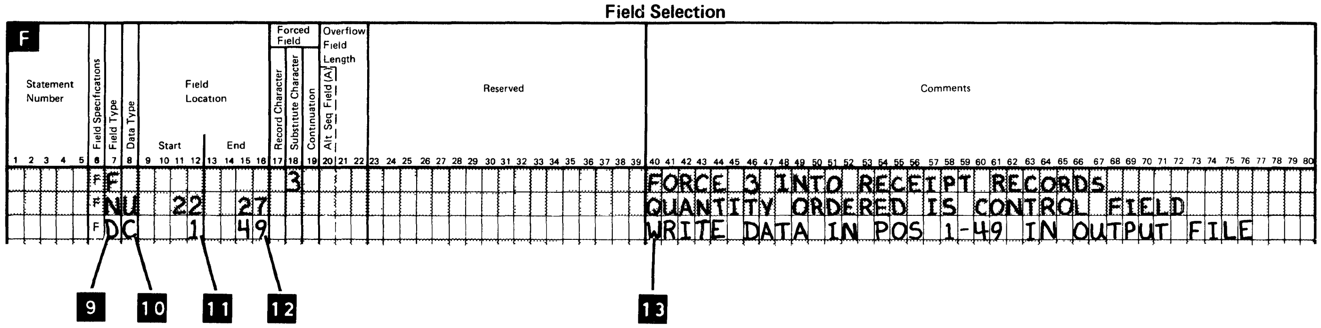
4 Identifies the control field as a normal one.

5 Identifies the type of data in the control field as signed unpacked (zoned) decimal data. The sign is to be moved to the work record. This entry enables the sort program to compare both the zone and digit portions of each character and the sign of each character.

6, **7** Specify the location of the control field in the input records as starting in position 22 and ending in position 27.

8 Is a comment about the job.

Statement 3



Field Selection

Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt. Seq. Field (A) (20-22)	Reserved (23-39)	Comments (40-80)
	F	F				3					FORCE 3 INTO RECEIPT RECORDS
	F	N	U	22	27						QUANTITY ORDERED IS CONTROL FIELD
	F	D	C	1	49						WRITE DATA IN POS 1-49 IN OUTPUT FILE

9 Identifies the field as a data field.

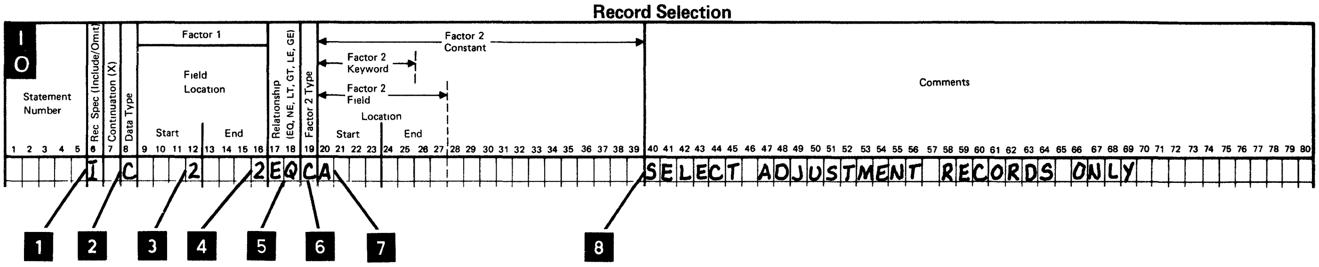
10 Specifies that the sort program should interpret the data in this data field as character data where both the zone and digit portions are included in the output record.

11, **12** Identify the location of the data field within the input records. The sort program will take the data in positions 1 through 49 of the input records and write it into the output records. The output data starts in position 1 of the input records and ends in position 49.

13 Is a comment about the job.

Set 3 Record Selection Entries

The following is the second of two include statements used in this example:



```
     I C   2   2EQCA                   SELECT ADJUSTMENT RECORDS ONLY
```

1 Specifies that only the records described in this statement should be included in the sort.

2 Identifies the data in position 2 as character data. The sort program looks at both the zone and digit portions of character data.

3, **4** Identify the location of the factor 1 field in the input records.

5 Specifies that the data in position 2 of each record must equal the character A for the records to be included in the sort.

6 Specifies that the data to which the character A will be compared is constant data.

7 Identifies the factor 2 character constant. This character constant is compared to the contents of position 2, factor 1, to determine their relationship. If they compare as equals, then the records are included in the sorted output file.

8 Is a comment about the sort job.

Field Selection Entries

These specifications cause the records containing adjustments to be grouped together. The adjustment records contain two types of information:

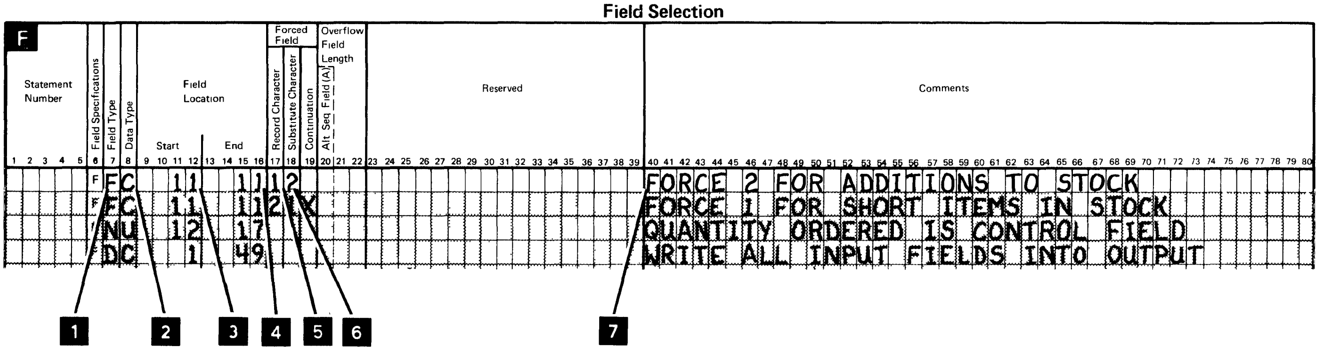
- *Short items in stock* indicated by a 2 in position 11
- *Additions to stock* indicated by a 1 in position 11.

The input file is sorted so all the records containing short items in stock will be written into the output file first. Any records containing additions to stock will also be grouped together and written into the output file last.

To cause the records to sort in reverse order, a conditional force is used to place a substitute character into the first control field position of each record. In this example, a 1 is forced into the first control field position of those records containing a 2 in position 11. The character 2 is the adjustment code for short items in stock. A 2 is forced into the first control field position of each record containing a 1 in position 11 (for additions to stock).

In this way, because of the descending order of the sort, the additions to stock will be grouped together and written into the output file before the short items in stock.

Statement 1



Field Selection

F

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length Alt Seq Field (A)	Reserved	Comments
	F	F	C	11	11	1	2				FORCE 2 FOR ADDITIONS TO STOCK
	F	F	C	11	11	2	1	X			FORCE 1 FOR SHORT ITEMS IN STOCK
	F	N	U	12	17						QUANTITY ORDERED IS CONTROL FIELD
	F	D	C	1	49						WRITE ALL INPUT FIELDS INTO OUTPUT

1 Specifies a forced control field.

2 Identifies the data in the forced field as character data. Therefore the sort program will compare both the zone and digit portions of each character, or the whole character.

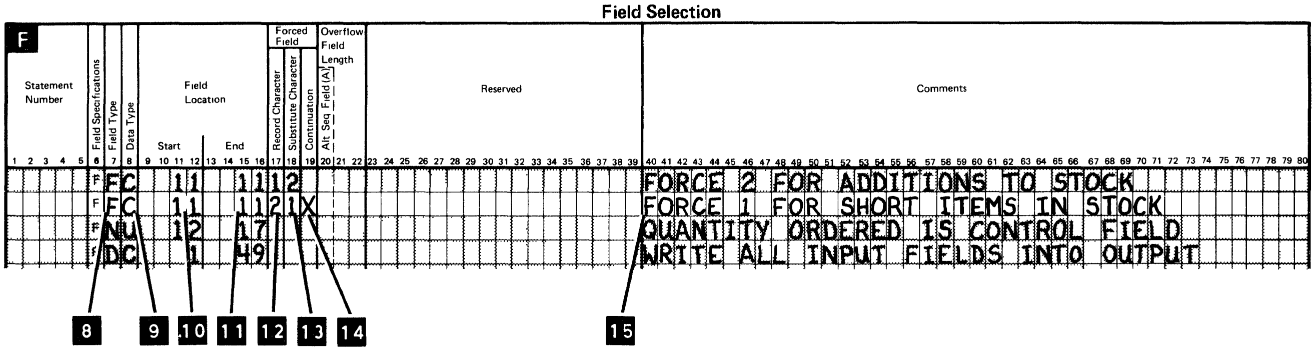
3, **4** Indicate that the field in the input record which is to be tested is located in position 11.

5 Identifies the record character that will be replaced. If position 11 contains a 1, the sort program should force a 2 into the record.

6 Identifies the character that will be forced into the first position of the work records. The character 2 will identify any additions to stock.

7 Is a comment about the sort job.

Statement 2


Field Selection
F
8
9
10
11
12
13
14
15

Statement Number (1–5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9–12)	Field Location End (13–16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Overflow Field Length / Alt Seq Field (A) (20–22)	Reserved (23–39)	Comments (40–80)
	F	F	C	11	11	1	2				FORCE 2 FOR ADDITIONS TO STOCK
	F	F	C	11	11	2	1	X			FORCE 1 FOR SHORT ITEMS IN STOCK
	F	N	U	12	17						QUANTITY ORDERED IS CONTROL FIELD
	F	D	C	1	49						WRITE ALL INPUT FIELDS INTO OUTPUT

8 Is the same as **1**.

9 Is the same as **2**.

10, **11** Are the same as **3** and **4**.

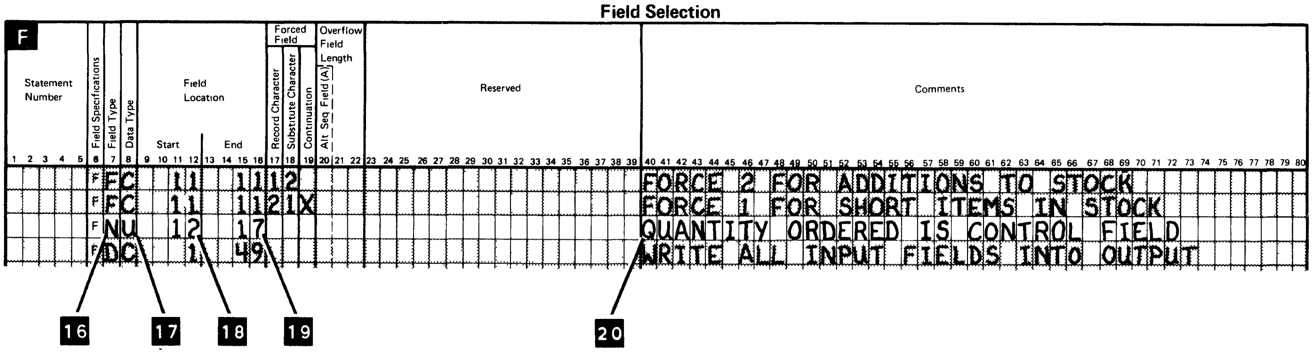
12 Is the same as **5**, except that if the records contain a 2 in position 11, the 2 should be replaced by the character 1.

13 Is the same as **6**, except the character 1 will identify any short items in stock.

14 Specifies that this specification statement is a continuation of the preceding line.

15 Is the same as **7**.

Statement 3


16
17
18
19
20

Field Selection

F Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length Alt Seq Field (A)	Reserved	Comments
	F	F	C	11	11	1	2				FORCE 2 FOR ADDITIONS TO STOCK
	F	F	C	11	11	2	1	X			FORCE 1 FOR SHORT ITEMS IN STOCK
	F	N	U	12	17						QUANTITY ORDERED IS CONTROL FIELD
	F	D	C	1	49						WRITE ALL INPUT FIELDS INTO OUTPUT

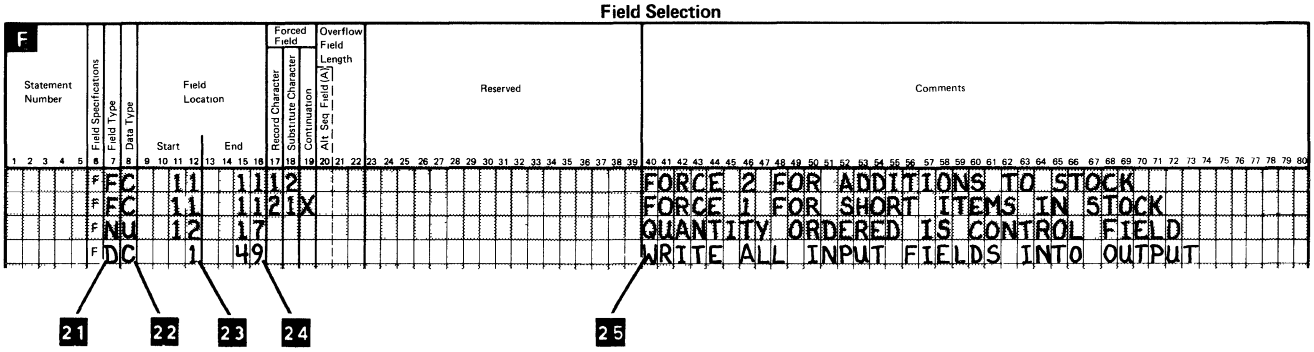
16 Identifies the field described in this specification statement as a normal control field.

17 Specifies that the control field contains unpacked numeric data whose sign, in addition to the zone and digit portions of each character, will be used in the compare operations.

18, 19 Identify the location of the control field in the input records as positions 12 through 17.

20 Is the same as 7 and 15, a comment about the job.

Statement 4


21
22
23
24
25

Field Selection

F Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length Alt Seq Field (A)	Reserved	Comments
	F	F	C	11	11	1	2				FORCE 2 FOR ADDITIONS TO STOCK
	F	F	C	11	11	2	1	X			FORCE 1 FOR SHORT ITEMS IN STOCK
	F	N	U	12	17						QUANTITY ORDERED IS CONTROL FIELD
	F	D	C	1	49						WRITE ALL INPUT FIELDS INTO OUTPUT

21 Identifies the field as a data field that will be written into the output file.

22 Specifies the data as character data whose zone and digit portions will be compared by the sort program.

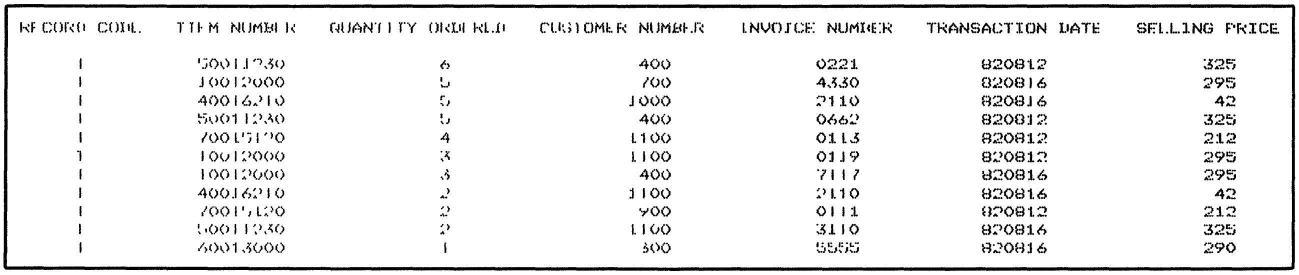
23, 24 Provide the location of the data in the input records that will be written into the output file.

25 Is a comment about the job.

Output

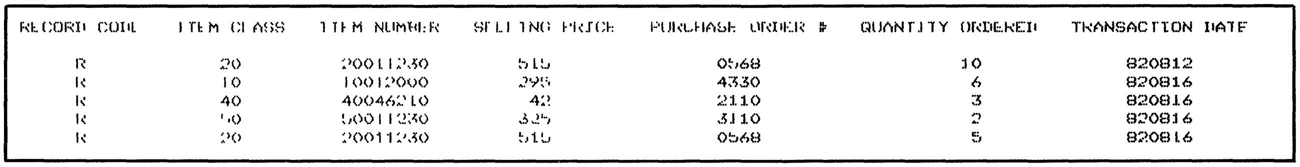
The output file might contain the following:

Issue Records



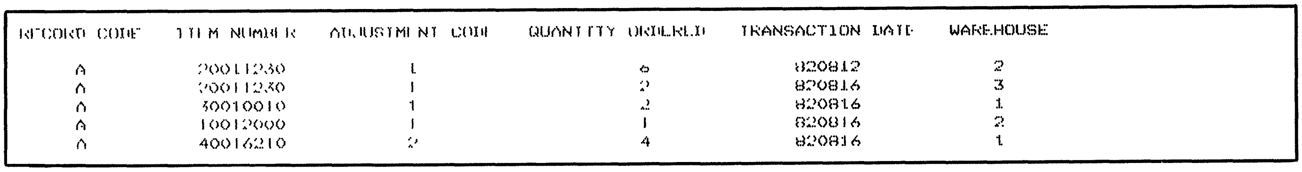
```
RECORD CODE   ITEM NUMBER   QUANTITY ORDERED   CUSTOMER NUMBER   INVOICE NUMBER   TRANSACTION DATE   SELLING PRICE

     I          50011230           6                 400              0221            820812              325
     I          10012000           5                 700              4330            820816              295
     I          40016210           5                1000              2110            820816               42
     I          50011230           5                 400              0662            820812              325
     I          70015120           4                1100              0113            820812              212
     I          10012000           3                1100              0119            820812              295
     I          10012000           3                 400              7117            820816              295
     I          40016210           2                1100              2110            820816               42
     I          70015120           2                 900              0111            820812              212
     I          50011230           2                1100              3110            820816              325
     I          60013000           1                 300              5555            820816              290
```

Receipt Records



```
RECORD CODE   ITEM CLASS   ITEM NUMBER   SELLING PRICE   PURCHASE ORDER #   QUANTITY ORDERED   TRANSACTION DATE

     R            20         20011230         515              0568               10               820812
     R            10         10012000         295              4330                6               820816
     R            40         40046210          42              2110                3               820816
     R            50         50011230         325              3110                2               820816
     R            20         20011230         515              0568                5               820816
```

Adjustment Records



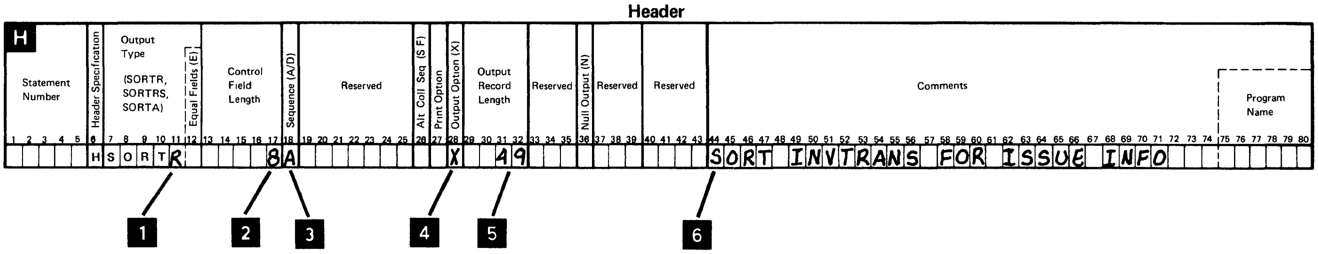
```
RECORD CODE   ITEM NUMBER   ADJUSTMENT CODE   QUANTITY ORDERED   TRANSACTION DATE   WAREHOUSE

     A          20011230           1                  6              820812            2
     A          20011230           1                  2              820816            3
     A          30010010           1                  2              820816            1
     A          10012000           1                  1              820816            2
     A          40016210           2                  4              820816            1
```

Example 8 of a Sort Job. Sort a File Containing Multiple Record Types and Write Only Issue Records Into Output File

Input

This example is based on the input file used in Example 7, INVTRANS. Its purpose is to show how to omit all other records but those containing information about items issued to customers, or an I in position 2 of the input records. All records containing an I in position 2 of the input file should be included in the sort job and written into the output file. The sort specifications for this job were defined like the following:

Header Specification Entries



Header

H Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
1 2 3 4 5	6	7 8 9 10 11	12	13 14 15 16 17	18	19 20 21 22 23 24 25	26	27	28	29 30 31 32	33 34 35	36	37 38 39	40 41 42 43	44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	75 76 77 78 79 80
	H	SORTR		8	A				X	49					SORT INVTRANS FOR ISSUE INFO	

1 Indicates that a regularly sorted output file will be generated.

2 Specifies the number of characters in the field in the input records to use in sorting the records. In this example, item number, which contains 8 characters, is the control field.

