Appendix
A

SORT

SYS-ED/
COMPUTER
EDUCATION
TECHNIQUES, INC.

1 Sorting Data

There are a number of utility programs available for sorting data.

The two most common are ICEMAN and SYNCSORT. Many installations change the name of the program to SORT.

Sort JCL

Both sort programs use the JCL shown below:

```
//SORT1     EXEC     PGM=ICEMAN
//SYSOUT    DD       SYSOUT=*
//SORTIN    DD       DSN=SYSED.UNSORTED.DATA,
//                   DISP=SHR
//SORTOUT   DD       DSN=SYSED.SORTED.DATA,
//                   DISP=(,CATLG),
//                   UNIT=SYSDA,
//                   SPACE=(CYL,5)
//SYSIN     DD       *
  SORT  FIELDS=(10,4,CH,A)
```

SYSOUT directs the sort messages to a SYSOUT class. SORTIN is the ddname identifying the input data to be sorted. After sorting, the data is written to the SORTOUT DD statement.

Sometimes the sort program requires sort work areas to be specified. These are unnamed temporary disk data sets with the ddname SORTWKO1,SORTWKO2, AND SORTWKO3. Consult the reference manual for your sort program to determine when you must include these statements.

Sort Control Statements

The SYSIN DD statement points to a control statement which tells the sort program:

- What fields to sort.
- What format the data is in.
- What order to sort into.

The format of the statement is either:

```
SORT FIELDS=(loc,len,format,seq...)
SORT FIELDS=(loc,len,seq...).FORMAT=format
```

where:

- loc indicates the location of the field which is the sort key. Location 1 is the first character is the record.
- len designates the number of characters in the sort field.
- format describes the format of data contained in the sort key.

  The common formats are:

CH	Character	ZD	Zoned Decimal	PD	Packed Decimal	BI	Binary

- seq specifies the sequence into which the records are to be stored.

  The two options are:

A	Ascending		D	Descending

Example:

```
SORT   FIELDS=(10,4,CH,A)
```

This statement sorts a file on a 4-byte long key containing character data. The key begins in column 10. Records are to be sorted into ascending sequence.

You can sort using multiple sort keys, by specifying location, length, format and sequence for each of the keys. For instance:

```
SORT   FIELDS=(26,8,PD,D,215,2,PD,A)
```

Here the file has two sort keys. The first is 8 bytes long starting in column 26. The second is 2 bytes long beginning in column 215. Both keys are in packed decimal format. The first key will be sequenced in descending order, the second in ascending order.

When multiple keys have the same data format, an optional form of the SORT statement can be used:

```
SORT   FIELDS=(26,8,D,215,2,A),FORMAT=PD
```

This form simplifies the coding that must be done on files with several sort keys. It is processed the same as the first form.

2 SyncSort: Product Information

SyncSort is a high performance sort/merge/copy utility.

SyncSort is designed to conserve system resources, provide significant performance benefits and operate efficiently.

SyncSort can be initiated through job control language or invoked from a program written in COBOL, PL/1 or Assembler language.

Exit routines may be written in COBOL, Fortran or Assembler language to give a JCL sort additional programming flexibility.

SyncSort's Basic Functions

SyncSort has three basic functions:

Sorting	Rearranging data set records to produce a specific sequence.
Merging	Combining up to 32 pre-sequenced data sets into one data set which has the same sequence.
Copying	Reproducing a data set without going through the sorting process.

2.1 Sorting

SyncSort provides three sorting techniques:

Disk Sort	The standard sorting technique.
MAXSORT	A maximum capacity sorting technique with an enhanced breakpoint/restart capability. MAXSORT can sort any collection of data--regardless of size--using a limited amount of disk space.
TapeSort	An alternative sorting technique used when intermediate storage must be assigned to tape.

A sort logically consists of four phases:

1. The control statements and JCL information are read and analyzed and the operational parameters for the sort are established.