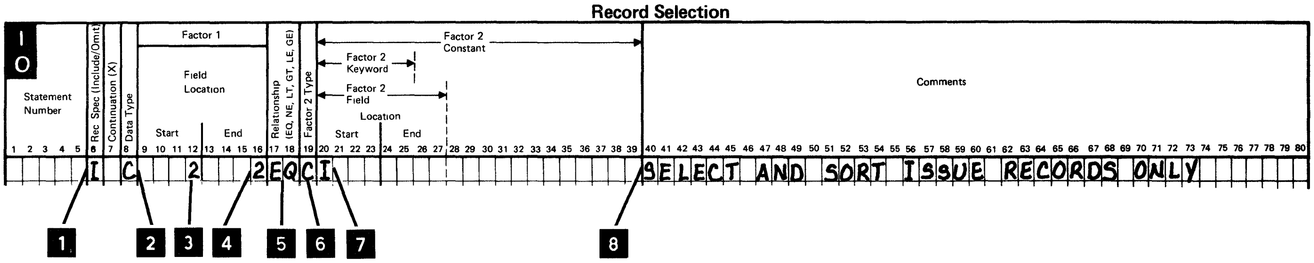
3 Specifies that the output data will be written in ascending order.

4 Specifies that the sort program will drop the control field data after the records are sorted. Consequently, the output records will contain only the fields specified as data fields. As you can see, because the output will contain all the data in positions 1 through 49 of the input records, the item numbers in positions 3 through 10 will also be included.

5 Specifies that the output record will contain 49 characters from the input record.

6 Provides a record of your sort job.

Record Selection Entries



I C 2 2 EQ C I SELECT AND SORT ISSUE RECORDS ONLY

1 Specifies that only the records described in this statement should be included in the sort.

2 Identifies the data in position 2 as character data. The sort program looks at both the zone and digit portions of character data.

3, **4** Identify the location of the factor 1 field.

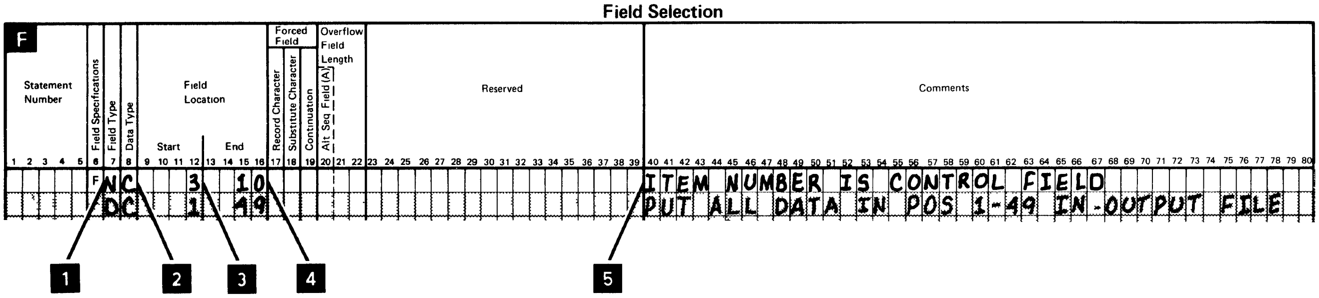
5 Specifies that the data in position 2 of each record must equal the character I for the records to be included in the sort.

6 Specifies that the data to which the character I will be compared. is constant data.

7 Identifies the factor 2 character constant. This character constant is compared to the contents of position 2, factor 1, to determine their relationship. If they compare as equals, then the records are included in the sorted output file.

8 Is a comment about the job.

Field Selection Entries



Field Selection

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length Alt Seq Field (A)	Reserved	Comments
	F	N	C	3	10						ITEM NUMBER IS CONTROL FIELD
		D	C	1	49						PUT ALL DATA IN POS 1-49 IN-OUTPUT FILE

F
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
1
2
3
4
5

Two field specifications statements are used to describe the issue records. The first is designated as statement 1, the second as statement 2.

Statement 1

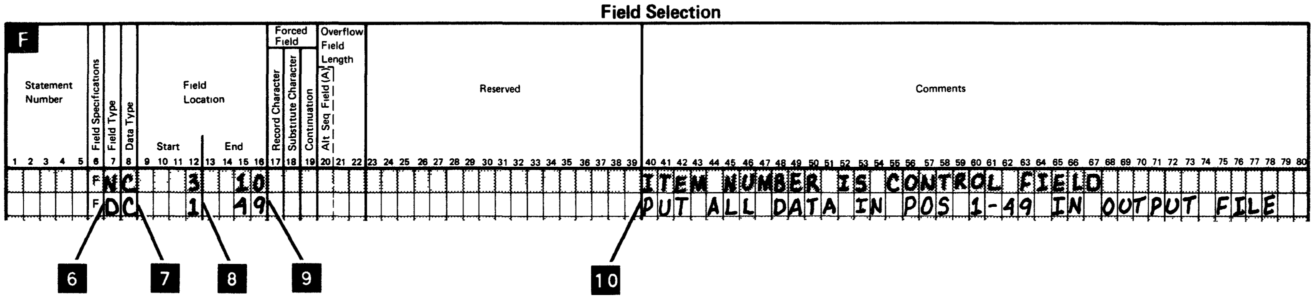
1 Identifies the field as a normal control field.

2 Specifies that the sort program should interpret the data in the control field as character data where both the zone and digit portions are compared.

3, **4** Identify the location of the control field within the input records. The control field starts in position 3 of the input records and ends in position 10, which makes it an eight character field.

5 Provides documentation about the sort job. This comment simply means the item number is the control field.

Statement 2



```
     FNC   3  10                       ITEM NUMBER IS CONTROL FIELD
     FDC   1  49                       PUT ALL DATA IN POS 1-49 IN OUTPUT FILE
```

6 Identifies the field as a data field.

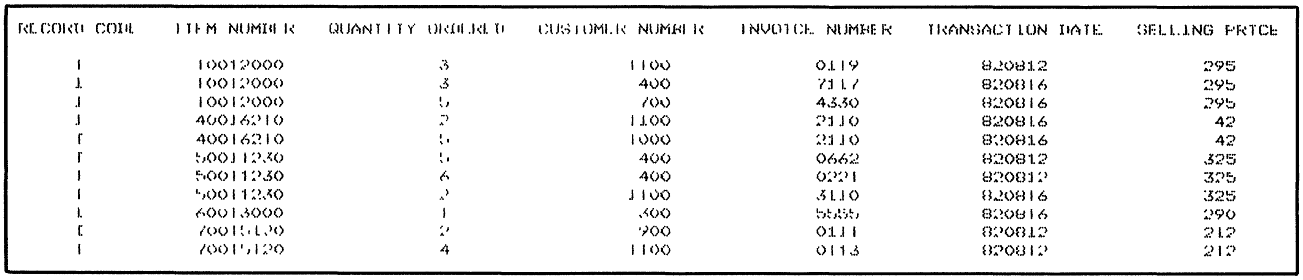
7 Specifies that the sort program should interpret the data in this data field as character data where both the zone and digit portions are included in the output record.

8, **9** Identify the location of the data field within the input records. The sort program will take the data in positions 1 through 49 of the input records and write it into the output records. The output data starts in position 1 of the input records and ends in position 49.

10 Is a comment that provides documentation about the output for the sort job.

Output

The output file will look similar to the following:



```
RECORD CODE   ITEM NUMBER   QUANTITY ORDERED   CUSTOMER NUMBER   INVOICE NUMBER   TRANSACTION DATE   SELLING PRICE

     I         10012000            3                1100              0119             820812              295
     I         10012000            3                 400              7117             820816              295
     I         10012000            5                 700              4330             820816              295
     I         40016210            2                1100              2110             820816               42
     I         40016210            5                1000              2110             820816               42
     I         50011230            5                 400              0662             820812              325
     I         50011230            6                 400              0221             820812              325
     I         50011230            2                1100              3110             820816              325
     I         60013000            1                 300              5555             820816              290
     I         70015120            2                 900              0111             820812              212
     I         70015120            4                1100              0113             820812              212
```

Note: In this example, the order of the groups of records with the same customer number is unpredictable unless equal control field ordering is used. See *Sorting Records with Identical Control Fields* in Chapter 2 for more information about equal control field ordering.

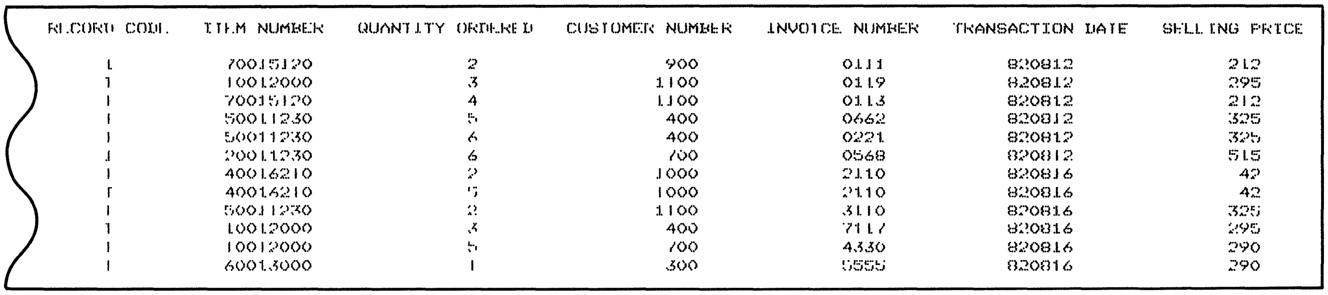
Example 9 of a Sort Job. Sort INVTRANS and Summarize Adjustments

Assume in this example that you want to check certain stock items for shortages. The file containing all inventory transactions will be sorted and another file containing only items with adjustments will be generated.

Example 9 shows how the sort program summarizes data in specified fields. In this example, the number of items requiring adjustment will be added together and sorted by item number. The input file contains different record types: delivery records, adjustment records, and receipt records. In this example, all delivery and receipt records are omitted from the sort. Only those records having a record code (A) in position 2 will be included in the sorted output. The sort specifications for this job follow.

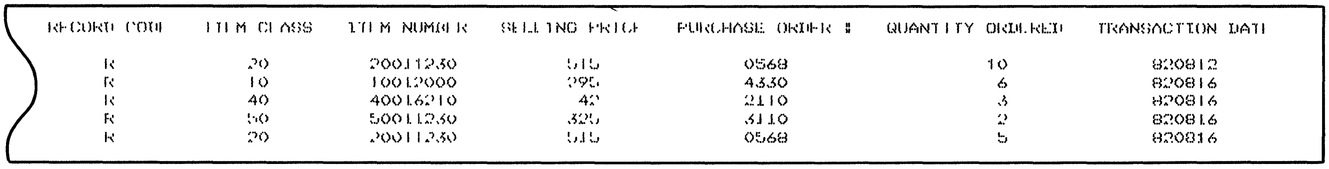
Input

This example also uses the same input file used in Examples 7 and 8, INVTRANS, which contains the following data:



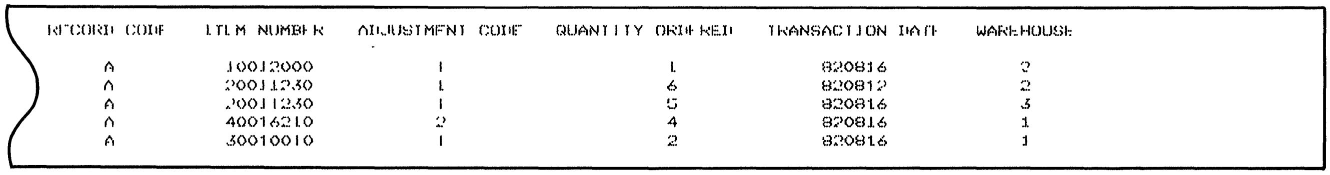
```
RECORD CODE   ITEM NUMBER   QUANTITY ORDERED   CUSTOMER NUMBER   INVOICE NUMBER   TRANSACTION DATE   SELLING PRICE

     I         70015120           2                  900              0111            820812              212
     I         10012000           3                 1100              0119            820812              295
     I         70015120           4                 1100              0113            820812              212
     I         50011230           5                  400              0662            820812              325
     I         50011230           6                  400              0221            820812              325
     I         20011230           6                  700              0568            820812              515
     I         40016210           2                 1000              2110            820816               42
     I         40016210           5                 1000              2110            820816               42
     I         50011230           2                 1100              3110            820816              325
     I         10012000           3                  400              7117            820816              295
     I         10012000           5                  700              4330            820816              290
     I         60013000           1                  300              5555            820816              290
```

Delete Code	Record Code	Item Number	Quantity Ordered	Customer Number	Invoice Number	Transaction Date	Selling Price		
1	2	3 10	11 16	17 24	25 32	33 38	39 43		49



```
RECORD CODE   ITEM CLASS   ITEM NUMBER   SELLING PRICE   PURCHASE ORDER #   QUANTITY ORDERED   TRANSACTION DATE

     R            20        20011230          515              0568                10               820812
     R            10        10012000          295              4330                 6               820816
     R            40        40016210           42              2110                 3               820816
     R            50        50011230          325              3110                 2               820816
     R            20        20011230          515              0568                 5               820816
```

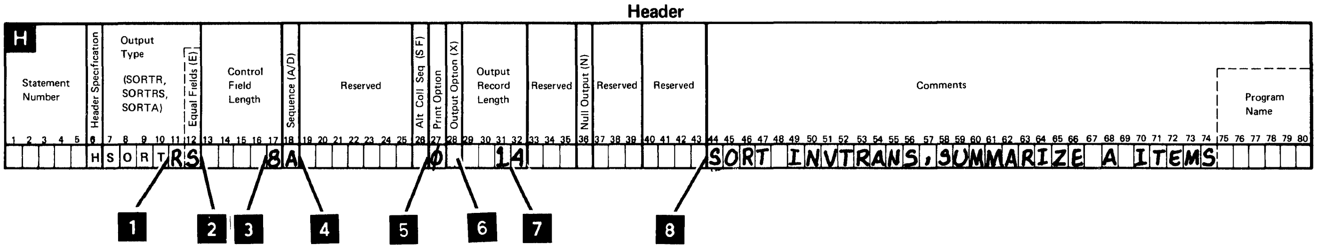
Delete Code	Record Code	Item Class	Item Number	Selling Price	Purchase Order Number	Quantity Ordered	Transaction Date		Blank
1	2	3 4	5 12	13 17	18 21	22 27	28 33		49



```
RECORD CODE   ITEM NUMBER   ADJUSTMENT CODE   QUANTITY ORDERED   TRANSACTION DATE   WAREHOUSE

     A         10012000            1                  1               820816            2
     A         20011230            1                  6               820812            2
     A         20011230            1                  5               820816            3
     A         40016210            2                  4               820816            1
     A         30010010            1                  2               820816            1
```

Delete Code	Record Code	Item Number	Adjustment Code	Quantity Ordered	Transaction Date	Warehouse	Blank
1	2	3 10	11	12 17	18 23	24	25 49

Header Specification Entries



```
     HSORTRS    8A        Ø   14           SORT INVTRANS, SUMMARIZE A ITEMS
```

1 Indicates that a sorted output file containing fields of accumulated totals, or summary data, will be generated. In a summary sort, the input records are sorted in a manner similar to that of a regular sort except that specified fields will contain data that was added together.

2 Specifies the job as a summary sort. Data in specified fields will be added together.

3 Specifies the number of characters in the field in the input records to use in sorting the records. In this example, item number, which contains 8 characters, is the control field.

4 Specifies that the output records will be written in ascending order.

5 Specifies that the print option indicator should be a zero. A zero in column 27 specifies that the sort program will print the following information about the sort job:

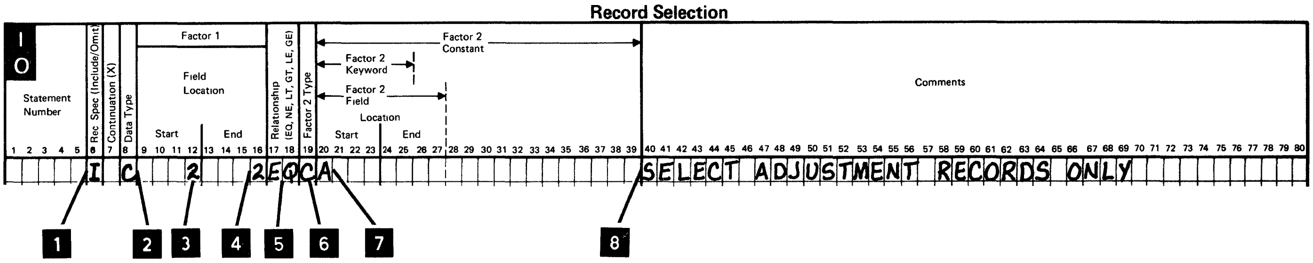
- Sort specification statements
- Diagnostic messages indicating any errors encountered in sort specifications
- Program status messages identifying various stages of the job
- Action messages (followed by displayed messages) identifying circumstances requiring attention before the job can continue
- Displayed messages that appear on the display screen

6 Specifies that the sort program should keep the control fields. You can, however, specify that the data in the control field be written into the output file by specifying the control field as a data field in a field specifications statement.

7 Specifies that the output record will contain only 14 characters from the input record, the item numbers and the summarized quantities for the items requiring adjustment.

8 Provides a record of your sort job.

Record Selection Entries



I C 2 2 EQ C A SELECT ADJUSTMENT RECORDS ONLY

1 Specifies that only the records described in this statement should be included in the sort.

2 Identifies the data in position 2 as character data. The sort program looks at both the zone and digit portions of character data.

3, **4** Identify the location of the factor 1 field that will be compared with factor 2.

5 Specifies that the data in position 2 of each record must equal the character A for the records to be included in the sort.

6 Specifies that the data to which the character A will be compared is constant data.

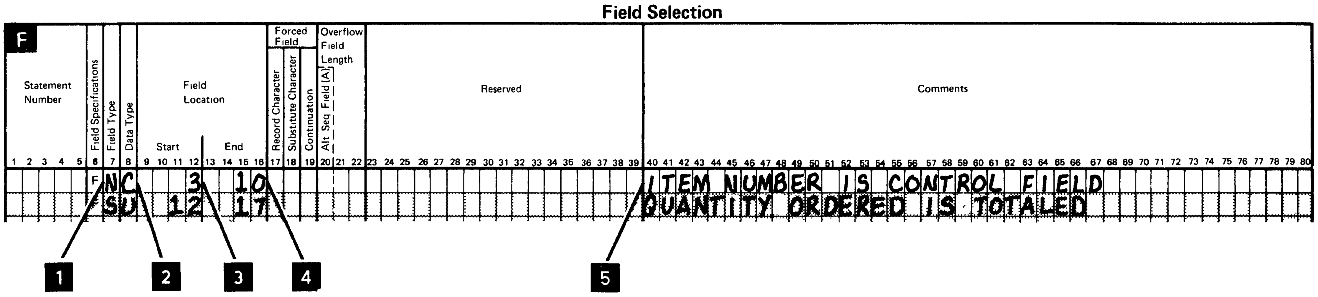
7 Identifies the factor 2 as a character constant. This character constant is compared to the contents of position 2, factor 1, to determine their relationship. If they compare as equals, then the records are included in the sorted output file.

8 Provides documentation for your sort job.

Field Selection Entries

Two field selection statements are used to describe the adjustment records. The first is designated as Statement 1 and the second as Statement 2.

Statement 1



Field Selection

F Statement Number 1–5	Field Specifications 6	Field Type 7	Data Type 8	Field Location Start 9–12	Field Location End 13–16	Forced Field Record Character 17	Forced Field Substitute Character 18	Continuation 19	Alt Seq Field (A) 20	Overflow Field Length 21–22	Reserved 23–39	Comments 40–80
	F	N	C	3	10							ITEM NUMBER IS CONTROL FIELD
	F	S	U	12	17							QUANTITY ORDERED IS TOTALED

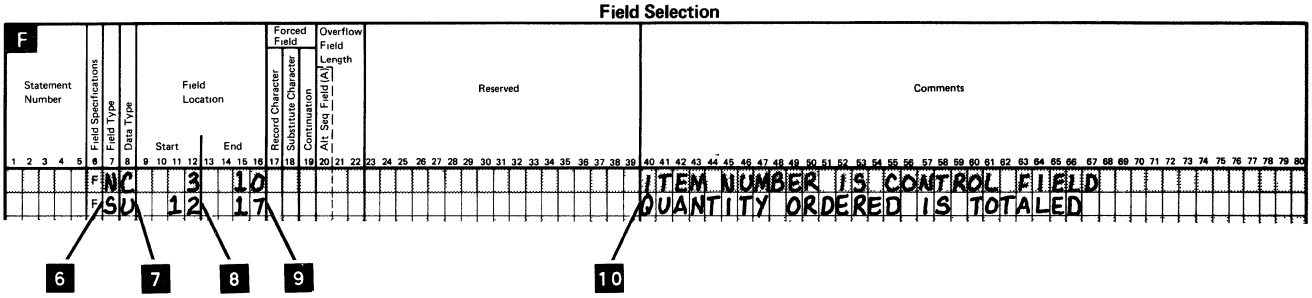
1 Identifies the field as a normal control field.

2 Specifies that the sort program should interpret the data in the control field as character data where both the zone and digit portions are compared.

3, **4** Identify the location of the control field within the input records. The control field starts in position 3 of the input records and ends in position 10, which makes it an eight position field.