2. The input data is read into main storage and sorted.

3. If necessary, intermediate results are written to temporary storage devices.

4. The sorting process completes and the sorted data is written to the specified output device(s).

Merging	A merge combines up to 32 pre-sequenced data sets into one data set which has the same sequence. A merge has two phases which perform these functions: 1. The control statements and JCL information are read and analyzed and the operational parameters for the merge are established. 2. The files are merged and the merged data is written to the specified output device(s).
Copying	A copy reproduces a file, completely bypassing the sorting process. A copy has two phases which perform these functions: 1. The control statements and JCL information are read and analyzed and the operational parameters for the copy are established. 2. The copied file is written to the specified output device(s).

2.2 Utilities and Other Features

General Multiple Output

The multiple output facility (OUTFIL) allows multiple output files to be generated with just one pass of the sort. Each of these files can have unique specifications that determine:

- Which records are to be included.
- How the records are to be formatted.
- Which report capabilities are to be used.

Moreover, all these files can be written to the same output device, or each can be written to a different device.

Creating Reports

SyncSort's SortWriter feature (OUTFIL) allows the user to design comprehensive reports easily and efficiently.

SortWriter options allow the user to:

- Format output data with headers and trailers, which can include data fields.
- Realign totals, subtotals, record counts and subcounts.
- Pad records with blanks, characters and binary zeros.
- Convert and edit numeric data.

Automatic pagination, page numbering and dating are also provided.

Selecting Records, Reformatting Records and Summarizing Fields

Record selection, record reformatting and summing are other important SyncSort Data Utility features.

Record selection via the INCLUDE/OMIT feature permits certain records to be included in or omitted from an input data set based on comparisons between two data fields or between a data field and a constant.

Record reformatting after input and/or before output, provided by the INREC/OUTREC capability, allows the user to:

- Delete or repeat portions of records.
- Insert spaces, characters and binary zeros.
- Realign fields.
- Convert numeric data to its printable format.
- Convert data to its printable hexadecimal format.

The ability to delete irrelevant fields before sorting via INREC can provide important performance benefits.

The SUM features allows records with equal sort control fields to be deleted and optionally summarizes numeric fields on those records.

3 Program Applications & Data Sets

SyncSort's job control statements follow the standard operating system conventions described in the job control language manuals.

For every data set used, each program application will require:

- A JOB statement.
- An EXEC statement.
- A DD (data definition) statement.

The single exception to this is the dynamic allocation of work files via DYNALLOC or DYNATAPE.