5 Provides documentation about the sort job.

Statement 2



Field Specifications	Field Type	Data Type	Start	End	Comments
F	N	C	3	10	ITEM NUMBER IS CONTROL FIELD
F	S	U	12	17	QUANTITY ORDERED IS TOTALED

6 Identifies the field as a summary data field. The sort program treats summary data fields as columns of data that must be added together. When one is encountered, the data in it is in fact added to any total in a matching record.

7 Specifies that the sort program should interpret the data in this data field as unpacked numeric data where the zone, digit, and sign portions are included in the output record.

8, **9** Identify the location of the field to be summarized within the input records. The sort program will take the data in positions 12 through 17 of the input records, add it together, and write it into the specified positions in the output records.

10 Indicates a comment about the job.

Output

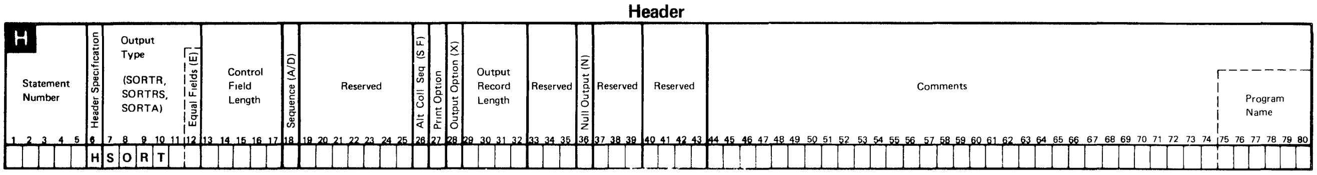
The output file would look like the following:

```
ITEM NUMBER          QUANTITY ORDERED

 10012000                   1
 20011250                  11
 30010010                   2
 40016210                   4
```

Chapter 7. Header Specification

The header specification identifies the job as a sort job to the system. Only one header specification is used for each sort job. This chapter provides-reference information about the column entries required on the header specification for the system. Examples and additional explanations of some column entries for the header specification are provided in Chapter 2. For quick reference, a summarized chart of the column entries is given on the next page.

The following is an example of a header specification.



Header

H Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
1 2 3 4 5	6	7 8 9 10 11	12	13 14 15 16 17	18	19 20 21 22 23 24 25	26	27	28	29 30 31 32	33 34 35	36	37 38 39	40 41 42 43	44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	75 76 77 78 79 80
	H	S O R T														

Header Specification Column Entry Chart

The following chart shows the column entries for the header specification.

Columns	Entries	Purpose
1-2	00-99	Page Number.
3-5	000	Header statement number (no entry required).
6	H	Header statement identification.
7-12	SORTR	Regular sort.
	SORTRS	Summary sort.
	SORTA	Record address (or relative record number) sort.
12	E	Maintain original sequence in input file for records with identical control fields.
13-17	1-256	Control field length.
18	A	Write records into the output file in ascending order by control field.
	D	Write records into the output file in descending order by control field.
26	Blank	Use standard collating sequence in compare operation when sorting records.
	S	Use an alternative collating sequence for the entire control field when making comparisons.
	F	Use an alternative collating sequence for specified control fields when making comparisons.
27	0 or blank	Print: Sort specifications.
		Diagnostic messages.
		Program-status messages.
		Display: Action messages.
		Displayed messages.
	1	Print: Program-status messages.
		Display: Action and displayed messages.
	2	Print: Action messages only.
		Display: Displayed messages.
	3	Display: Displayed messages.
28	Blank	Write control field data in output file for SORTR and SORTRS jobs.
	X	Do not write control field data in output file for SORTR and SORTRS jobs.
29-32	1-4096	Output record length (Required for SORTR and SORTRS jobs.)
36	Blank	SORT-7724 message will be printed or displayed (No records are selected for sorting.)
	N	SORT-7724 message will not be printed or displayed (No records are selected for sorting.)
40-74	Any characters	For comments only.

Column Entries For Header Specification

Page Number (Columns 1 and 2) and Statement Number (Columns 3 through 5)

You can enter any numbers from 0 through 99 in columns 1 and 2 for page numbers.

The pages should be numbered in ascending order.

Because the statement number for the header specification is always 000 in columns 3 through 5, no entry is required in these columns.

As the sort program reads sort specifications, it checks the page and statement numbers to make sure they are in ascending order.

If the numbers are not in ascending order and specifications are being processed, the sort program warns you by placing an S (for sequence error) next to the statement that is out of sequence.

After displaying this warning, the sort program continues reading the specification statements, then issues a message (SORT-7725, warning: errors found) requiring a response from you. At this time, you can either continue or end the job.

Header Specification (Column 6)

Header specification (column 6) contains an H to identify the statement as a header specification. Because the H is already printed in the column, no entry is required.

Disk Sort Job (Columns 7 through 12)

Job (columns 7 through 12) identify the type of sorted output you want. You can specify the following:

- SORTR indicates regularly sorted output.
- SORTRS indicates sorted output with accumulated totals (or summary data).
- SORTA identifies the sorted output as record addresses (relative record numbers) only.

Identical or Equal Control Fields (Column 12)

For SORTR or SORTA jobs only, enter an E in column 12 to ensure that records having identical or equal control fields will be ordered as they appear in the input file(s). (For more information, see *Sorting Records with Identical Control Fields* in Chapter 2 of this manual.)

For ascending sorts, records with like control fields will be output in the order they appear in the input file.

For descending sorts, the records will be output in reverse order.

When you use equal control field ordering, you must add 3 to the length of the largest control field in columns 13 through 17 of the header specification. Sort adds a 3-byte relative record number to the end of each control field when this option is selected.

Control Field Lengths (Columns 13 through 17)

Enter the largest total of the control field lengths (taken from columns 9 through 16 of the field specifications).

If you have more than one record type, add the lengths of the control fields for each type of input record.

Enter the largest of the totals in columns 13 through 17, and right-adjust the entry. The total, including the 3 added if column 12 contains an E, cannot be more than 256.

For more information, see *Calculating Control Field Lengths* in Chapter 2 of this manual.

If column 12 contains an E, add 3 to the largest of the totals of the control field lengths and enter the total in columns 13 through 17, and right-adjust the entry.

Ascending or Descending Order (Column 18)

To specify the order of records in the output file, enter one of the following.

A	Ascending order by control field
D	Descending order by control field.

Collating Sequence (Column 26)

This column specifies the *collating sequence* to be used when data is compared to determine whether one character is equal to, greater than, or less than another character.

Blank The sort program will use the standard collating sequence (See the *Standard EBCDIC Collating Sequence Charts* in Appendix B.)

S Indicates that you will use an alternative sequence on the entire control field

F The sort program will change the standard collating sequence for specified control fields.

If you use an alternative collating sequence you must also supply ALTSEQ statements immediately after the header specification. See Appendix B for information about how to code ALTSEQ (alternative sequence) statements.

Note: Do not use packed or zoned factors in an include or omit record selection specification (P or U in column 8) if you specify an alternate collating sequence on the entire control field.

An F in column 26 changes the standard collating sequence for normal and/or opposite control fields only.

To change a normal or opposite control field, place an A in column 20 of the control field line on the field specifications. You must also supply ALTSEQ statements immediately following the header specifications.

The following chart shows the standard collating sequence. Other charts showing additional EBCDIC character arrangements are provided in Appendix B of this manual.

Order in the Sequence	**Character**	**Order in the Sequence**	**Character**
1 (lowest)	blank	25	#
2	¢	26	@
3	.	27	' (apostrophe)
4	<	28	=
5	(	29	"
6	+	30	a
7	|	31	b
8	&	32	c
9	!	33	d
10	$	34	e
11	*	35	f
12	)	36	g
13	;	37	h
14	¬	38	i
15	- (minus)	39	j
16	/	40	k
17	¦	41	l
18	,	42	m
19	%	43	n
20	___ (underscore)	44	o
21	>	45	p
22	?	46	q
23	\	47	r
24	:	48	~

Order in the Sequence	Character	Order in the Sequence	Character
49	s	73	O
50	t	74	P
51	u	75	Q
52	v	76	R
53	w	77	\
54	x	78	S
55	y	79	T
56	z	80	U
57	{	81	V
58	A	82	W
59	B	83	X
60	C	84	Y
61	D	85	Z
62	E	86	0
63	F	87	1
64	G	88	2
65	H	89	3
66	I	90	4
67	}	91	5
68	J	92	6
69	K	93	7
70	L	94	8
71	M	95 (highest)	9
72	N		

Print Option (Column 27)

Column 27 indicates the types of messages you can print during a job.

Note: Sort messages are discussed in the *System Messages* manual; however, examples of these messages are provided in Chapter 2 of this manual.

0 or Blank	Sort specifications Diagnostic messages Program-status messages Action messages Displayed messages
1	Program-status messages Action messages Displayed messages
2	Action messages Displayed messages
3	Displayed messages

It might be a good idea to specify a zero or a blank when you are testing your sort job, and a 3 for displaying messages after you have run the job successfully and have made it a part of your active programs.

Output Option (Column 28)

For SORTR and SORTRS jobs only, this entry indicates that you will either keep the data in control fields and put it into the output file or drop the control field data from the records after they are sorted. The entries are:

Blank	Keep control field data and write into output file
X	Drop control field data; do not write it into output file.

Output Record Length (Columns 29 through 32)

For SORTR and SORTRS jobs only, this entry specifies the length of records in the output file. The output records can be from 1 to 4096 characters long. (See *Calculating the Output Record Length* in Chapter 2 for more information.)

If you do not drop control fields, the output record length should include both the length of the control fields and the length of the specified data fields.

If you drop control fields, the output record length includes only the length of the specified data fields.

If you specify E (in column 12 of the H-specification) for equal fields:

- The control field length is increased by three bytes.
- The resulting output record length is increased by three bytes since the control field is added to the output data fields.

Null Output (Column 36)

This entry determines whether an error should be displayed if the sort program reads all the input records and no records were selected for processing. A blank in this column causes the sort program to display or print the following message:

```
SORT-7724 NO INPUT RECORDS INCLUDED.
```

N in this column prevents SORT-7724 from being displayed or printed.

If the message is printed or displayed during your sort job, you can select one of these options:

0 Create an empty output file.

3 Cancel the job without creating an output file.

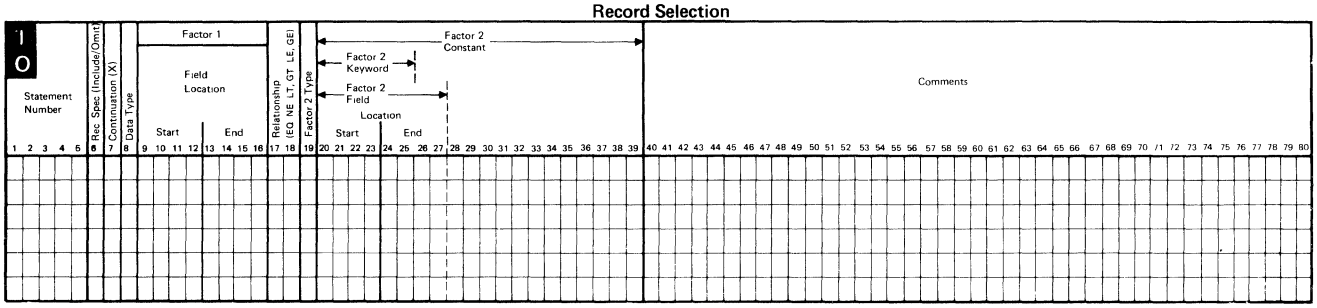
Comments (Columns 40 through 74)

These columns are for your *comments*. You can use any characters in these columns to document your sort job. If you specify that the sort program print your sort specifications, (the column 27 entry of the header specification is zero or blank), the comments you include in these columns are printed. Comments have no effect on the job, unless a plus sign (+) is the last character in a specification statement. The sort program interprets the statement following a plus sign (+) as a continuation of the preceding statement. A comment after the plus sign (+) will notify the program that the following statement is not a continuation.

Chapter 8. Record Selection Specifications

Record selection specifications identify to the sort program which records in a file you want sorted. Possible column entries are summarized in a chart on the next page for quick reference and are explained in this chapter. See Chapter 2 for examples of how you can use some column entries.

The following is an example of the record selection specifications on the sort specifications form.



Record Selection

I O

Statement Number

Rec Spec (Include/Omit)

Continuation (X)

Data Type

Factor 1

Field Location

Start End

Relationship (EQ NE LT, GT LE, GE)

Factor 2 Type

Factor 2 Constant

Factor 2 Keyword

Factor 2 Field

Location

Start End

Comments

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Record Selection Specifications Column Entry Chart

The following chart lists the column entries for the record selection specifications.

Columns	Entries	Purpose
1-2	00-99	Page Number.
3-5	01n-06n	Statement number. You can leave column 5 (n) blank, or enter any value to keep the specifications in ascending order.
6	I	Include statement.
	O	Omit statement.
7	Blank	This is the first statement of a set of I or O record type statements.
	A	AND statements. These specifications continue the definition of the record described on the preceding statement.
	O	OR statements. These specifications define a different type of record than the one on the previous statement.
	*	Comments.
8	C	Use both zone and digit portions of characters in compare operations.
	Z	Use only the zone portions of characters in compare operations.
	D	Use only the digit portions of characters in compare operations.
	P	Signed packed decimal data.
	U	Signed zoned decimal data.
9-12	1-4096	The input record position in which the factor 1 field starts (may be blank if field is only one position long).
13-16	1-4096	The input record position in which the factor 1 field ends.
17-18	EQ	Factor 1 must equal factor 2.
	NE	Factor 1 must not equal factor 2.
	LT	Factor 1 must be less than factor 2.
	GT	Factor 1 must be greater than factor 2.
	LE	Factor 1 must be less than or equal to factor 2.
	GE	Factor 1 must be greater than or equal to factor 2.
19	C	Factor 2 is a constant.
	F	Factor 2 is another field in the same input record.
	K	Factor 2 is a keyword.
20-23	1-4096	The input record position in which the factor 2 field starts (may be blank if field is only one position long).
24-27	1-4096	The input record position in which the factor 2 field ends.
20-39	Any characters	The characters making up the factor 2 constant.
40-74	Any characters	For comments only.

Page Number (Columns 1 and 2) and Statement Number (Columns 3 through 5)

Page number (columns 1 and 2) and columns 3 through 5 (statement number) form a 5-digit sequence number. As the sort program reads sort specifications, it checks the statement numbers to be sure they are in ascending order.

If the numbers are not in ascending order (if page 02 specifications come after page 01 specifications), and specifications are being issued, the sort program warns you by placing an S (for sequence error) next to statement 02. After issuing this warning, the sort program continues to read the specification statements, then issues a message requiring a response. At this time, you can either continue the job or end it.

Use column 5 when you want to insert a specification statement without numbering the other statements again.

Column 5 can be left blank, or you can enter any number to keep your specifications in ascending order. For example, to insert a specification statement between statements 01000 and 01010, number it 01005, enter it, and continue to fill out the remainder of the form. If you have more than six record type statements, you should use another coding form and start at line 01 of that form.

Be sure any statement that is out of sequence is clearly marked as such. You can do this by writing a note in the margin of the page with an arrow pointing to where the statement should be inserted.

Record Specifications for Include/Omit (Column 6)

Record specifications for Include/Omit (column 6) identifies the statement type for a sort job.

Enter an I in column 6 to specify an include statement. Enter an O to specify an omit statement. If you are sorting a file containing only one type of records, an entry is not required.

Sorting More Than One Type of Record in a Job

You can sort more than one type of record in a sort job using AND and OR entries in columns 6 and 7. Use AND when you describe data of the same type and test several conditions. Use OR when you describe data of different record types.

Continuation or Comments (Column 7)

Continuation or comments (column 7) indicates a statement's relationship to the statement that comes before it. To indicate relationships, the following entries can be made in column 7.

Blank Indicates the first statement of a set of include or omit statements. (The type of set is indicated by the column 6 entry: I for include or O for omit.)

A Indicates an AND relationship. This statement is a continuation of the preceding statement.

O Indicates an OR relationship. This statement applies to a different record type than the preceding statement, but the control field specifications for both are the same.

* Indicates comments. Comments are printed only if column 27 of the header specification contains a zero or blank.

Confusion may occur when *Include* and *Omit* specifications are combined with *And* and *Or* continuations in *NE* relationships. Use the following table for equivalencies.

Column 6	Column 7	Columns 17 and 18	is the same as	Column 6	Column 7	Columns 17 and 18
I		EQ		O		NE
I		NE		O		EQ
I	A	EQ		O	O	NE
I	A	NE		O	O	EQ
I	O	EQ		O	A	NE
I	O	NE		O	A	EQ

The following tables show the combination entries you can make in columns 6 and 7 for include and omit sets.

This table shows the entries for include sets.

Include Sets[1]

Type of Set	Column 6 Entry	Column 7 Entry	Explanation
Include AND statements	H, F, or O		Header statement, field statement, or omit statement.
	I	b̸	New record type indicated by a blank in column 7.
	I	A	Statements that describe the same record type (as the previous statement) have an A in column 7.
	F		Field statement(s).
Include OR statements	H, F, or O		Header statement, field statement, or omit statement:
	I	b̸	New record type indicated by a blank in column 7.
	I	O	Statements that describe a different record type (than the previous statement) have an O in column 7.
	F		Field statement(s).
Include AND and OR statements	H, F, or O		Header statement, field statement, or omit statement.
	I	b̸	New record type indicated by a blank in column 7.
	I	O	This statement designates a record type that is different from, but has the same field statements as, the record type described in the previous statement(s).
	I	A	This statement continues the same record type of a previous statement or statements. This record type can be continued (IA), or a different record type can be started (IO), provided all record types have the same field statements. Record types with different field statements would have to start a new include set.
	F		Field statement(s) for record types.
Include only one record type (implied include-all)	H		Header statement.
			No record type statements.
	F		Field statement(s).
Include-all	H, F, or O		Header statement, field statement, or omit statement.
	I		This statement tells the sort program to sort all of the records that have not been described by any preceding include and omit statements. Records referred to in this manner must have identical field specifications.
	F		Field statement(s).

Note: Records not described in include sets will not be sorted.

[1] Every include set must end with field statements. An include set can be followed by another include set, an omit set, or // END.

This table shows the entries you can make for omit sets.

Omit Sets[1]

Type of Set	Column 6 Entry	Column 7 Entry	Explanation
Omit AND statements (one record type)	H or F		Header statement or field statement (last statement of include set).
	O	b̸	New record type indicated by a blank in column 7.
	O	A	Statements that describe the same record type (as the previous statement) have an A in column 7.
Omit OR statements (different record types)	H or F		Header statement or field statement (last statement of include set).
	O	b̸	New record type indicated by a blank in column 7.
	O	O	Statements that describe a different record type (than the previous statement) have an O in column 7.
Omit AND and OR statement (different record types)	H or F		Header statement or field statement (last statement of include set).
	O	b̸	New record type indicated by a blank in column 7.
	O	A	Statements that describe the same record type (as the previous statement) have an A in column 7.
	O	O	Statements that describe a different record type (than the previous statement) have an O in column 7.

[1] There are no field statements in omit sets. Each omit set must be followed by an include or an include-all set.

Comparing Data (Column 8)

This entry determines how the sort program interprets data in factor 1 and factor 2 during compare operations.

For alphameric data:

Entry	Meaning	Maximum Length
C	Use both zone and digit portions of the characters.	256 characters
Z	Use only the zone portion of the character.	1 character
D	Use only the digit portions of the characters.	16 characters

For numeric data:

Entry	Meaning	Maximum Length
P (packed data)	Use the digit portions of numbers only (with a sign in the last digit).	8 bytes or 15 digits and a sign.
U (zoned data)	Use both the zone and digit portions of the number (with each zone and digit representing a value from 0 through 9).	16 digits and a sign.

Factor 1 Location (Columns 9 through 16)

These columns identify the locations of the factor 1 fields in the input records. Factor 1 fields are used in compare operations.

Start (columns 9 through 12) identify where the factor 1 field starts in the record. End (columns 13 through 16) identify where the factor 1 field ends.

To identify the location of more than one factor 1 field for the records you will describe, you should do two things:

1. Describe each field in a separate record selection statement.

2. Put an A in column 7 of every record selection statement (except the first); this indicates to the sort program that all the statements apply to the same record type.

Factor 1 field entries must be right-adjusted: the Start entry must end in column 12; the End entry must end in column 16.

To describe a factor 1 field that is only 1 character long, leave columns 9 through 12 (Start) blank and enter the number of the record position that contains the character in columns 13 through 16 (End).

Length of Factor 1

Factor 1 can contain from 1 to 256 characters. However, factor 1 cannot be longer than the length of the records being sorted or reformatted. The length of factor 1 is controlled by the column 8 entry.

The following chart lists the column 8 entries and the maximum lengths of factor 1.

Column 8	Maximum Factor 1 Field Length
C	256 characters
Z	1 character
D	16 characters
P	8 characters
U	16 characters

The following entries also control the length of factor 1 fields:

- When factor 2 is a constant, the length of factor 1 must be 20 characters or less.

- When factor 2 is a keyword, the length of factor 1 must be 6 if the keyword is UDATE, and 2 if the keyword is UMONTH, UDAY, or UYEAR.

Relation (Columns 17 and 18)

Relation (columns 17 and 18) are used to specify what the results of the comparison (between factor 1 and factor 2) must be. Enter one of the following to specify which relationship should exist between factor 1 and factor 2 for the records to be sorted:

Entry	Meaning
EQ	Factor 1 must equal factor 2.
NE	Factor 1 must not equal factor 2.
LT	Factor 1 must be less than factor 2.
GT	Factor 1 must be greater than factor 2.
LE	Factor 1 must be less than or equal to factor 2.
GE	Factor 1 must be greater than or equal to factor 2.

EQ and NE are the only entries you can use to compare zone portions of characters (Z in column 8).

If an alternative collating sequence on the entire control field (S in column 26 of the header specification) is used, both factor 1 and factor 2 are changed before the comparison is made.

Field, Constant, or Keyword (Column 19)

This entry identifies factor 2 as one of the following:

- Another field in the input record
- A constant
- A keyword that represents all or part of the program date.

Factor 2 is compared with factor 1 to determine which records should be sorted. You can enter the following in column 19 of the record selection specifications:

Entry	Meaning
C	Factor 2 is a constant. Enter the constant in columns 20 through 39, starting in column 20.
F	Factor 2 is a field. Use columns 20 through 27 to identify the location of the factor 2 field in the records.
K	Factor 2 is a keyword. Enter a K in column 19, and start the keyword in column 20.

Factor 2 Location (Columns 20 through 27)

This entry identifies the location of factor 2 when it is another field in the input records.

Start (columns 20 through 23) identify the starting position of the field.

End (columns 24 through 27) identify where the field ends.

The End and Start column entries must be right-adjusted: the Start entry ends in column 23; the End entry ends in column 27.

Factor 2 must also be the same length as the factor 1 field and in the same record as the factor 1 field.

Factor 2 can be used to compare a field containing one character. To describe fields that are only 1 character long, you can leave columns 20 through 23 (Start) blank, and enter the number of the record position that contains the character in columns 24 through 27 (End).

Factor 2 as a Constant (Columns 20 through 39)

When factor 2 is a constant, any arrangement of characters and blanks can be used as entries. Enter the constant in columns 20 through 39, starting in column 20.

When factor 2 is a constant:

- The constant must be the same length as the factor 1 field. If the constant is longer than the factor 1 field, sort prints a warning message.

- If factor 1 contains a packed number, the length of the constant (including the sign) must be twice the length of the factor 1 field.

- If the constant is an alphameric constant, it must be the same length as factor 1 and start in column 20.

  For alphameric constants, a D in column 8 indicates that only the digit portion of a character will be used in the compare operations.

- Numeric constants (U or P in column 8) must be right-adjusted within the field length specified in factor 1 (within twice the field length if factor 1 is a packed number).

  If factor 1 is a packed number, the last character in the constant must be its sign, plus (+) or minus (-). If factor 1 is a zoned number and the constant is a negative number, the last character in the constant must indicate both the numeric value of the last digit and the negative sign for the entire constant.

Note: If you specify a plus sign (+) as the last nonblank character in a statement, the sort program will be unable to determine the end of the statements preceding the plus sign and the beginning of the next statement (which could be another include statement or a field specification).

Factor 2 as a Keyword (Columns 20 through 39)

These columns identify factor 2 as a date keyword. The keyword specifies whether all or part of the program date should be compared with factor 1 (columns 7 through 16). When factor 2 is a keyword, the column 19 entry on the record selection specifications must be a K. The keyword starts in column 20; all unused columns through column 39 should be left blank.

The factor 2 keyword can be one of the following:

Keyword	**Part of Program Date**	**Factor 1 Field Length**
UDATE	Entire program date	6 characters
UMONTH	Month portion of program date	2 characters
UDAY	Day portion of program date	2 characters
UYEAR	Year portion of program date	2 characters

The factor 1 field length must be 6 for UDATE, and must be 2 for UMONTH, UDAY, or UYEAR.

If the UDATE keyword is used, the program date must be in the same format as the date contained in the input records.

If factor 2 is UDATE, record selection on or before, or on or after a certain date (columns 17 and 18 contain LT, GT, LE, or GE) only works with the international format (yymmdd). If the program date and the input records date are not in the international format (yymmdd), the keywords UYEAR, UMONTH,and UDAY, must be used to select the records.

Comments (Columns 40 through 74)

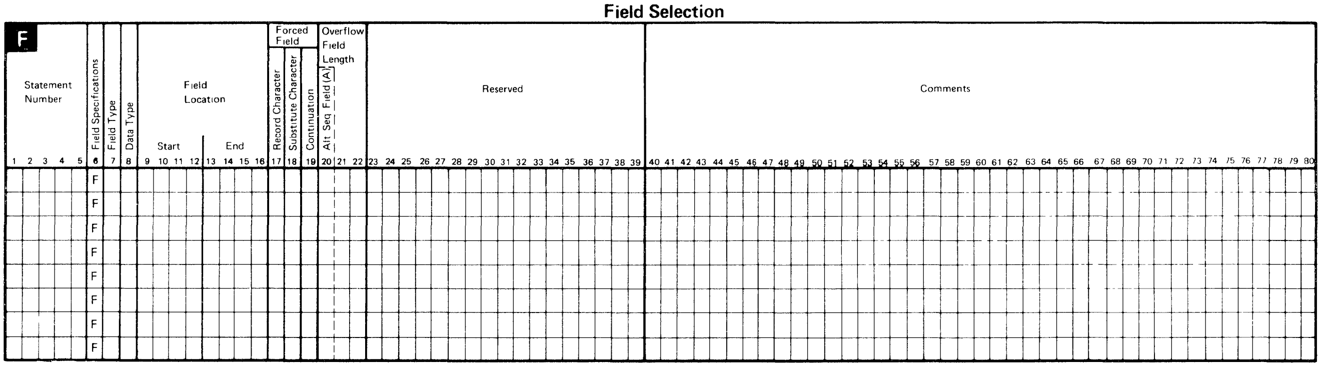
Comments (columns 40-74) are for your comments. If you specify that the sort program print your sort specifications (column 27 of the header line is either blank or contains a zero), any comments will be printed along with your sort specifications. Your comments have no effect on the program's operation.

Note: If you specify a plus sign (+) as the last nonblank character in a statement, the sort program will be unable to determine the end of the statements preceding the plus sign and the beginning of the next statement (which could be another include statement or a field specification).

Chapter 9. Field Selection Specifications

Field selection specifications tell the sort program how to arrange and format the input records in the work and output files. Possible column entries are summarized on the next page for quick reference and are explained in this chapter. Detailed explanations and examples for some entries are in Chapter 2, the guide portion of this manual.

The following is an example of the field selection specifications on a sort specifications form.



Field Selection

F Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length Alt Seq Field (A)	Overflow Field Length	Reserved	Comments
1 2 3 4 5	6	7	8	9 10 11 12	13 14 15 16	17	18	19	20	21 22	23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39	40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
	F											
	F											
	F											
	F											
	F											
	F											
	F											
	F											

Field Selection Column Entry Chart

Columns	Entries	Purpose
1-2	00-99	Page Number.
3-5	07n-14n	Statement number. You can leave column 5 (n) blank, or enter any value to keep the specifications in ascending order.
6	F	Field specification statement.
7	N	Normal control field.
	O	Opposite control field.
	F	Forced control field.
	D	Data field.
	S	Summary data field.
	*	Comments.
8	C	Use both zone and digit portions of characters in the field.
	Z	Use only zone portion of 1-character field.
	D	Use only digit portions of characters in the field.
	P	Signed packed decimal data.
	U	Signed zoned decimal data.
	V	Force a data character into the data field.
9-12	1-4096	Start position of a field in the record (can be blank if the field is 1 character long).
13-16	1-4096	End position of a field in the record.
17	Any character	Forced control fields; the character you want the sort program to change. Also used for summary overflow indicator fields; the character to be used for the overflow indicator.
18	Any character	Forced control or data field; the character you want the sort program to substitute. Also used for summary overflow indicator fields; the character to which the overflow field is initialized.
19	Blank	Forced control field statement is not a continuation of the preceding statement.
	Any character other than blank	Forced control field statement is a continuation of the preceding statement.
20-22	1-256	Overflow field length entry for summary sort only.
	A (column 20 only)	Alternative collating sequence by field.
23-39	Blank	Reserved for system use.
40-74	Any characters	Comments.

Field Selection Column Entries

Page Number (Columns 1 and 2) and Statement Number (Columns 3 through 5)

Page number (columns 1 and 2) and statement number (columns 3 through 5) form a 5-digit sequence number. As the sort program reads sort specifications, it checks the statement numbers to make sure they are in ascending order.

If the numbers are not in ascending order and specifications are being printed or displayed, the sort program places a warning (an S) next to the statement that is out of sequence. The S indicates a sequence error.

After issuing this warning, the sort program continues to read the specification statements, then issues a message requiring a response from you. At this time, you can either continue or end the job.

Column 5 can be left blank, or you can enter any value to keep your specifications in ascending order. If you have more than eight field lines, you should use another coding form and start at line 07 of that form.

Use column 5 when you want to insert a specification statement without numbering the other statements again. For example, to insert a specification statement between statements 01000 and 01010, you can number it 01005, enter it, and continue to fill out the specifications form.

Be sure any statements that are out of sequence are clearly marked. You can do this by writing a note in the margin of the page with an arrow pointing to where the insert belongs.

Field Specifications (Column 6)

Field specifications (column 6) contains an F, identifying the statement type as a field selection statement. Because an F is already printed on the sort specifications form, you need not make an entry in this column.

For all types of sorted output, field selection specifications describe the fields the sort program uses to sort the records (the control field). Also, for regular sort (SORTR) and summary sort (SORTRS), field selection statements describe the data that is written into the output file. For SORTRS sorts, field selection statements also describe the fields that are added together.

Field Type or Comments (Column 7)

Field type or comments (column 7) specifies whether you are describing a control field, a data field, or a comment statement. If you describe a control field, the column 7 entry indicates how the field will be used.

You can make the following entries in column 7.

Column 7 Entry — **Entry Explanations**

N — This is a normal control field. Sort this field so that the data from the field is in the order specified in column 18 of the header statement.

O — This is an opposite control field. Sort this field so the data from the field is in the order opposite that specified in column 18 of the header statement.

F — This is a forced control field. Change the control field according to the entries in columns 9 through 19. These forced control fields are allowed:

Forced without condition. Forcing a character into a control field before the records are sorted.

Forced with conditions. Forcing a character into a control field only when a condition is met.

Force-all. Forcing a character into a control field before the records are sorted if the control field does not contain one of several entries.

D — This is a data field. Use this entry for SORTR and SORTRS jobs only. (If you use a D entry in a SORTA job, the statement will be treated as a comment statement.)

S — This is a field containing data that is to be added together for all records with identical control fields. Use this entry for SORTRS sorts only. (If you use an S entry in a SORTA sort, the statement will be treated as a comment statement. If you specify an S in a SORTR sort, the field will be treated like a normal data field.)

* — This is a comment statement.

See *Specifying Control Fields* in Chapter 2 for more information and examples of how to use these entries.

The column 7 entry can be used with the column 8 entry to get certain results. See *Combination Entries (Columns 7 and 8)* in this chapter for a list of the ways you can combine the column 7 and column 8 entries.

Character Portions Used in Sorting Work Records (Column 8)

Your column 8 entry indicates what portion of the input record characters you want the sort program to use in building and sorting the work records. The column 8 entry is critical in ensuring that the sort generates the results you intend.

For unsigned alphameric data:

Column 8 Entry	**Character Portion Used**	**Maximum Field Length**
C	Use both zone and digit portions of the characters.	256 characters
Z	Use only the zone portion of the character.	1 character
D	Use only the digit portions of the characters.	16 characters

For signed numeric data:

Column 8 Entry	**Character Portion Used**	**Maximum Field Length**
P	Use the digit portions of characters only, with a sign in the last digit.	8 characters (or 15 digits and sign)
U	Use both the zone and digit portions of numbers, with each zone and digit representing a value from 0 through 9.	16 characters

For forced data:

Column 8 Entry	**Character Portion Used**	**Maximum Field Length**
V	Force a data character constant into the data field.	1 character

Combination Entries (Columns 7 and 8)

The following chart lists the combination of entries you can make in columns 7 and 8.

Column 7 Entry	Column 8 Entry	Maximum Field Length
N or O	C	256
	Z	1
	D	16
	P	8
	U	16
F	C	1
	Z	1
	D	1
D	C	256
	Z	1
	D	16
	P	8
	U	16
	V	1
S	C	256
	Z	1
	D	16
	P	8
	U	16
	V	1
*		Comments

Field Location (Columns 9 through 16)

Location (columns 9 through 16) describe the fields in the input record you will use in a sort job. You can describe both control fields and data fields.

The order in which you describe the fields in the field specifications determines their order in the sorted output records.

Start (columns 9 through 12) identify the field's starting position; columns 13 through 16 (End) identify the field's ending position.

The starting and ending locations entries should be right-adjusted. The Start entry ends in column 12; the End entry ends in column 16.

If you describe fields that are only 1 character long, you can leave columns 9 through 12 (Start) blank, and enter the number of the record position that contains the character in columns 13 through 16 (End). You should right-adjust this entry also.

The length of the field depends on the column 8 entry. See the column 8 section for the maximum field lengths possible for each entry.

Record Character (Column 17)

Make an entry in column 17 (record character) to conditionally force a character into a control field, or define a character as a summary overflow indicator.

For a Forced Control Field

The column 17 entry identifies the character in the input record (defined in columns 13 through 16) you want to replace. The sort program checks to see if the input record contains the character you specified in column 17. If it does, the character in column 18 replaces the specified character in the control field.

For a Summary Overflow Indicator Field

Enter the character you want to place in the output record if overflow occurs. If a blank is specified for a summary overflow indicator field, the sort program assumes an asterisk (*) should be placed in the field. Summary overflow indicator fields are valid only for summary sort (SORTRS) jobs. For more information about summary overflow indicator fields, see *Specifying a Summary Indicator* in Chapter 2.

Substitute Character (Column 18)

You should make an entry in column 18 *(substitute character)* only when you use a forced control field or a forced data field. The character in column 18 either replaces the control field character you specify in column 17, adds a new character to the control field, or adds a new character to the data field.

Also, if you want to use substitute characters, you can force a 1-character field only. Only entire characters can be forced into another field.

You can enter a character in column 18 to specify a field as a summary overflow indicator field. The column 18 entry, in this case, is the character to which the overflow field is set if overflow does not occur. Any character can be used to define the field as a summary overflow indicator field. Summary overflow indicator fields are valid only for summary sort (SORTRS) jobs.

Continuation (Column 19)

Any character (except blank) entered in column 19 specifies that that statement refers to the same control field in the work record as the preceding statement.

For example, if a control field in the input record can contain any one of several characters and you want to specify replacements for more than one of those characters, you could use a separate statement to define each character you want to replace and the character you want to replace it with. An entry in column 19 indicates that you will continue replacing characters.

Overflow Field Length (Columns 20 through 22)

Use columns 20 through 22 with a file of accumulated totals (SORTRS) to eliminate the possibility of an overflow condition in a summary data field.

To eliminate a possible overflow condition, increase the length of the field and place the entry for the new length in columns 20 through 22. The overflow field length entry should:

- Equal the length of the summary data field plus the expected overflow length

- Be right-adjusted to column 22

- Not exceed the maximum field length (For information about the maximum field length, see *Field Location (Columns 9 through 16)* earlier in this chapter.)

The summary data in the output record is right-adjusted. A maximum of 24 fields can be summarized for each record type in a sort job.

If packed fields are summarized, columns 20 through 22 should specify the number of bytes of packed data contained in the field.

Alternative Collating Sequence by Field (Column 20)

Column 20 must contain an A for any normal or opposite control field that is to be altered by the alternative collating sequence when column 26 of the header line contains an F.

If you specify an alternative collating sequence for a particular field, that field will be changed according to the alternative collating sequence whenever it is used again as a control field for that record type.

Comments (Columns 40 through 74)

Columns 40 through 74 are for your comments. Any characters can be used in these columns. Your comments have no effect on how the program works unless a plus sign (+) is the last character in a specification statement. In this case, your sort job might generate unpredictable results. If you specify that the sort program print your sort specifications (column 27 of the header specification is either blank or contains a zero), comments are printed along with your sort specifications.

Appendix A. Calculating the Sizes of Files for Sort

Output File

Use this formula to calculate how many blocks the output file needs; round up the result to the nearest whole number.

$$\text{BLOCKS} = \frac{\begin{pmatrix}\text{Number of Records that will}\\ \text{be Selected for Sorting}\end{pmatrix} \times \begin{pmatrix}\text{Output}\\ \text{Record Length}\end{pmatrix}}{\text{Number of Bytes in a Block} = 2560}$$

If you use the RECORDS parameter in the output file statement, use this formula:

$$\text{RECORDS} = \begin{array}{l}\text{Number of Records that will}\\ \text{be Selected for Sorting}\end{array}$$

Work File

Use either of the following formulas to determine approximately how many blocks or records to specify in the FILE statement for the work file; round up the result to the nearest whole number.

$$\text{BLOCKS} = 2+\left(\frac{\begin{pmatrix}\text{Number of Records that}\\ \text{will be Selected for Sorting}\end{pmatrix} \text{ x } \begin{pmatrix}\text{Work Record}\\ \text{Length}\end{pmatrix} \text{ x } 1.1}{\text{Number of Bytes in a Block} = 2560}\right)$$

or

$$\text{RECORDS} = \begin{matrix}\text{Number of Records that will}\\ \text{be Selected for Sorting}\end{matrix} + \left(\frac{2 \text{ x} \begin{pmatrix}\text{Number of Bytes in}\\ \text{a Block (2560)}\end{pmatrix}}{\text{Work Record Length}}\right)$$

Note: The sort program includes an 8-byte work block vector in every work block written to the work file. Therefore, the work record must be larger than a file that would hold the exact number of records to be sorted. This is why both formulas provide only an approximate number of records or blocks to specify in the FILE statement for the work file.

Work Record Length

Type of Sort	Type of Output	Work Record Length
SORTA	Record addresses	Length of control fields plus 3
SORTR or SORTRS	Control fields only	Length of control fields as specified in the header statement
	Control fields and data	Length of data plus length of control fields (output record length specified in the header statement)
	Data only	Length of data plus length of control fields (output record length specified in the header statement plus the control field length specified in the header statement)

Appendix B. Collating Sequences

The sort program uses a collating sequence to compare the characters in the control fields in the input records to determine whether one character is equal to, greater than, or less than another character and then sort your input records in the order specified. Collating sequences are logical sequences used to put items of data into a particular order.

Usually, the sorted output records will be ordered according to the Standard EBCDIC Collating Sequence. However, you can change the collating sequence for all or selected characters by specifying an Alternative Collating Sequence. Use the header specification to define the order in which records in the input file will be sorted.

This appendix provides examples of the Standard EBCDIC Collating Sequence charts and explanations and examples of how to define an alternative collating sequence.

Standard Collating Sequence

The Standard EBCDIC Collating Sequences are arrangements of data based on the EBCDIC character set. There are variations in the standard collating sequences, depending on the following:

- Whether you use both the zone and digit portions of characters used in the compare operation
- Whether you use the zone portions of characters only in compare operations
- Whether you use the digit portions of characters only in compare operations.

The following charts show the complete collating sequence for each situation.

The following chart shows the standard collating sequence. Other charts showing additional EBCDIC character arrangements are provided in other charts in this Appendix.

Order in the Sequence	Character	Order in the Sequence	Character
1 (lowest)	blank	25	#
2	¢	26	@
3	.	27	' (apostrophe)
4	<	28	=
5	(	29	"
6	+	30	a
7	|	31	b
8	&	32	c
9	!	33	d
10	$	34	e
11	*	35	f
12	)	36	g
13	;	37	h
14	¬	38	i
15	- (minus)	39	j
16	/	40	k
17	¦	41	l
18	,	42	m
19	%	43	n
20	___ (underscore)	44	o
21	>	45	p
22	?	46	q
23	\	47	r
24	:	48	~

Order in the Sequence	Character	Order in the Sequence	Character
49	s	73	O
50	t	74	P
51	u	75	Q
52	v	76	R
53	w	77	\
54	x	78	S
55	y	79	T
56	z	80	U
57	{	81	V
58	A	82	W
59	B	83	X
60	C	84	Y
61	D	85	Z
62	E	86	0
63	F	87	1
64	G	88	2
65	H	89	3
66	I	90	4
67	}	91	5
68	J	92	6
69	K	93	7
70	L	94	8
71	M	95 (highest)	9
72	N		

This chart shows the Standard EBCDIC Collating Sequence sort uses when comparing the entire character, both the zone and digit portions.