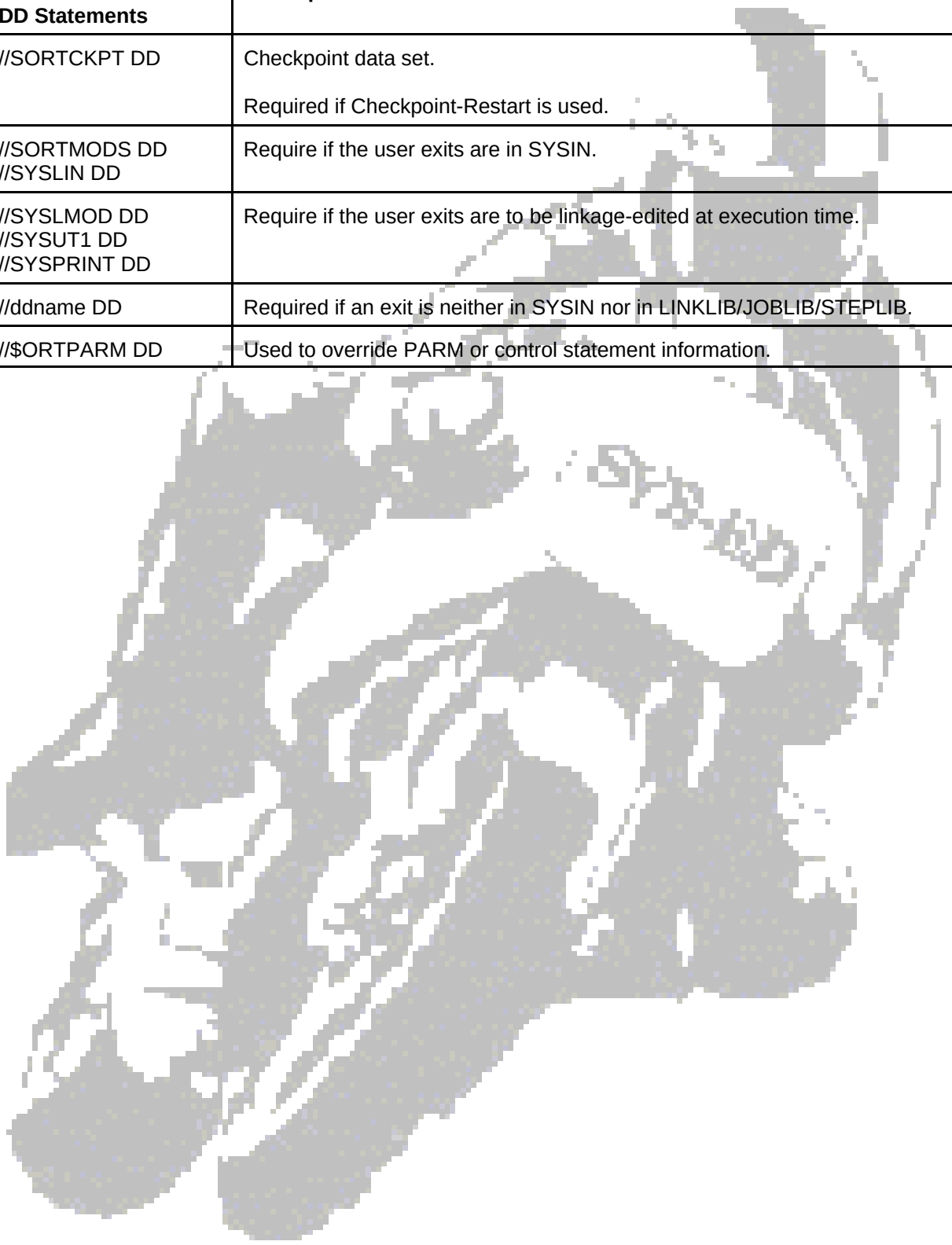
DD Statement Requirements

Disk Sort DD Statements	**Description**
//SYSIN DD	Control statement data set. Required unless the invoking program supplies the address of a 24-bit or a 31-bit, extended, parameter list.
//SYSOUT DD	Message data set. Required unless all messages are routed to console.
//SORTWKnn DD	Disk work area definition. Required unless incore sort, DYNALLOC used, MERGE, or restarting at a MAXSORT merge breakpoint.
//SORTIN DD	SORT input data set. Required unless there is an E15. Ignored if the invoking program supplies an inline E15 exit routine; optional if the MODS statement activates an E15 exit routine.
//SORTINnn DD	MERGE input data set. Required unless there is an E32.
//SORTOUT DD	Output data set. Required unless there is an E35. Ignored if the invoking program supplies an inline E35 exit routine; optional if the MODS statement activates an E35 exit routine.
//SORTOFxx DD //SORTOFx DD	Required if multiple output files are used.

Disk Sort DD Statements	**Description**
//SORTCKPT DD	Checkpoint data set. Required if Checkpoint-Restart is used.
//SORTMODS DD //SYSLIN DD	Require if the user exits are in SYSIN.

//SYSLMOD DD
//SYSUT1 DD
//SYSPRINT DD
	Require if the user exits are to be linkage-edited at execution time.

//ddname DD	Required if an exit is neither in SYSIN nor in LINKLIB/JOBLIB/STEPLIB.
//$ORTPARM DD	Used to override PARM or control statement information.



3.1 EXEC Statement

The EXEC statement indicates to the operating system that the job is a sort/merge application.

```
//stepname  EXEC { PGM'SYNCSORT  }  [ ,PARM')...)  ]
                 { PGM'SORT     }
                 { PGM'IERRCO00 }
                 { PGM'IGHRCO00 }
                 { PGM'ICEMAN   }
```

The PARM parameter may be used to pass the sort/merge program a variety of keyword parameters, modifying it to meet the needs of the individual application.