When Both Zone and Digit Portions of Characters are Used			When Both Zone and Digit Portions of Characters are Used			When Both Zone and Digit Portions of Characters are Used		
Order in the Sequence[1]	Character	Corresponding Hexadecimal Number[2]	Order in the Sequence[1]	Character	Corresponding Hexadecimal Number[2]	Order in the Sequence[1]	Character	Corresponding Hexadecimal Number[2]
1 (lowest)	blank	40	33	d	84	65	H	C8
2	¢	4A	34	e	85	66	I	C9
3		4B	35	f	86	67	}	D0
4	<	4C	36	g	87	68	J	D1
5	(	4D	37	h	88	69	K	D2
6	+	4E	38	i	89	70	L	D3
7	|	4F	39	j	91	71	M	D4
8	&	50	40	k	92	72	N	D5
9	!	5A	41	l	93	73	O	D6
10	$	5B	42	m	94	74	P	D7
11	*	5C	43	n	95	75	Q	D8
12	)	5D	44	o	96	76	R	D9
13	;	5E	45	p	97	77	\	E0
14	¬	5F	46	q	98	78	S	E2
16	- (minus)	60	47	r	99	79	T	E3
17	/	61	48	~	A1	80	U	E4
15	¦	6A	49	s	A2	81	V	E5
18	,	6B	50	u	A3	82	W	E6
19	%	6C	51	u	A4	83	X	E7
20	_ (underscore)	6D	52	v	A5	84	Y	E8
21	>	6E	53	w	A6	85	Z	E9
22	?	6F	54	x	A7	86	0	F0
23	\	79	55	y	A8	87	1	F1
24	:	7A	56	z	A9	88	2	F2
25	#	7B	57	{	C0	89	3	F3
26	@	7C	58	A	C1	90	4	F4
27	' (apostrophe)	7D	59	B	C2	91	5	F5
28	=	7E	60	C	C3	92	6	F6
29	!	7F	61	D	C4	93	7	F7
30	a	81	62	E	C5	94	8	F8
31	b	82	63	F	C6	95	9	F9
32	c	83	64	G	C7	(highest)		

[1] When several characters share the same position in the sequence, they are considered equal. For example, if you are using only the digit portions of characters, b, k, s, B, K, S, and 2 (position 3) are considered equal.

[2] This is the number you use in ALTSEQ statements to identify a character that you want to shift to a different order in the sequence.

This chart shows the Standard EBCDIC Collating Sequence sort uses when comparing only the zone portion of characters.

When Only the Zone Portion of Characters is Used		
Order in the Sequence[1]	Character	Corresponding Hexadecimal Number[2]
1 (lowest)	¢	4A
	.	4B
	<	4C
	(	4D
	+	4E
	|	4F
2	!	5A
	$	5B
	*	5C
	)	5D
	;	5E
	¬	5F
3	/	61
	¦	6A
	,	6B
	%	6C
	_ (underscore)	6D
	>	6E
	?	6F
4	`	79
	:	7A
	#	7B
	@	7C
	' (apostrophe)	7D
	=	7E
	"	7F
5	a	81
	b	82
	c	83
	d	84
	e	85
5 (cont.)	f	86
	g	87
	h	88
	i	89
6	j	91
	k	92
	l	93
	m	94
	n	95
	o	96
	p	97
	q	98
	r	99
7	~	A1
	s	A2
	t	A3
	u	A4
	v	A5
	w	A6
	x	A7
	y	A8
	z	A9
8	&	50
	{	C0
	A	C1
	B	C2
	C	C3
	D	C4
	E	C5
	F	C6
	G	C7
	H	C8
	I	C9
9	- (minus)	60
	}	D0
	J	D1
	K	D2
	L	D3
	M	D4
	N	D5
	O	D6
	P	D7
	Q	D8
	R	D9
10	\	E1
	S	E2
	T	E3
	U	E4
	V	E5
	W	E6
	X	E7
	Y	E8
	Z	E9
11 (highest)	blank	40
	0	F0
	1	F1
	2	F2
	3	F3
	4	F4
	5	F5
	6	F6
	7	F7
	8	F8
	9	F9

[1] When several characters share the same position in the sequence, they are considered equal. For example, if you are using only the digit portions of characters, b, k, s, B, K, S, and 2 (position 3) are considered equal.

[2] This is the number you use in ALTSEQ statements to identify a character that you want to shift to a different order in the sequence.

This chart shows the Standard EBCDIC Collating Sequence sort uses when comparing only the digit portion of characters.

When Only the Digit Portion of Characters is Used		
Order in the Sequence[1]	Character	Corresponding Hexadecimal Number[2]
1 (lowest)	blank	40
	&	50
	- (minus)	60
	{	C0
	}	D0
	0	F0
2	/	61
	a	81
	j	91
	~	A1
	A	C1
	J	D1
	\	E1
	1	F1
3	b	82
	k	92
	s	A2
	B	C2
	K	D2
	S	E2
	2	F2
4	c	83
	l	93
	t	A3
	C	C3
	L	D3
	T	E3
	3	F3
5	d	84
	m	94
	u	A4
	D	C4
	M	D4
	U	E4
	4	F4
6	e	85
	n	95
	v	A5
	E	C5
	N	D5
	V	E5
	5	F5
7	f	86
	o	96
	w	A6
	F	C6
	O	D6
	W	E6
	6	F6
8	g	87
	p	97
	x	A7
	G	C7
	P	D7
	X	E7
	7	F7
9	h	88
	q	98
	y	A8
	H	C8
	Q	D8
	Y	E8
	8	F8
10	`	79
	i	89
	r	99
	z	A9
	I	C9
	R	D9
	Z	E9
	9	F9
11	¢	4A
	!	5A
	¦	6A
	:	7A
12	.	4B
	$	5B
	,	6B
	#	7B
13	<	4C
	*	5C
	%	6C
	@	7C
14	(	4D
	)	5D
	_ (underscore)	6D
	' (apostrophe)	7D
15	+	4E
	;	5E
	>	6E
	=	7E
16 (highest)	|	4F
	¬	5F
	?	6F
	"	7F

[1] When several characters share the same position in the sequence, they are considered equal. For example, if you are using only the digit portions of characters, b, k, s, B, K, S, and 2 (position 3) are considered equal.

[2] This is the number you use in ALTSEQ statements to identify a character that you want to shift to a different order in the sequence.

Defining an Alternative Collating Sequence

To define an alternative collating sequence, you should do the following:

1. Indicate on the header (the alternative collating sequence column, alt.coll.seq.) and field (alternative collating sequence field, alt.seq. field column) specifications that a sequence other than the normal one is to be used.

2. Determine the change you want to make in the standard collating sequence.

3. Define your changes to the collating sequence on the *Translation Table and Alternative Collating Sequence Coding Form.*

4. Define your changes using the ALTSEQ (alternative sequence) statement.

When you specify an alternative collating sequence, you want the records in the output field to be sorted in an order different from the orders permitted by the standard collating sequences.

You can specify an alternative collating sequence for the following:

- The entire control field

- Specified control fields.

Specifying an Alternative Collating Sequence on the Entire Control Field

To specify an alternative collating sequence on the entire control field:

1. Enter an S in column 26 of the header specification.

2. Code an ALTSEQ (alternative sequence) statement immediately following the header specification statement.

When an alternative collating sequence on the entire control field is used, the sort program first changes the entire input record to the specified alternative sequence of characters. Therefore, any include or omit record type checks will be against the alternative sequence data.

ALTSEQ statements also change the following:

- Factor 1 and factor 2 (including record type constants)

- Normal and opposite control fields

- Input field characters that condition forced control fields.

Specifying an Alternative Collating Sequence on Certain Control Fields

To specify an alternative collating sequence for specified normal and opposite control fields:

1. Place an F in column 26 of the header specification.

2. Place an A in column 20 of the control field statement of the field specifications (Alt. Seq. Field (A) for each normal or opposite control field you want to change).

3. Supply ALTSEQ statements immediately following the header specification.

The following can be helpful when specifying an alternative collating sequence on specified control fields:

- Record selection (including and omitting records) and conditional force (replacing single or all characters) are based on an input record that has not been changed by the alternative collating sequence.

- Any control field that has an A specified in column 20 should not be packed or zoned (P or U in column 8). Any control field that does not have an A specified in column 20 can be packed or zoned.

- Packed or zoned factor 1 and factor 2 can be specified in include or omit record selection specifications (P or U in column 8).

- If you specify an alternative collating sequence for a particular field, that field will be changed according to the alternative collating sequence whenever it is used again as a control field for that record type. This would occur only if the same input field were specified more than once as a control field.

When an alternative collating sequence on specified control fields is used, ALTSEQ statements change only specified normal and opposite control fields.

ALTSEQ statements never change data fields in records or forced control field characters.

Note: You should not use packed or zoned factor 1 and factor 2 in an include or omit record selection specification (P or U in column 8) if you specify an alternative collating sequence on the entire control field (S in column 26 of the header specification). However, you can use packed or zoned factor 1 and factor 2 fields in include and omit record selection specifications if you use an alternative collating sequence on specified control fields (F in column 26 of the header specification). Characters defined for an alternative collating sequence could be interpreted as other data fields.

Order of Sequence Specifications When Using An Alternative Collating Sequence

ALTSEQ statements follow the header specification statement. When you use an alternative collating sequence, your sort specifications must be in this order:

1. OCL statements

2. Sort specifications

   a. Header specification statement
   b. ALTSEQ statements
   c. **
   d. Record selection specification(s) statement(s)
   e. Field selection specifications statements

3. // END

When you move a character into the sequence position normally assigned to another character, both the new and the original character occupy the same position and are considered equal. If you do not want the two characters to be equal, you must also move the character that normally occupies that position (see the examples later in this appendix).

Use the following steps to code ALTSEQ statements.

1. Code ALTSEQ in the first six positions to tell the sort program that you want to change the standard collating sequence.

1	2	3	4	5	6	7	8	9	10	11	12	13
A	L	T	S	E	Q							

2. Leave the next two positions blank.

						7	8					
A	L	T	S	E	Q							

3. Enter the hexadeximal equivalent of the character you are taking out of its normal sequence.

								9	10			
A	L	T	S	E	Q			h	n			

4. Enter the hexadecimal equivalent of the value that the character specified in columns 9 through 10 will assume in the collating sequence.

								9	10	11	12	
A	L	T	S	E	Q			h	n	h	n	

5. Enter as many pairs (from Steps 3 and 4) as the number of characters you are taking out of normal sequence.

6. Leave no spaces between sets of hexadecimal numbers.

A	L	T	S	E	Q			h	n	h	n	h	n	h	n		

7. When you reach the end of one statement, you can continue on the next specification statement line (follow Steps 1 through 6).

8. Enter two asterisks in positions 1 and 2 to indicate the end of the ALTSEQ statements.

A	L	T	S	E	Q			n	n	n	n			Maximum of 96 Positions				
A	L	T	S	E	Q			n	n	n	n			Maximum of 96 Positions				
A	L	T	S	E	Q			n	n	n	n			Maximum of 96 Positions				
*	*																	

The following examples show how the collating sequence is altered so that:

1. A special character is inserted between two alphabetic characters.

2. Two characters have the same position in the sequence. (This means they are considered equal.)

Example of Altering Normal Collating Sequence

You may alter the normal collating sequence in a number of ways. For example, you may insert a character between two existing characters, you may take a character out of the sequence, or you may change characters (put A where Z is and Z where A is). Regardless of how you alter the sequence, you must specify every character that is to be changed by the alteration. For example, if you want the dollar sign ($) to be positioned in the collating sequence between A and B, the normal sequence is changed as follows:

Normal Sequence	**Altered Sequence**
A	A
B	$
C	B
D	C
E	D
F	E
G	F
H	G
I	H
	I

Notice on the *Translation Table and Alternative Collating Sequence Coding Form* that there are many characters between I and , R and S, Z and O. These characters can be represented in the computer by certain bit combinations. However, they have no printable graphic symbol. Because of this particular arrangement of graphics, nongraphics, graphics, and so on in the collating sequence, a character, when inserted between A and B, changes only the position of graphics B through I. All other graphics are not affected. B through I all move down one position causing the I to take the place of the nongraphic represented by hexadecimal CA. This does not matter, however, since the original character CA cannot be printed anyway. See the *Translation Table and Alternative Collating Sequence Coding Form* example for the entries on the form.

The alternative sequence input is defined as follows:

Column	**Entry**
1 through 6	ALTSEQ
7 and 8	Blanks
9 through 12	5BC2 ($ takes B′s position)
13 through 16	C2C3 (B takes C′s position)
17 through 20	C3C4 (C takes D′s position)
21 through 24	C4C5 (D takes E′s position)
25 through 28	C5C6 (E takes F′s position)
29 through 32	C6C7 (F takes G′s position)
33 through 36	C7C8 (G takes H′s position)
37 through 40	C8C9 (H takes I′s position)
41 through 44	C9CA (I is given a new position held by no other printable character)

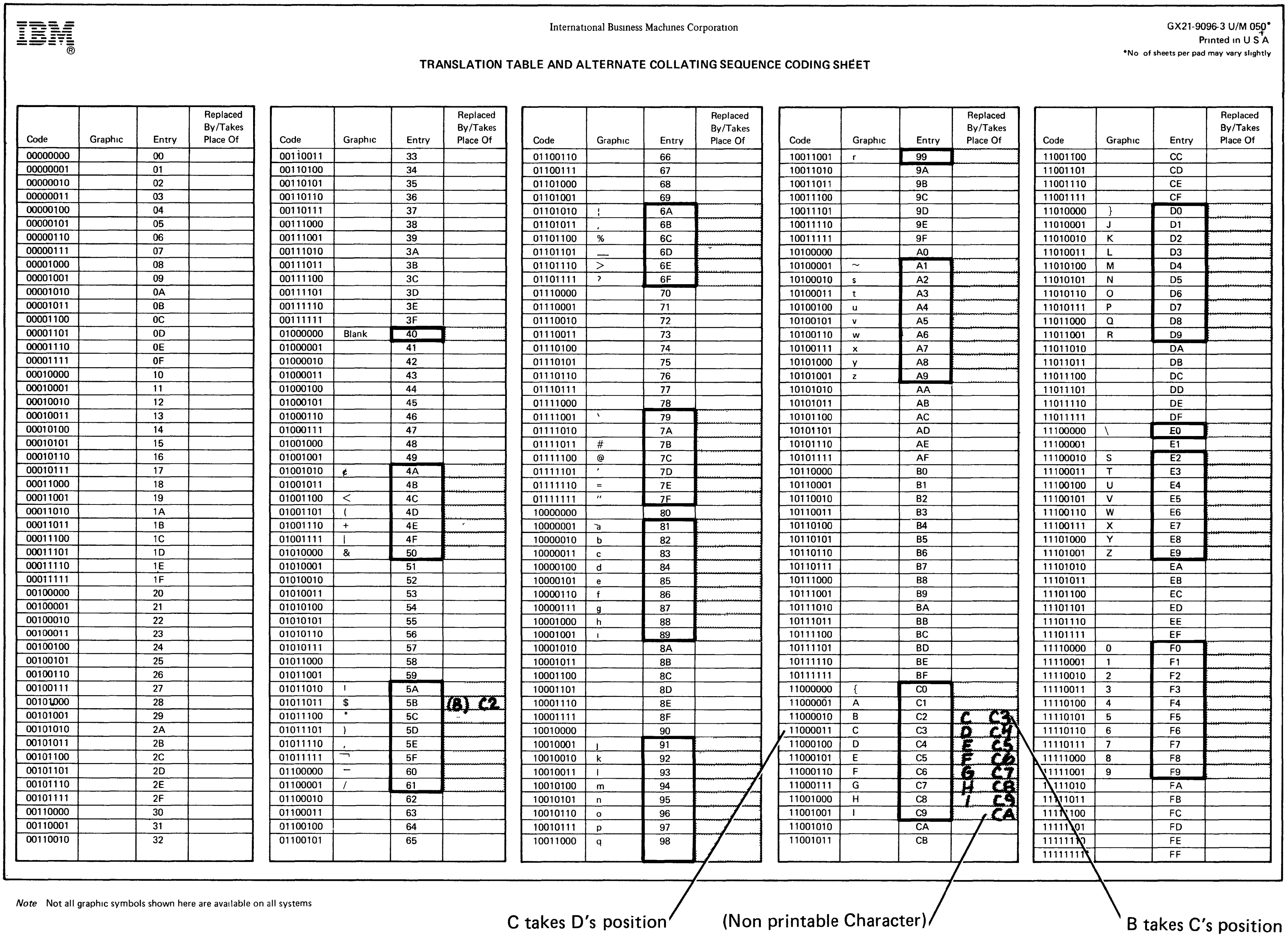
Example of Making Characters Equal

If you want one character to be considered the same as another character, the characters must hold the same position in the collating sequence. For example, you may wish a blank to be considered a zero. Therefore, you need to define an alternative collating sequence in which the blank is the same as the zero because it holds the same position in the sequence. The alternative collating sequence input card looks like this:

Column	**Entry**
1 through 6	ALTSEQ
7 and 8	Blanks
9 through 12	40F0 (blank takes the zero′s position)

Now whenever a blank is read and used in a comparison, it is considered a zero. Thus, if you were comparing numbers to 0036 to find an equal condition, 0036 and bb36 (where b = blank) both compare equal to 0036.

Note: Exercise care when using D (digit) or U (unpacked) fields with an alternative sequence. A sign in a D or U field will form a character which may be one of the characters being translated.



International Business Machines Corporation

GX21-9096-3 U/M 050*
Printed in U S A
*No of sheets per pad may vary slightly

TRANSLATION TABLE AND ALTERNATE COLLATING SEQUENCE CODING SHEET

Code	Graphic	Entry	Replaced By/Takes Place Of
00000000		00	
00000001		01	
00000010		02	
00000011		03	
00000100		04	
00000101		05	
00000110		06	
00000111		07	
00001000		08	
00001001		09	
00001010		0A	
00001011		0B	
00001100		0C	
00001101		0D	
00001110		0E	
00001111		0F	
00010000		10	
00010001		11	
00010010		12	
00010011		13	
00010100		14	
00010101		15	
00010110		16	
00010111		17	
00011000		18	
00011001		19	
00011010		1A	
00011011		1B	
00011100		1C	
00011101		1D	
00011110		1E	
00011111		1F	
00100000		20	
00100001		21	
00100010		22	
00100011		23	
00100100		24	
00100101		25	
00100110		26	
00100111		27	
00101000		28	
00101001		29	
00101010		2A	
00101011		2B	
00101100		2C	
00101101		2D	
00101110		2E	
00101111		2F	
00110000		30	
00110001		31	
00110010		32	

Code	Graphic	Entry	Replaced By/Takes Place Of
00110011		33	
00110100		34	
00110101		35	
00110110		36	
00110111		37	
00111000		38	
00111001		39	
00111010		3A	
00111011		3B	
00111100		3C	
00111101		3D	
00111110		3E	
00111111		3F	
01000000	Blank	40	
01000001		41	
01000010		42	
01000011		43	
01000100		44	
01000101		45	
01000110		46	
01000111		47	
01001000		48	
01001001		49	
01001010	¢	4A	
01001011	.	4B	
01001100	<	4C	
01001101	(	4D	
01001110	+	4E	
01001111	|	4F	
01010000	&	50	
01010001		51	
01010010		52	
01010011		53	
01010100		54	
01010101		55	
01010110		56	
01010111		57	
01011000		58	
01011001		59	
01011010	!	5A	
01011011	$	5B	(B) C2
01011100	*	5C	
01011101	)	5D	
01011110	;	5E	
01011111	¬	5F	
01100000	–	60	
01100001	/	61	
01100010		62	
01100011		63	
01100100		64	
01100101		65	

Code	Graphic	Entry	Replaced By/Takes Place Of
01100110		66	
01100111		67	
01101000		68	
01101001		69	
01101010	¦	6A	
01101011	,	6B	
01101100	%	6C	
01101101	_	6D	
01101110	>	6E	
01101111	?	6F	
01110000		70	
01110001		71	
01110010		72	
01110011		73	
01110100		74	
01110101		75	
01110110		76	
01110111		77	
01111000		78	
01111001	\`	79	
01111010		7A	
01111011	#	7B	
01111100	@	7C	
01111101	'	7D	
01111110	=	7E	
01111111	"	7F	
10000000		80	
10000001	a	81	
10000010	b	82	
10000011	c	83	
10000100	d	84	
10000101	e	85	
10000110	f	86	
10000111	g	87	
10001000	h	88	
10001001	i	89	
10001010		8A	
10001011		8B	
10001100		8C	
10001101		8D	
10001110		8E	
10001111		8F	
10010000		90	
10010001	j	91	
10010010	k	92	
10010011	l	93	
10010100	m	94	
10010101	n	95	
10010110	o	96	
10010111	p	97	
10011000	q	98	

Code	Graphic	Entry	Replaced By/Takes Place Of
10011001	r	99	
10011010		9A	
10011011		9B	
10011100		9C	
10011101		9D	
10011110		9E	
10011111		9F	
10100000		A0	
10100001	~	A1	
10100010	s	A2	
10100011	t	A3	
10100100	u	A4	
10100101	v	A5	
10100110	w	A6	
10100111	x	A7	
10101000	y	A8	
10101001	z	A9	
10101010		AA	
10101011		AB	
10101100		AC	
10101101		AD	
10101110		AE	
10101111		AF	
10110000		B0	
10110001		B1	
10110010		B2	
10110011		B3	
10110100		B4	
10110101		B5	
10110110		B6	
10110111		B7	
10111000		B8	
10111001		B9	
10111010		BA	
10111011		BB	
10111100		BC	
10111101		BD	
10111110		BE	
10111111		BF	
11000000	{	C0	
11000001	A	C1	
11000010	B	C2	C C3
11000011	C	C3	D C4
11000100	D	C4	E C5
11000101	E	C5	F C6
11000110	F	C6	G C7
11000111	G	C7	H C8
11001000	H	C8	I C9
11001001	I	C9	CA
11001010		CA	
11001011		CB	

Code	Graphic	Entry	Replaced By/Takes Place Of
11001100		CC	
11001101		CD	
11001110		CE	
11001111		CF	
11010000	}	D0	
11010001	J	D1	
11010010	K	D2	
11010011	L	D3	
11010100	M	D4	
11010101	N	D5	
11010110	O	D6	
11010111	P	D7	
11011000	Q	D8	
11011001	R	D9	
11011010		DA	
11011011		DB	
11011100		DC	
11011101		DD	
11011110		DE	
11011111		DF	
11100000	\	E0	
11100001		E1	
11100010	S	E2	
11100011	T	E3	
11100100	U	E4	
11100101	V	E5	
11100110	W	E6	
11100111	X	E7	
11101000	Y	E8	
11101001	Z	E9	
11101010		EA	
11101011		EB	
11101100		EC	
11101101		ED	
11101110		EE	
11101111		EF	
11110000	0	F0	
11110001	1	F1	
11110010	2	F2	
11110011	3	F3	
11110100	4	F4	
11110101	5	F5	
11110110	6	F6	
11110111	7	F7	
11111000	8	F8	
11111001	9	F9	
11111010		FA	
11111011		FB	
11111100		FC	
11111101		FD	
11111110		FE	
11111111		FF	

Note Not all graphic symbols shown here are available on all systems

C takes D's position

(Non printable Character)

B takes C's position

▭ Hexadedimal numbers of characters in the standard collating sequences.

⬚ Where you record the hexadecimal number of the character you are going to put in that relative position in the sequence.

Appendix C. Calling Sort from an Assembler Program

You can call the sort program directly from an assembler program. The assembler macroinstructions $SORT and $SRT, can be used to call the sort program and generate the sort parameter list. See the *Programming with Assembler* manual for information about $SORT and $SRT. For information about the sort parameter list, see *Sort Parameter List* in this appendix. When this sort interface is used, the sort specifications must be either in a source member or in the sort parameter list.

Note: If you have a COBOL program that uses sort, see the *Programming with COBOL* manual for more information about calling the sort program.

Remember the following when using this sort interface:

- All input and output files used in the sort job must be defined by FILE statements and must be closed before the sort program is called. (See the *System Reference* manual for an explanation of how to specify FILE statements.)

- If a sort job does not complete successfully (hexadecimal 10), control is returned to the assembler program with bit 3 of the indicator byte on in the sort parameter list.

Sort Parameter List

The size (in bytes) of the sort parameter list varies depending upon:

- Whether the sort specifications are in a source member or in the parameter list

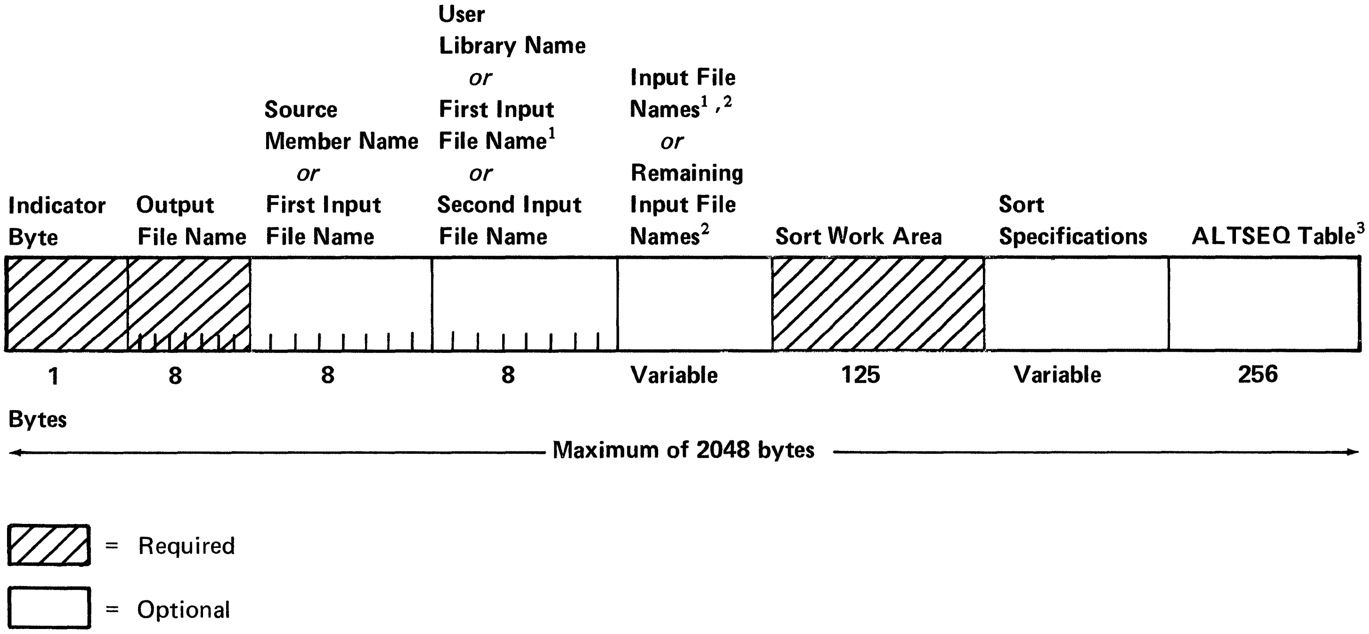
- The number of input file names being passed to the sort program

- The number of sort specifications, if they are in the parameter list

- Whether or not an ALTSEQ table is in the parameter list.

The parameter list can contain a maximum of 2048 bytes.

An overview of the sort parameter list follows.



Indicator Byte	Output File Name	Source Member Name *or* First Input File Name	User Library Name *or* First Input File Name[1] *or* Second Input File Name	Input File Names[1,2] *or* Remaining Input File Names[2]	Sort Work Area	Sort Specifications	ALTSEQ Table[3]
▨	▨	□	□	□	▨	□	□
1	8	8	8	Variable	125	Variable	256

Bytes

← Maximum of 2048 bytes →

▨ = Required

□ = Optional

[1] At least one input file name must be passed to the sort program.

[2] Each input file name (up to a maximum of eight) must be 8 bytes long.

[3] The ALTSEQ table must be in bytes 1793 through 2048.

An explanation of the contents of the parameter list follows.

The indicator byte, the first byte of the parameter list, contains indicators in the first four bits and the number of input files in the last four bits:

Indicator Byte

Bit 0:
On: Sort specifications are in a source member.
Off: Sort specifications are in the parameter list.

If bit 0 is off, bit 1 also must be off.

Bit 1:
On: Source member is in a user library.
Off: The source member containing the sort specifications is in #LIBRARY, or the sort specifications are in the parameter list.

The wrong sort specifications will be used by the sort program if the user library is not on disk and the source member in the system library is not the one desired.

Bit 2:

On: An ALTSEQ table is in bytes 1793 through 2048 (the last 256 bytes) of the parameter list.
Off: An ALTSEQ table is not in bytes 1793 through 2048 (the last 256 bytes) of the parameter list.

Bit 3:

On: The sort job was not completed successfully.
Off: The sort job was successful.

This bit must be off when calling the sort program.

Bits 4 through 7: Specifies the number of input file names (maximum of eight) in the parameter list. For example, 0010 in the last 4 bits indicates that two input file names are being passed to the sort program.

Output file name: Must be in bytes 2 through 9. The name must begin in byte 2; if the name is not 8 bytes long, it must be filled on the right with blanks.

Source member name: Must be in bytes 10 through 17 if the sort specifications are in a source member (bit 0 of the indicator byte is on). The name must begin in byte 10; if the name is not 8 bytes long, it must be filled on the right with blanks.

User library name: Must be in bytes 18 through 25 if the source member named in bytes 10 through 17 is in a user library (bit 1 of the indicator byte is on). The name must begin in byte 18; if the name is not 8 bytes long, it must be filled on the right with blanks.

The wrong sort specifications will be used by the sort program if the user library is not on disk and the source member in the system library is not the one desired.

Input file names: A maximum of eight input file names can be specified in the parameter list. The placement of the first file name in the parameter list depends on the status of bits 0 and 1 of the indicator byte.

If bits 0 and 1 of the indicator byte are both off, the first file name must be specified in bytes 10 through 17 of the parameter list.

If bit 0 is on and bit 1 of the indicator byte is off, the first input file name must be specified in bytes 18 through 25 of the parameter list

If bits 0 and 1 of the indicator byte are both on, the first input file name must be specified in bytes 26 through 33 of the parameter list.

Additional input file names must directly follow the 8 bytes of the previous input file name. An input file name must begin in the leftmost byte of the 8 bytes; if the file name is less than 8 bytes long it must be filled on the right with blanks.

Sort work area: Must immediately follow the 8 bytes of the last input file name. This 125-byte area is used as a work area when giving control to and returning control from the sort program.

Sort specifications: If bit 0 of the indicator byte is off, the sort specifications, corresponding to columns 6 through 39 on the sort specifications form, must immediately follow the 125-byte sort work area. The header specification must come first, followed by the record type and field specifications. Each sort specification must be 34 bytes long.

If an ALTSEQ table is specified (bit 2 of the indicator byte is on), do not specify the 34-byte sort specifications beyond byte 1792 of the parameter list. If an ALTSEQ table is not specified (bit 2 of the indicator byte is off), do not specify the 34-byte sort specifications beyond byte 2048 of the parameter list.

You must specify a // END statement immediately following the last 34-byte sort specification unless the last sort specification ends beyond byte 1758 (when an ALTSEQ table is specified) or beyond byte 2014 (when no ALTSEQ table is specified). In these cases, the sort program assumes a // END statement.

ALTSEQ table: Must be in bytes 1793 through 2048 (the last 256 bytes) of the parameter list if bit 2 of the indicator byte is on.

The table must contain 256 entries, each 1 byte long. The 1-byte entries correspond to the entries in the ENTRY column of the *Translation Table and Alternative Collating Sequence Coding Form.* When you want to alter the normal collating sequence in the table, enter an alternative collating sequence entry instead of the normal collating sequence entry. (See Appendix B for information on the collating sequence and an example of the *Translation Table and Alternative Collating Sequence Coding Form.*)

Note: The use of the sort interface increases the time needed to run a sort job. For improved performance when using the sort interface, increase the region size when you have enough main storage available to do so. (See the *System Reference* manual for information on how to increase the region size.)

Appendix D. Using Substitution Expressions

Sort allows you to use substitution expressions to replace information in statements that are generated *when you run a procedure that has the sort specifications inline in the procedure.* Substitution expressions can always be included in a procedure containing the sort specifications. The following example shows how to use substitution expressions:

Example of Using Substitution Expressions

Assume you have a BILLING procedure that is run twice a month:

- The procedure is run for the first time on the 15th day of the month, when customers whose names start with the characters A through M are billed.

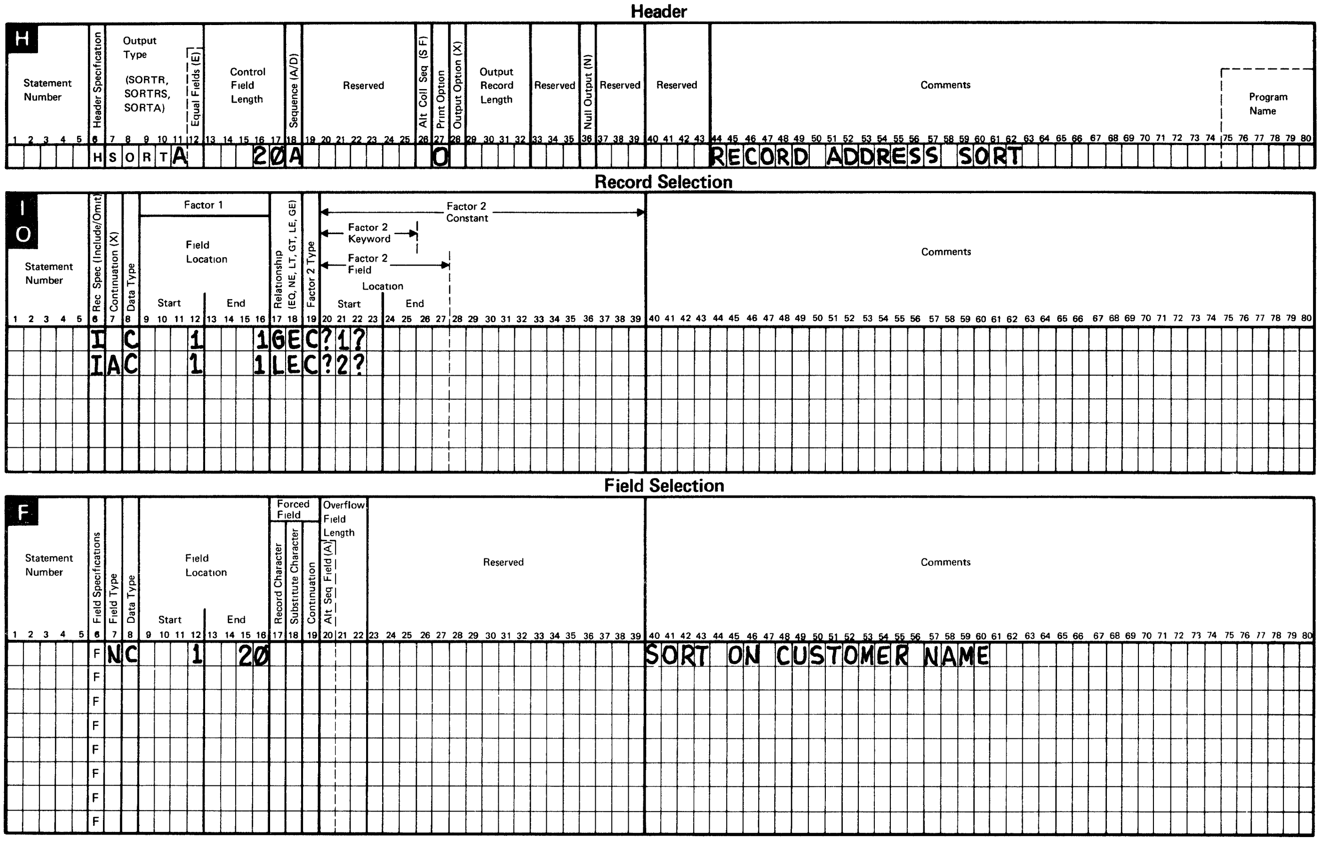
- The procedure is run for the second time on the last day of the month, when customers whose names start with the characters N through Z are billed.

All the records in a file named BILLING are sorted to group all customers to whom a bill will be sent. Assume the BILLING procedure has two parameters:

- The first initial of the first customers to be billed

- The first initial of the last customers to be billed.

You would define your sort specifications like this:



Header

Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
	H	SORTA		20	A			0							RECORD ADDRESS SORT	

Record Selection

Statement Number	Rec Spec (Include/Omit)	Continuation (X)	Data Type	Factor 1 Field Location Start	Factor 1 Field Location End	Relationship (EQ, NE, LT, GT, LE, GE)	Factor 2 Type	Factor 2 Constant / Keyword / Field Location	Comments
	I		C	1	1	GE	C	?1?	
	I	A	C	1	1	LE	C	?2?	

Field Selection

Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length Alt Seq Field (A)	Reserved	Comments
	F	N	C	1	20						SORT ON CUSTOMER NAME
	F										
	F										
	F										
	F										
	F										
	F										
	F										

After you define the sort specifications, you would enter your BILLING procedure that will run the sort job:

```
// LOAD #GSORT

// FILE NAME-INPUT,LABEL-BILLING

// FILE NAME-OUTPUT,LABEL-ADDROUT,RETAIN-T,RECORDS-500

// RUN

     HSORTA 20A      0        GENERATE RECORD ADDRESS SORT
     I C     1 1GEC?1?
     IAC     1 1LEC?2?
     FNC     1 20             SORT ON CUSTOMER NAME

// END
```

After you enter the sort procedure into the system, enter BILLING A,M to run the sort job for the 15th day of the month.

When the job is run:

1 The letter A is substituted for the expression: ?1?.

This specification states that the sort job should include all customers whose first initial is greater than or equal to parameter 1.

2 The expression ?2? is replaced by the letter M.

The sort specifications tell the sort program to include the names of all customers whose first initial is less than or equal to parameter 2.

When you use substitution expressions, comments should not be put on the same line as the substitution expression. If you put comments on the same line as the substitution expression, any data that follows the substitution may be shifted left or right depending on the number of characters replaced.

Some information that can be substituted includes:

- Positional parameters on the procedure command that called the procedure
- Specified characters in the display station local data area.

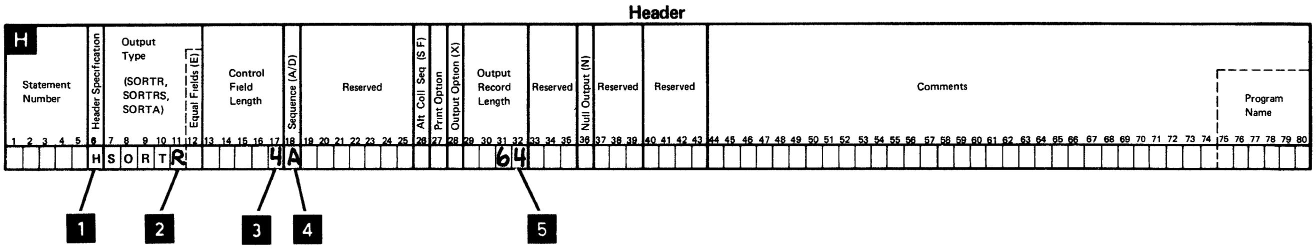
See the *System Reference* manual for detailed information about substitution expressions.

Appendix E. Sorting Integer (Binary) and Real Numbers

This appendix shows how either positive and negative integers or positive and negative real numbers in control fields can be used to sort records in a file.

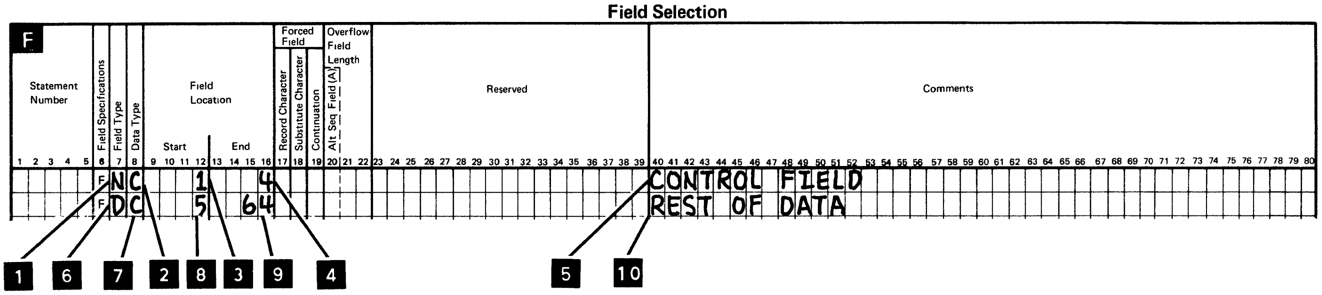
Defined sort specifications for example sort jobs are included, followed by a brief discussion of the entries made on the coding form.

Example 1. Sorting on a Control Field that Contains Either a Positive Integer or a Positive Real Number



Header

H Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
1 2 3 4 5	6	7 8 9 10 11	12	13 14 15 16 17	18	19 20 21 22 23 24 25	26	27	28	29 30 31 32	33 34 35	36	37 38 39	40 41 42 43	44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	75 76 77 78 79 80
	H	SORTR		4	A					64						



Field Selection

F Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Overflow Field Length Alt Seq Field (A)	Reserved	Comments
1 2 3 4 5	6	7	8	9 10 11 12	13 14 15 16	17	18	19	20 21 22	23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39	40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
	F	N	C	1	4						CONTROL FIELD
	F	D	C	5	64						REST OF DATA

Header Specification Entries

1 Identifies the statement as a header specification statement.

2 Identifies the job as a regular sort job.

3 The control field that contains either a positive integer or a positive real number has a length of 4 (the entry in column 17).

4 Specifies ascending order. The smallest positive integer or positive real number comes first; the largest positive integer or positive real number comes last in the sort sequence.

5 The control field is not dropped when data is written to the output file (column 28 is blank). Therefore, the length of the output record equals the length of the control field (4 bytes) added to the length of the data field (60 characters) specified by the field selection statements. This output record length totals 64, which is the entry for columns 29 through 32.

Record Selection Entries

Because all input records are being used and all have the same field specifications, no record selection statements are needed (include-all is implied).

Field Selection Entries

1 Indicates a normal control field.

2 Indicates that the control field contains character data where both the zone and digit portions of characters are to be compared.

3, 4 Identify the starting and ending positions of the control field within the input records. This control field is included in the sorted output file (column 28 of the header statement is blank).

5 Is a comment.

6 Indicates a normal data field.

7 Identifies the data as character data.

8, 9 Identify the fields to be written into the output file.

10 Is a comment.

Example 2. Sorting on a Control Field Containing a Positive or Negative Integer

Header

H Statement Number	Header Specification	Output Type (SORTR, SORTRS, SORTA)	Equal Fields (E)	Control Field Length	Sequence (A/D)	Reserved	Alt Coll Seq (S F)	Print Option	Output Option (X)	Output Record Length	Reserved	Null Output (N)	Reserved	Reserved	Comments	Program Name
1 2 3 4 5	6	7 8 9 10 11	12	13 14 15 16 17	18	19 20 21 22 23 24 25	26	27	28	29 30 31 32	33 34 35	36	37 38 39	40 41 42 43	44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	75 76 77 78 79 80
	H	SORTR		5	A				X	64						

Record Selection

I O Statement Number	Rec Spec (Include/Omit)	Continuation (X)	Data Type	Factor 1 Field Location Start	Factor 1 Field Location End	Relationship (EQ, NE, LT, GT, LE, GE)	Factor 2 Type	Factor 2 Constant / Factor 2 Keyword / Factor 2 Field Location Start	End		Comments
1 2 3 4 5	6	7	8	9 10 11 12	13 14 15 16	17 18	19	20 21 22 23	24 25 26 27	28 29 30 31 32 33 34 35 36 37 38 39	40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
	I		C		1	LE	C	″			ALL POSITIVE REAL NUMBERS

Field Selection

F Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Alt Seq Field (A)	Overflow Field Length	Reserved	Comments
1 2 3 4 5	6	7	8	9 10 11 12	13 14 15 16	17	18	19	20	21 22	23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39	40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
	F	F	C			2						FORCE 2 FOR POS. REAL NUMBERS
	F	N	C	1	4							CONTROL FIELD
	F	D	C	1	64							DATA FIELD
	F											
	F											
	F											
	F											
	F											

Record Selection

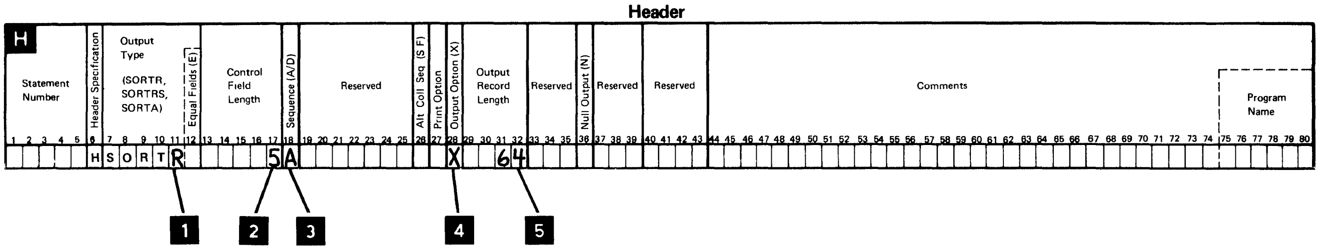
I O Statement Number	Rec Spec (Include/Omit)	Continuation (X)	Data Type	Factor 1 Field Location Start	Factor 1 Field Location End	Relationship (EQ, NE, LT, GT, LE, GE)	Factor 2 Type	Factor 2 Constant / Factor 2 Keyword / Factor 2 Field Location Start	End		Comments
1 2 3 4 5	6	7	8	9 10 11 12	13 14 15 16	17 18	19	20 21 22 23	24 25 26 27	28 29 30 31 32 33 34 35 36 37 38 39	40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
	I										INCLUDE NEGATIVE INTEGERS IN SORT

Field Selection

F Statement Number	Field Specifications	Field Type	Data Type	Field Location Start	Field Location End	Forced Field Record Character	Forced Field Substitute Character	Continuation	Alt Seq Field (A)	Overflow Field Length	Reserved	Comments
1 2 3 4 5	6	7	8	9 10 11 12	13 14 15 16	17	18	19	20	21 22	23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39	40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
	F	F	C			1						FORCE 1 FOR NEGATIVE INTEGERS
	F	N	C	1	4							CONTROL FIELD
	F	D	C	1	64							DATA FIELD

Note: Real numbers can also be sorted as shown in this example.

Header Specification Entries



Columns	Entry
6	H
7–12	SORTR
13–17	5
18	A
28	X
29–32	64

1 Identifies the job as a regular sort job.

2 Specifies the total length of the control field (for either a positive or a negative integer) as 5 bytes.

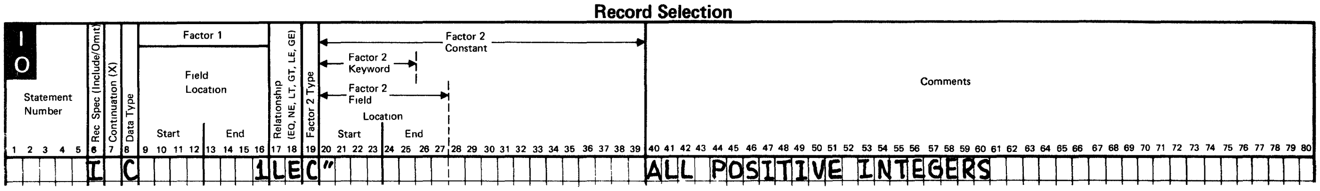
3 Indicates ascending order. Since negative integers are stored in two′s complement form, the smallest negative integer comes first; the largest positive integer comes last in the sort sequence (for example: -2, -1, 0, 1, 2).

4 Indicates that the control field is dropped when data is written into the output file.

5 When the control field is dropped, only the data portion remains. The data portion in this example is the entire input record. Therefore, all 64 characters in the input records will be written into the output records.

Record Selection Entries

Two record types are specified (positive integers and negative integers). Both record types are to be included in the sort. Therefore, two sets of include specifications (with any corresponding field selection specifications) are required.



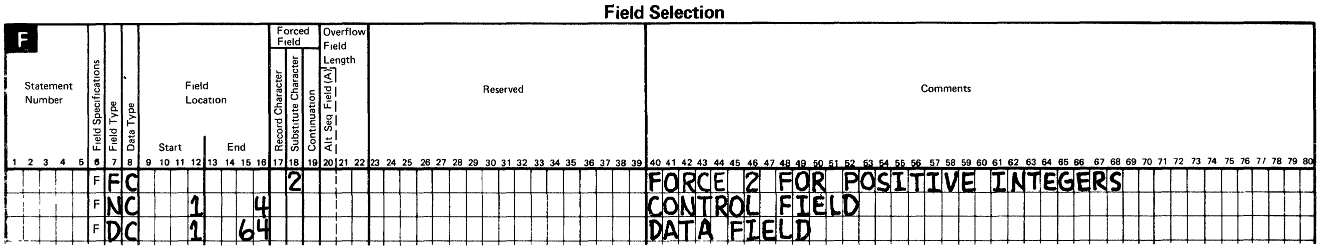
Record Selection

I O Statement Number (1-5)	Rec Spec (Include/Omit) (6)	Continuation (X) (7)	Data Type (8)	Factor 1 Field Location Start (9-12)	Factor 1 Field Location End (13-16)	Relationship (EQ, NE, LT, GT, LE, GE) (17-18)	Factor 2 Type (19)	Factor 2 Constant / Keyword / Field Location (20-39)	Comments (40-80)
	I		C		1	LE	C	″	ALL POSITIVE INTEGERS

This conditional include statement specifies that if position 1 (the leftmost byte of the 4-byte control field from the input record) is less than or equal to the character constant ″ (see note), the record will be included in the sort. The character ″ indicates that the record contains a positive integer.

Note: The special character ″ has a bit configuration of 01111111, which is equal to the high order byte of the maximum positive integer.

Field Selection Entries



Field Selection

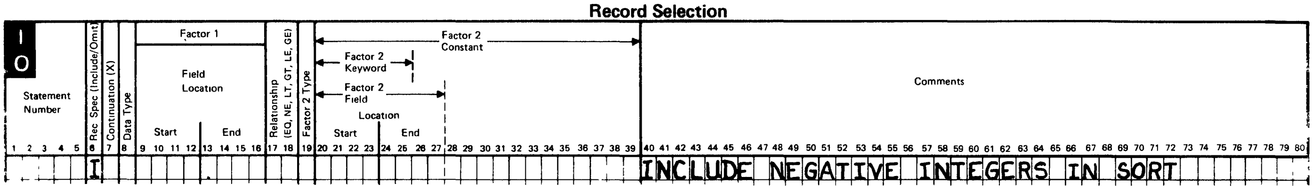
F Statement Number (1-5)	Field Specifications (6)	Field Type (7)	Data Type (8)	Field Location Start (9-12)	Field Location End (13-16)	Forced Field Record Character (17)	Forced Field Substitute Character (18)	Continuation (19)	Alt Seq Field (A) (20)	Overflow Field Length (21-22)	Reserved (23-39)	Comments (40-80)
	F	F	C				2					FORCE 2 FOR POSITIVE INTEGERS
	F	N	C	1	4							CONTROL FIELD
	F	D	C	1	64							DATA FIELD

The preceding field selection statements specify that if the input records meet the conditions defined in the include statement (contents of position 1 equals ″), the sort program should force a 2 into the first position of its control field.

In this example, the control field is in positions 1 through 4.

The last statement specifies that all the data in positions 1 through 64 be written into the output file.

Record Selection Entries

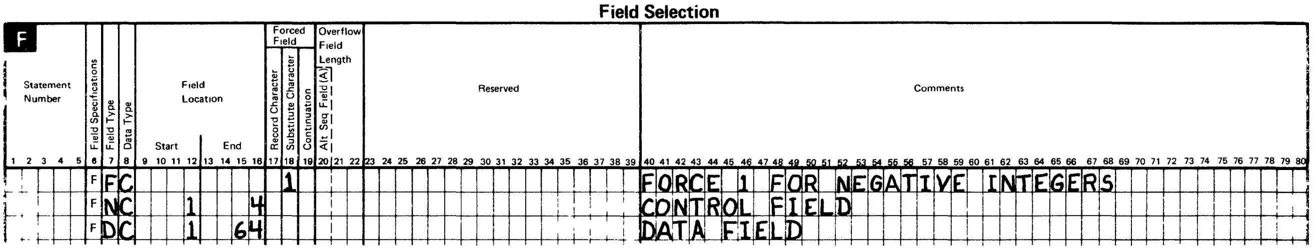


Record Selection

```
     I                                 INCLUDE NEGATIVE INTEGERS IN SORT
```

This include-all statement specifies that all records not previously selected (or omitted) be included in the sort. Because the first include-all statement selected records with positive integers only, this include statement will select only records containing negative integers (or all remaining records).

Field Selection Entries



Field Selection

```
     FFC        1                      FORCE 1 FOR NEGATIVE INTEGERS
     FNC   1   4                       CONTROL FIELD
     FDC   1  64                       DATA FIELD
```

The first field selection statement specifies that the character 1 will be forced into the first position of the work record for each record with a negative integer. All records with negative integers will be forced in front of all records with positive integers when the records are sorted.

The next statement defines the control field as a four character field in positions 1 through 4 of the input records; and the last field specification statement defines data fields that will be written into the output file.

Example 3. Record Selection Based on Binary Values

To select records based on a binary value, you can use one or both of the following include methods.

For Characters in the 64-Character Set

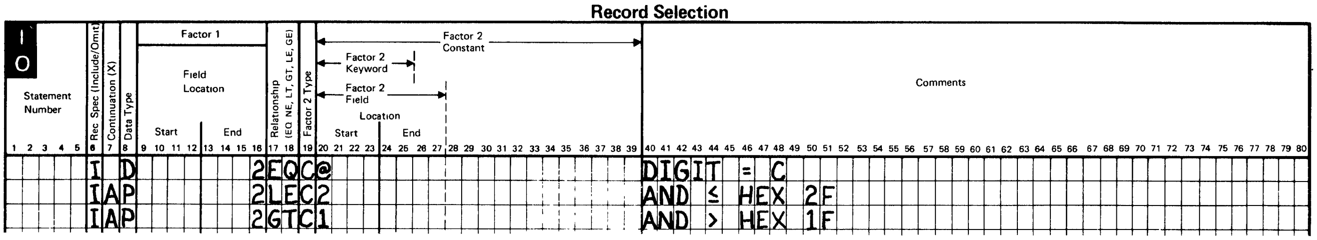
For binary values whose hexadecimal equivalents can be represented by one or two characters in the 64-character set (Appendix B, *Collating Sequences*), use either an *include character* (I and C in columns 6 and 8 of the record selection specifications) or an *include zone* (I and Z in columns 6 and 8 of the record selection specifications) ANDed with an *include digit* (IAD in columns 6 through 8 of the record selection specifications).

For Characters Not in the 64-Character Set

For binary values whose hexadecimal zone portions do not appear in the 64-character set, use two include specifications for the zone portion ANDed with an *include digit* for the digit portion.

For example, assume you want to select records that contain a value of 44 in position 2.

The binary representation of this number is 0010 1100, which is a hex 2C. For the digit portion, select a character from the table in Appendix B with a digit portion of C (for example, <, *, %, and @). Use this character in an include statement, comparing its digit portion to the binary value of 1100. Because there is no character in the 64-character set whose zone portion equals 2, use two include statements to describe constants with a zone portion greater than 1 and less than or equal to 2. Describe these constants as packed, which reverses the zone and digit portions from hexadecimal F1 and F2 to hexadecimal 1F and 2F as shown in the following example.



Record Selection

```
     I D       2EQC@                   DIGIT = C
     IAP       2LEC2                   AND ≤ HEX 2F
     IAP       2GTC1                   AND > HEX 1F
```

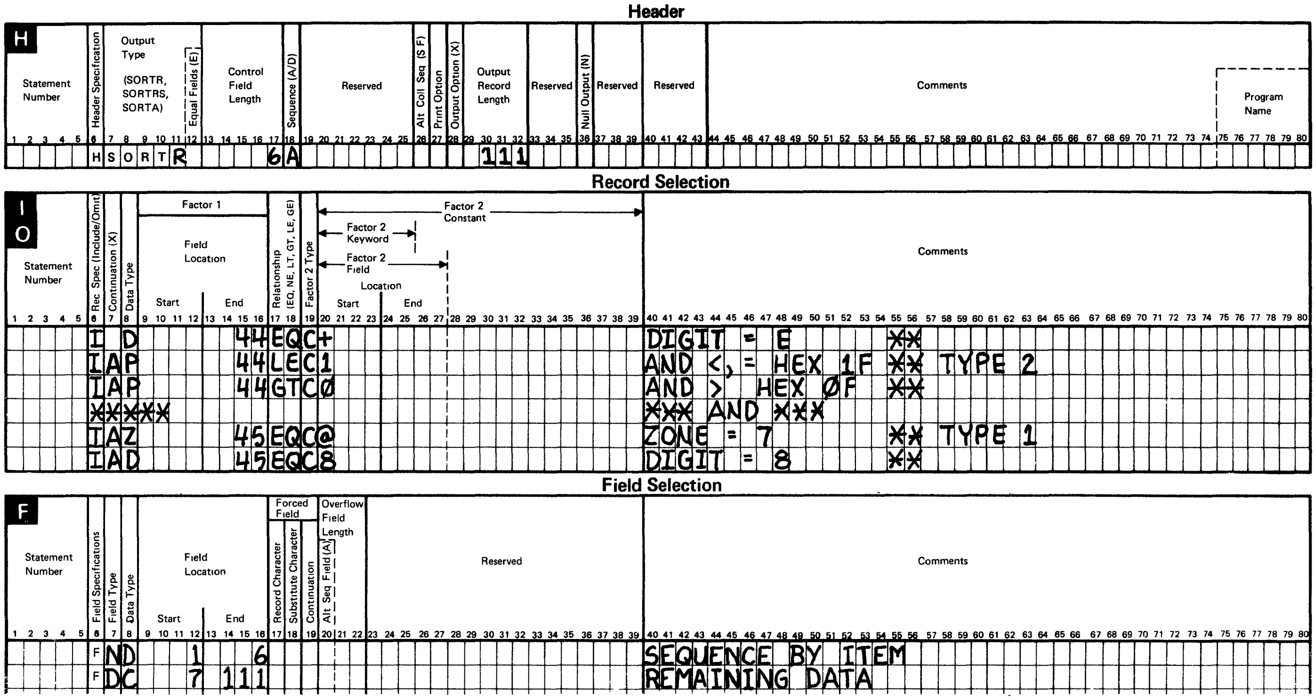
I
O
Statement Number
Rec Spec (Include/Omit)
Continuation (X)
Data Type
Factor 1
Field Location
Start
End
Relationship (EQ, NE, LT, GT, LE, GE)
Factor 2 Type
Factor 2 Constant
Factor 2 Keyword
Factor 2 Field Location
Start
End
Comments
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

The record selection specifications shown will cause selection of all records containing (in column 2) a value between 31 (hexadecimal 1F) and 47 (hexadecimal 2F), and whose digit portion is equal to C.

	Hexadecimal Value	**Binary Value**	**Decimal Value**
Field to be selected	2C	= 0010 1100 =	44
Character @	7C	= 0111 1100 =	124
Packed character 2	2F	= 0010 1111 =	47
Packed character 1	1F	= 0001 1111 =	31

Note: The 16 hexadecimal values whose zone portions are B cannot be accessed with these types of entries. Therefore, there can be no record selection based on these values.

This example selects records from an inventory file whose reorder point is 7800. The reorder point is a 2-byte quantity in columns 44 and 45 of the inventory records. The quantity is stored in binary (7800 = 0001 1110 0111 1000), which is hexadecimal 1E78.



Header

```
     HSORTR     6A           111
```

Record Selection

```
     I D      44EQC+                   DIGIT = E      **
     IAP      44LEC1                   AND <,= HEX 1F ** TYPE 2
     IAP      44GTCØ                   AND >   HEX ØF **
     *****                             *** AND ***
     IAZ      45EQC@                   ZONE = 7       ** TYPE 1
     IAD      45EQC8                   DIGIT = 8      **
```

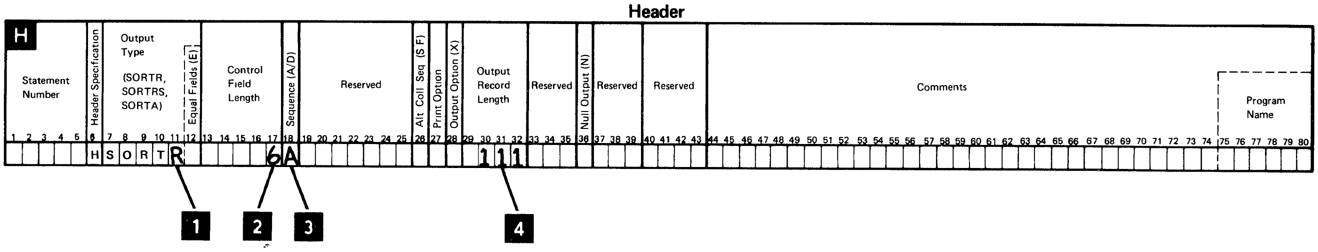
Field Selection

```
     FND   1   6                       SEQUENCE BY ITEM
     FDC   7 111                       REMAINING DATA
```

These specifications will cause selection of all records containing (in column 44) a value between 15 (hexadecimal 0F) and 31 (hexadecimal 1F), and whose digit portion equals E. They will further limit selection of records to those with a binary value in column 45 whose zone portion equals 7, and whose digit portion equals 8. Therefore, all records selected will have hexadecimal 1E78 in columns 44 and 45.

	Hexadecimal Value	**Binary Value**	**Decimal Value**
Field to be selected, column 44	1E	= 0001 1110 =	30
Character	4E	= 0100 1110 =	78
Packed character 1	1F	= 0001 1111 =	31
Packed character 0	0F	= 0000 1111 =	15
Field to be selected, column 45	78	= 0111 1000 =	120
Character @	7C	= 0111 1100 =	124
Character 8	F8	= 1111 1000 =	248

Header Specification Entries



1 Identifies the job as a regular sort job.

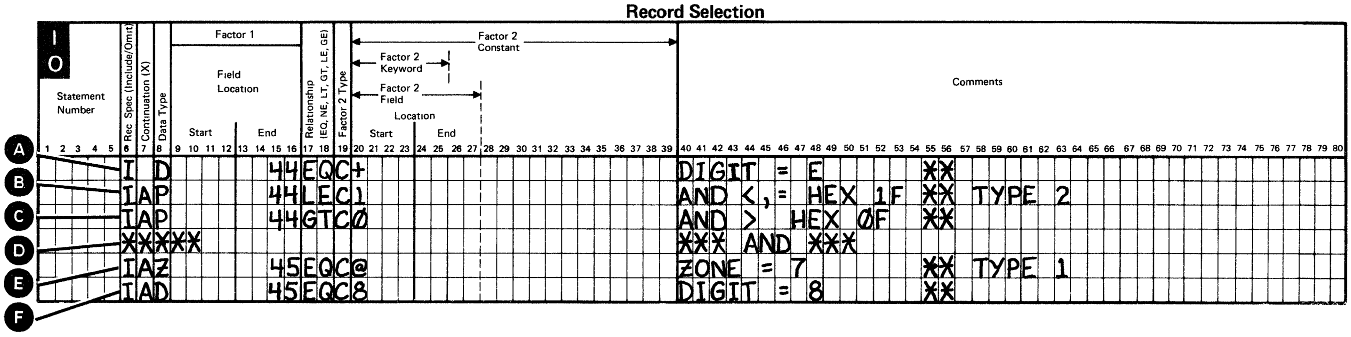
2 Indicates the total length of the control field as 6 bytes.

3 Indicates ascending order. The file will be sorted according to the standard collating sequence shown in Appendix B.

4 Indicates that the output file will contain all the data in the input file.

The control field is not dropped when data is written to the output file (column 28 is blank). Therefore, the length of the output record equals the length of the control field (6 bytes) plus the length of the data field (105 characters) specified by statements 07 and 08. This output record length totals 111.

Record Selection Entries

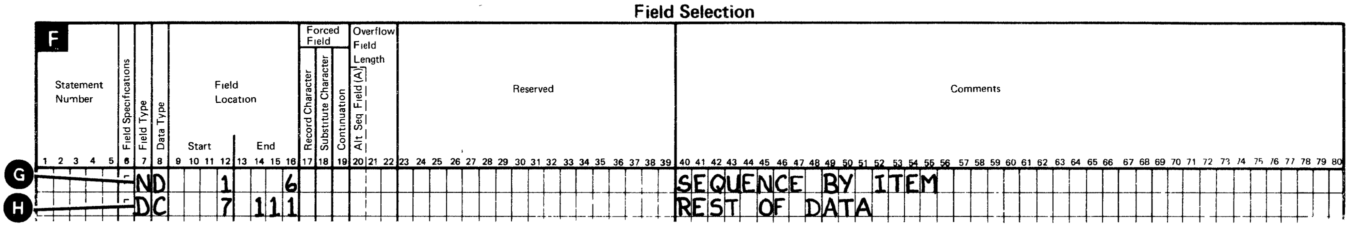

Record Selection
I O
Statement Number
Rec Spec (Include/Omit)
Continuation (X)
Data Type
Factor 1
Field Location
Start
End
Relationship (EQ, NE, LT, GT, LE, GE)
Factor 2 Type
Factor 2 Constant
Factor 2 Keyword
Factor 2 Field
Location
Start
End
Comments
A
B
C
D
E
F

```
     I D      44EQC+                   DIGIT = E      **
     IAP      44LEC1                   AND <,= HEX 1F ** TYPE 2
     IAP      44GTCØ                   AND >  HEX ØF  **
     *****                             *** AND ***
     IAZ      45EQC@                   ZONE = 7       ** TYPE 1
     IAD      45EQC8                   DIGIT = 8      **
```

Statement A selects records whose digit portion in column 44 is E. Statements B and C select records whose packed value in column 44 is less than or equal to hexadecimal 1F and, at the same time, is greater than a packed value of hexadecimal 0F.

The only hexadecimal values that satisfy both specifications for statement B and C are hexadecimal 11 through hexadecimal 1F. Statement D is a comment statement. Statements E and F select records whose zone portion in column 45 is 7 and whose digit portion is 8.

Field Selection Entries


Field Selection
F
Statement Number
Field Specifications
Field Type
Data Type
Field Location
Start
End
Forced Field
Record Character
Substitute Character
Continuation
Overflow Field Length
Alt Seq Field (A)
Reserved
Comments
G
H

```
      ND   1   6                       SEQUENCE BY ITEM
      DC   7 111                       REST OF DATA
```

Statement G indicates that the control field is in positions 1 through 6 of the input records. Only the digit portions of the characters will be used. This control field will be included in the output file.

Statement H indicates that the other 105 input record characters are included as data in the output file.

Glossary

#LIBRARY. The library, provided with the system, that contains the System Support Program Product. See *system library.*

absolute value. The numeric value of a real number regardless of its sign (positive or negative).

accumulate. To collect. For example, to accumulate the values in a field.

accumulating. The process of totaling the values in a particular field as records are being processed.

address. A name, label, or number that identifies a location in storage, a device in a network, or any other data source.

addrout file. See *address output file.*

allocate. To assign a resource, such as a disk file or a diskette file, to perform a specific task.

alphabetic character. Any one of the letters A through Z (uppercase and lowercase). Some program products extend the alphabet to include the special characters #, $, and @.

alphameric. Consisting of letters, numbers, and often other symbols, such as punctuation marks and mathematical symbols.

alphanumeric. See *alphameric.*

alternative collating sequence. A user-defined collating sequence that replaces the standard EBCDIC collating sequence.

ascending key sequence. The arrangement of data in order from the lowest value of the key field to the highest value of the key field. Contrast with *descending key sequence.*

assembler. A program that converts assembler language statements to machine instructions.

assembler instruction statement. A statement that controls what the assembler does, rather than what the user program does.

assembler language. A symbolic programming language in which the set of instructions includes the instructions of the machine and whose data structures correspond directly to the storage and registers of the machine.

backup copy. A copy, usually of a file or of a library member, that is kept in case the original file or library member is unintentionally changed or destroyed.

binary. (1) Pertaining to a system of numbers to the base two; the binary digits are 0 and 1. (2) Involving a choice of two conditions, such as on-off or yes-no.

bit. Either of the binary digits 0 or 1. See also *byte.*

block. (1) A group of records that is recorded or processed as a unit. Same as *physical record.* (2) Ten sectors (2560 bytes) of disk storage.

buffer. An area of storage, temporarily reserved for performing input or output, into which data is read or from which data is written.

byte. The amount of storage required to represent one character; a byte is 8 bits.

call. To activate a program or procedure at its entry point. Compare with *load.*

cancel. To end a task before it is completed.

character. A letter, digit, or other symbol.

close. To end the processing of a file.

COBOL (common business-oriented language). A high-level programming language, similar to English, that is used primarily for commercial data processing.

code. (1) Instructions for the computer. (2) To write instructions for the computer. Same as *program.* (3) A representation of a condition, such as an error code.

collating sequence. The sequence in which characters are ordered within the computer for sorting, combining, or comparing.

comment. Words or statements in a program or procedure that serve as documentation rather than as instructions.

compile. To translate a program written in a high-level programming language into a machine language program.

concatenate. (1) To link together. (2) To join two character strings.

conditional force. In sort, the replacement of control field characters before the records are sorted if the control field in the input record contains a particular entry.

consecutive processing. The processing of records in the order in which they exist in a file. Same as *sequential processing*. See also *random processing*.

constant. A data item with a value that does not change. Contrast with *variable*.

constant field. A field that is defined by a display format to contain a value that does not change.

continuation line. A line of a source statement into which characters are entered when the source statement cannot be contained on the previous line or lines.

control field. A field that identifies a record's relationship to other records (such as a part number in an inventory record). In RPG, control fields are compared from record to record to determine when certain operations are to be performed. In sort, control fields determine the order of records in the sorted file.

creation date. The program date at the time a file is created. See also *program date*, *session date*, and *system date*.

current library. The first library searched for any required members. The current library can be specified during sign-on or while running programs and procedures.

DDM. See Distributed Data Management.

decimal. Pertaining to a system of numbers to the base ten; decimal digits range from 0 through 9.

default value. A value stored in the system that is used when no other value is specified.

delete. To remove. For example, to delete a file.

delete character. A character that identifies a record to be removed from a file.

delete-capable file. A file from which records can be logically removed without compressing the file.

descending key sequence. The arrangement of data in order from the highest value of the key field to the lowest value of the key field. Contrast with *ascending key sequence*.

diagnostic. Pertaining to the detection and isolation of an error.

direct file. A disk file in which records are referenced by the relative record number. Contrast with *indexed file* and *sequential file*.

disk. A storage device made of one or more flat, circular plates with magnetic surfaces on which information can be stored.

disk drive. The mechanism used to read and write information on disk.

disk file. A set of related records on disk that are treated as a unit.

diskette. A thin, flexible magnetic plate that is permanently sealed in a protective cover. It can be used to store information copied from the disk.

display. (1) A visual presentation of information on a display screen. (2) To show information on the display screen.

display screen. The part of the display station on which information is displayed.

display station. A device that includes a keyboard from which an operator can send information to the system and a display screen on which an operator can see the information sent or receive information from the system.

Distributed Data Management. Part of the SSP which lets you use files on remote systems. See the *Distributed Data Management Guide*, SC21-8011, for more information.

dump. (1) To copy the contents of all or part of storage, usually to an output device. (2) Data that has been dumped.

EBCDIC. See *extended binary-coded decimal interchange code*.

EBCDIC character. Any one of the symbols included in the 8-bit EBCDIC set.

enter. To type in information on a keyboard and press the Enter key in order to send the information to the computer.

extendable disk file. A file that the system can increase in size whenever more space is needed.

extended binary-coded decimal interchange code (EBCDIC). A set of 256 eight-bit characters.

field. One or more characters of related information (such as a name or an amount).

file. A set of related records treated as a unit.

file name. The name used by a program to identify a file. See also *label.*

force-all. In sort, a specification that tests whether the control field in the input record contains a particular entry. If it does not, the control field character is replaced before the record is sorted.

forced control field. In sort, a one-position control field that results from replacing one character with another character, or from forcing a character into a control field position.

format. (1) A defined arrangement of such things as characters, fields, and lines, usually used for displays, printouts, or files. (2) To arrange such things as characters, fields, and lines.

generation. For some remote systems, the translation of configuration information into machine language.

hexadecimal. Pertaining to a system of numbers to the base sixteen; hexadecimal digits range from 0 (zero) through 9 (nine) and A (ten) through F (fifteen).

include set. In sort, sequence specifications that identify one or more record types to be sorted.

index. (1) A table containing the key value and location of each record in an indexed file. (2) A computer storage position or register, the contents of which identify a particular element in a set of elements.

indexed file. A file in which the key and the position of each record is recorded in a separate portion of the file called an index. Contrast with *direct file* and *sequential file.*

indicator. An internal switch that communicates a condition between parts of a program or procedure.

informational message. A message that provides information to the operator, but does not require a response.

initialize. To prepare for use. For example, to initialize a diskette.

input. Data to be processed.

instruction. A statement that specifies an operation to be performed by the computer and the locations in storage of all data involved in that operation.

integer. A positive or negative whole number; that is, an optional sign followed by a number that does not contain a decimal point.

job. (1) A unit of work to be done by a system. (2) One or more related procedures or programs grouped into a procedure.

job file. A disk file that exists until the job that uses it ends.

job step. A unit of work represented by a single program or a procedure that contains a single program. A job consists of one or more job steps.

K-byte. 1024 bytes.

label. (1) The name in the disk or diskette volume table of contents that identifies a file. See also *file name.* (2) The name that identifies a statement.

left-adjust. To place or move an entry in a field so that the leftmost character of the field is in the leftmost position. Contrast with *right-adjust.*

library. (1) A named area on disk that can contain programs and related information (not files). A library consists of different sections, called library members. (2) The set of publications for a system.

library member. A named collection of records or statements in a library. The types of library members are *load member*, *procedure member*, *source member*, and *subroutine member.*

literal. A symbol or a quantity in a source program that is itself data, rather than a reference to data.

load. To move data or programs into storage.

local data area. A 512-byte area on disk that can be used to pass information between jobs and job steps during a session. A separate local data area exists for each command display station.

macroinstruction. A single instruction that represents a set of instructions.

main storage. The part of the processing unit where programs are run. Contrast with *control storage.*

member. See *library member.*

menu. A displayed list of items from which an operator can make a selection.

message. Information sent to an operator or programmer from a program. A message can be either displayed or printed.

message identification. A field in the display or printout of a message that directs the user to the description of the message in a message guide or a reference manual. This field consists of up to four alphabetic characters, followed by a dash, followed by the message identification code.

message identification code (MIC). A four-digit number that identifies a record in a message member. This number can be part of the message identification.

multiple. More than one.

multiprogramming. The processing of two or more programs at the same time.

normal control field. In sort, a control field that is sorted in the sequence specified in the header specification.

null character. The character hex 00, used to represent the absence of a printed or displayed character.

numeric. Pertaining to any of the digits 0 through 9.

object program. In COBOL, a set of instructions in machine-runnable form. The object program is produced by a compiler from a source program.

OCL. See *operation control language.*

open. To prepare a file for processing.

operation. A defined action, such as adding or comparing, performed on one or more data items.

operation control language (OCL). A language used to identify a job and its processing requirements to the System Support Program Product.

opposite control field. In sort, a control field that is sorted in the opposite sequence of that specified in the header specification.

output. The result of processing data.

overflow field. In sort, a field that allows for field expansion.

overlay. (1) To write over (and therefore destroy) an existing file. (2) A program segment that is loaded into main storage and replaces all or part of a previously loaded program segment.

packed decimal format. A format in which each byte (except the rightmost byte) within a field represents two numeric digits. The rightmost byte contains one digit and the sign. For example, the decimal value +123 is represented as 0001 0010 0011 1111. Contrast with *zoned decimal format.*

parameter. A value supplied to a procedure or program that either is used as input or controls the actions of the procedure or program.

physical record. A unit of data that is moved into or out of the computer.

position. The location of a character in a series, as in a record, a displayed message, or a computer printout.

positional parameter. A parameter that must appear in a specified location, relative to other positional parameters.

printout. Information from the computer that is produced by a printer.

procedure. A set of related operation control language statements (and, possibly, utility control statements and procedure control expressions) that cause a specific program or set of programs to be performed.

procedure command. A command that runs a procedure.

procedure level. The relative position of a procedure within nested procedures. For example, if procedure A calls procedure B, and procedure B in turn calls procedure C, then procedure C is a third-level procedure.

procedure member. A library member that contains the statements (such as operation control language statements) necessary to perform a program or set of programs.

program. (1) A sequence of instructions for a computer. See *source program* and *load module.* (2) To write a sequence of instructions for a computer. Same as *code.*

program date. The date associated with a program (job step). See also *creation date, session date,* and *system date.*

program product. A licensed program for which a fee is charged.

prompt. A displayed request for information or operator action.

queue. A line or list formed by items waiting to be processed.

real number. A number, containing a decimal point, stored in fixed-point or floating-point format.

record. A collection of fields that is treated as a unit.

record address file. An input file that indicates to a program which records are to be read from a disk file, and the order in which these records are to be read from the disk file.

record type. The classification of records in a file.

recovery procedure. (1) An action performed by the operator when an error message appears on the display screen. Usually, this action permits the program to continue or permits the operator to run the next job. (2) The method of returning the system to the point where a major system error occurred and running the recent critical jobs again.

region. The amount of main storage available for a program. See also *job region* and *step region.*

relational expression. A logical statement that describes the relationship (such as greater than or equal) of two arithmetic expressions or data items.

relational operator. The reserved words or symbols used to express a relation condition or a relational expression.

relative record number. A number that specifies the location of a record in relation to the beginning of the file.

remote file. A file that resides on a separate system. This system may be communicating with your system via telecommunications. See the *Distributed Data Management Guide*, SC21-8011, for more information.

reset. To return a device or circuit to a clear state.

resident file. A file that exists on disk until it is specifically deleted or changed to a scratch file.

right-adjust. To place or move an entry in a field so that the rightmost character of the field is in the rightmost position. Contrast with *left-adjust.*

scratch file. A file, usually used as a work file, that exists until the program that uses it ends.

sector. (1) An area on a disk track or a diskette track reserved to record information. (2) The smallest amount of information that can be written to or read from a disk or diskette during a single read or write operation.

selection field. A field tested for a condition to determine whether a record should be included in a sort.

sequential file. A file in which records occur in the order in which they were entered. Contrast with *direct file* and *indexed file.*

SEU. See *source entry utility.*

significant digit. Any digit of a number that follows the leftmost digit which is not a zero and that is within the accuracy allowed.

sort sequence specifications. Source statements that specify the sequence of a sort.

source. A system, a program within a system, or a device that makes a request to a target. Contrast with *target.*

source entry utility (SEU). The part of the Utilities Program Product used by the operator to enter and update source and procedure members.

source member. A library member that contains information in the form in which it was entered, such as RPG specifications. Contrast with *load member.*

source program. A set of instructions that are written in a programming language and that must be translated to machine language before the program can be run.

source statement. A statement written in a programming language.

special character. A character other than an alphabetic or numeric character. For example; *, +, and % are special characters.

specification sheets. Forms on which a program is coded and described.

SSP. See *System Support Program Product.*

statement. An instruction in a program or procedure.

summary data field. In sort, a data field designated for accumulated totals.

system. The computer and its associated devices and programs.

system date. The date assigned by the system operator during the initial program load procedure. See also *creation date*, *program date*, and *session date*.

system library. The library, provided with the system, that contains the System Support Program Product and is named #LIBRARY.

System Support Program Product (SSP). A group of licensed programs that manage the running of other programs and the operation of associated devices, such as the display station and printer. The SSP also contains utility programs that perform common tasks, such as copying information from diskette to disk.

task. A unit of work (such as a user program) for the main storage processor.

transient. Pertaining to a System Support Program Product program that does not reside in main storage or to a temporary storage area for such a program.

unconditional force. In sort, a specification that always results in a character being forced into the control field before the records are sorted.

Utilities Program Product. A program product that contains the data file utility (DFU), the source entry utility (SEU), the work station utility (WSU), and the screen design aid (SDA).

utility control statement. A statement that gives a utility program information about the way the program is to perform or the output it is to produce.

utility program. A System Support Program Product program that allows you to perform a common task, such as copying information from diskette to disk.

variable. A name used to represent a data item whose value can change while the program is running. Contrast with *constant*.

work file. A file that is used for temporary storage of data being processed.

work record. A record built by the sort program for later processing.

work station. A device that lets people transmit information to or receive information from a computer; for example, a display station or printer.

zoned decimal format. A format for representing numbers in which the digit is contained in bits 4 through 7 and the sign is contained in bits 0 through 3 of the rightmost byte; bits 0 through 3 of all other bytes contain 1s (hex F). For example, in zoned decimal format, the decimal value of +123 is represented as 1111 0001 1111 0010 1111 0011. Contrast with *packed decimal format*.

zoned decimal item. A numeric data item that is represented internally in zoned decimal format.

zoned field. A field that contains data in the zoned decimal format.

Index

IBM System/36:
Sort Guide

SC21-7903-2

READER'S COMMENT FORM

Please use this form only to identify publication errors or to request changes in publications. Direct any requests for additional publications, technical questions about IBM systems, changes in IBM programming support, and so on, to your IBM representative or to your nearest IBM branch office. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

□ If your comment does not need a reply (for example, pointing out a typing error) check this box and do not include your name and address below. If your comment is applicable, we will include it in the next revision of the manual.

□ If you would like a reply, check this box. Be sure to print your name and address below.

Page number(s): Comment(s):

Please contact your nearest IBM branch office to request additional publications.

Name ______________________________

Company or
Organization ______________________________

Address ______________________________

____
City State Zip Code

No postage necessary if mailed in the U.S.A.

SC21-7903-2

Cut Along Line

Fold and tape Please do not staple Fold and tape

NO POSTAGE
NECESSARY IF
MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N. Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

IBM CORPORATION
Information Development
Department 245
Rochester, Minnesota, U.S.A. 55901

Fold and tape Please do not staple Fold and tape

IBM

IBM System/36:
Sort Guide

SC21-7903-2

What Is Your Opinion of This Manual?

Your comments can help us produce better manuals. Please take a few minutes to evaluate this manual as soon as you become familiar with it. Circle Y (Yes) or N (No) for each question that applies. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

FINDING INFORMATION

Y N Is the table of contents helpful?
What would make it more helpful?

Y N Is the index complete?
List specific terms that are missing.

Y N Are the chapter titles and other headings meaningful?
What would make them more meaningful?

Y N Is information organized appropriately?
What would improve the organization?

Y N Does the manual refer you to the appropriate places for more information?
List specific references that are wrong or missing.

UNDERSTANDING INFORMATION

Y N Is the purpose of this manual clear?
What would make it clearer?

Y N Is the information explained clearly?
Which topics are unclear?

Y N Are the examples clear?
Which examples are unclear?

Y N Are examples provided where they are needed?
Where should examples be added or deleted?

Y N Are terms defined clearly?
Which terms are unclear?

Y N Are terms used consistently?
Which terms are inconsistent?

Y N Are too many abbreviations and acronyms used?
Which ones are not understandable?

Y N Are the illustrations clear?
Which ones are unclear?

USING INFORMATION

Y N Does the information apply to your situation?
Which topics do not apply?

Y N Is the information accurate?
What information is inaccurate?

Y N Is the information complete?
What information is missing?

Y N Is only necessary information included?
What information is unnecessary?

Y N Are the examples useful models?
What would make them more useful?

Y N Is the format of the manual (shape, size, color) effective?
What would make the format more effective?

OTHER COMMENTS

Use the space below for any other opinions about this manual or about the entire set of manuals for this system.

YOUR BACKGROUND

What is your job title?

What is your primary job responsibility?

How many years have you used computers?

Which programming languages do you use?

How many times per month do you use this manual?

Your name
Company name
Street address
City, State, ZIP

No postage necessary if mailed in the U.S.A.

SC21-7903-2

Cut Along Line

Fold and tape | Please do not staple | Fold and tape

NO POSTAGE
NECESSARY IF
MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N. Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

IBM CORPORATION
Information Development
Department 532
Rochester, Minnesota, U.S.A. 55901

Fold and tape | Please do not staple | Fold and tape

IBM

IBM

International Business Machines Corporation

Sort Guide

Contents

File Number
S36-33

Order Number
SC21-7903-2

Printed in U.S.A.

SC21-7903-02