3.2 SYSOUT DD Statement

This statement defines the data set for SyncSort messages.

```
//SYSOUT DD    SYSOUT'A
```

If the SYSOUT DD statement is omitted, any message routed to the printer will be diverted to the console.

3.3 SORTIN DD Statement

The SORTIN DD statement defines the data set to be sorted or copied. The input file for a merge application is defined by the SORTINnn DD statement.

The SORTIN file must have physical sequential organization or be a member of a partitioned data set. The AMP subparameters BUFND, BUFNI, and/or BUFSP are recommended to optimize performance on a VSAM KSDS.

3.4 Concatenating Input Data Sets

Any number of data sets may be concatenated to SORTIN provided that the operating system supports concatenation and the RECFMs are compatible.

It is possible to merge up to 32 data sets. Each input data set carries a SORTINnn name, with nn a two-digit decimal number between 01 and 32. The only restriction on this numbering is that each file receive a different number. Numbers may be skipped, or used out of order. There are no restrictions as to which input files are to receive which numbers.

Each input data set must have the same RECFM and the records in each file must already be in the desired sequence.

3.5 SORTOUT DD Statement

The SORTOUT, SORTos, SORTnn statements are used to define one or more output files. These output data sets may be directed to a BSAM or VSAM supported device.

The AMP subparameters BUFND, BUFNI, and/or BUFSP are recommended to optimize performance on a VSAM KSDS.

3.6 SORTWKnn DD Statement

Up to 32 data sets may be specified for intermediate storage when sorting. Each work file carries a SORTWKnn name, with nn a two-digit number between 01 and 32. Each SORTWKnn must be allocated on a single unit and a single volume.

Although SORTWK space can be allocated in blocks, tracks, or cylinders, allocating in cylinders will yield optimal performance. The CONTIG option of the SPACE parameter should be avoided since it may delay allocation and offers no performance advantage. There is no need to specify RLSE and a secondary allocation value on the SORTWKnn DD statement at installations that have set these defaults at SyncSort installation time.

SORTWKnn DD statements are not used for merge or copy applications. They are not required for sorts executed using the DYNALLOC option.

3.7 Tape Sort

When intermediate storage is on tape, from 3 to 32 data sets may be specified. Tape SORTWKnn files must begin with SORTWK01 and be numbered consecutively.

3.8 SYSIN DD Statement

The data set defined by the SYSIN DD statement contains SyncSort control statements. The SYSIN DD statement is required in order to initiate the sort/merge through job control language.

3.9 $ORTPARM DD Statement

The data set defined by the $ORTPARM DD statement may contain PARM parameters and any of the sort control statements. Parameters and control statements passed via the $ORTPARM DD statement generally override all other passed, whether the sort/merge is called from a program or initiated through job control language.

The $ORTPARM data sets must be formatted in accordance with the following rules:

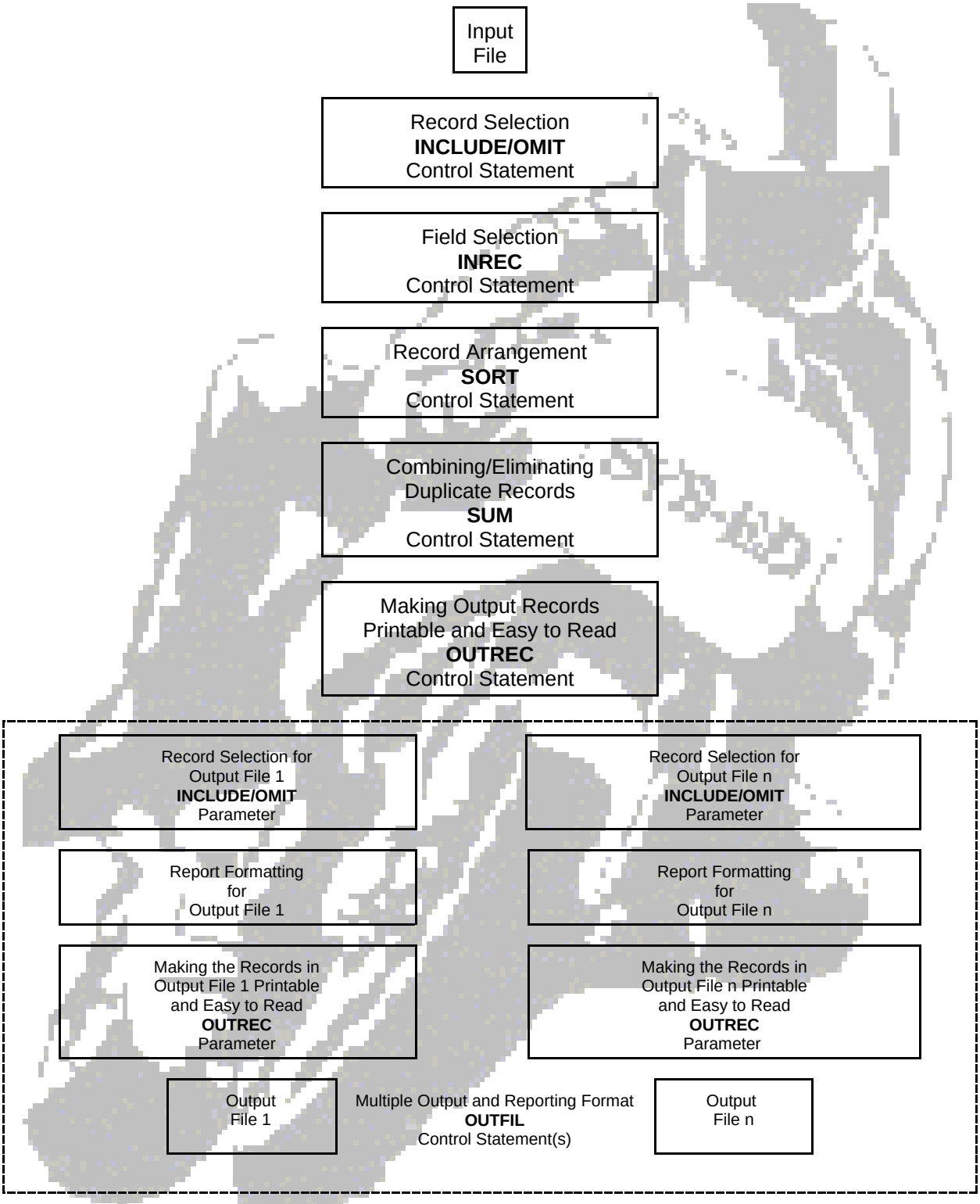
- PARM specifications included in the $ORTPARM data sets must be specified before any sort control statement specification.
- PARMS must be specified without the keyword PARM= and without quote marks.
- A comma in columns 2 - 70 of a PARM card image followed by a blank, or a comma alone in column 71, may be used to indicate that the next record is part of the current statement.

3.10 Exit Routines Requiring Linkediting at Execution Time

The following DD statements are required whenever an exit routine is to be linkedited at execution time.

SORTMODS DD	The partitioned data set defined must be large enough to contain all the exit routes entered in SYSIN.
SYSLIN DD	The SYSLIN DD statement defines the temporary data set that will contain the linkage editor control statements created by SyncSort for the exit routine(s).
SYSLMOD DD	The SYSLMOD DD statement defines the temporary data set that will contain the linkedited exit module(s).
SYSPRINT DD	The SYSPRINT DD statement defines the message data set for the linkediting of sort exits.
SYSUT1 DD	The SYSUT1 DD statement is used to define the temporary data set used as a work area when SyncSort linkedits an exit routine.

4 Data Utility Processing Sequence



5 Coding Considerations

Control cards are coded starting in column 2 and beyond.

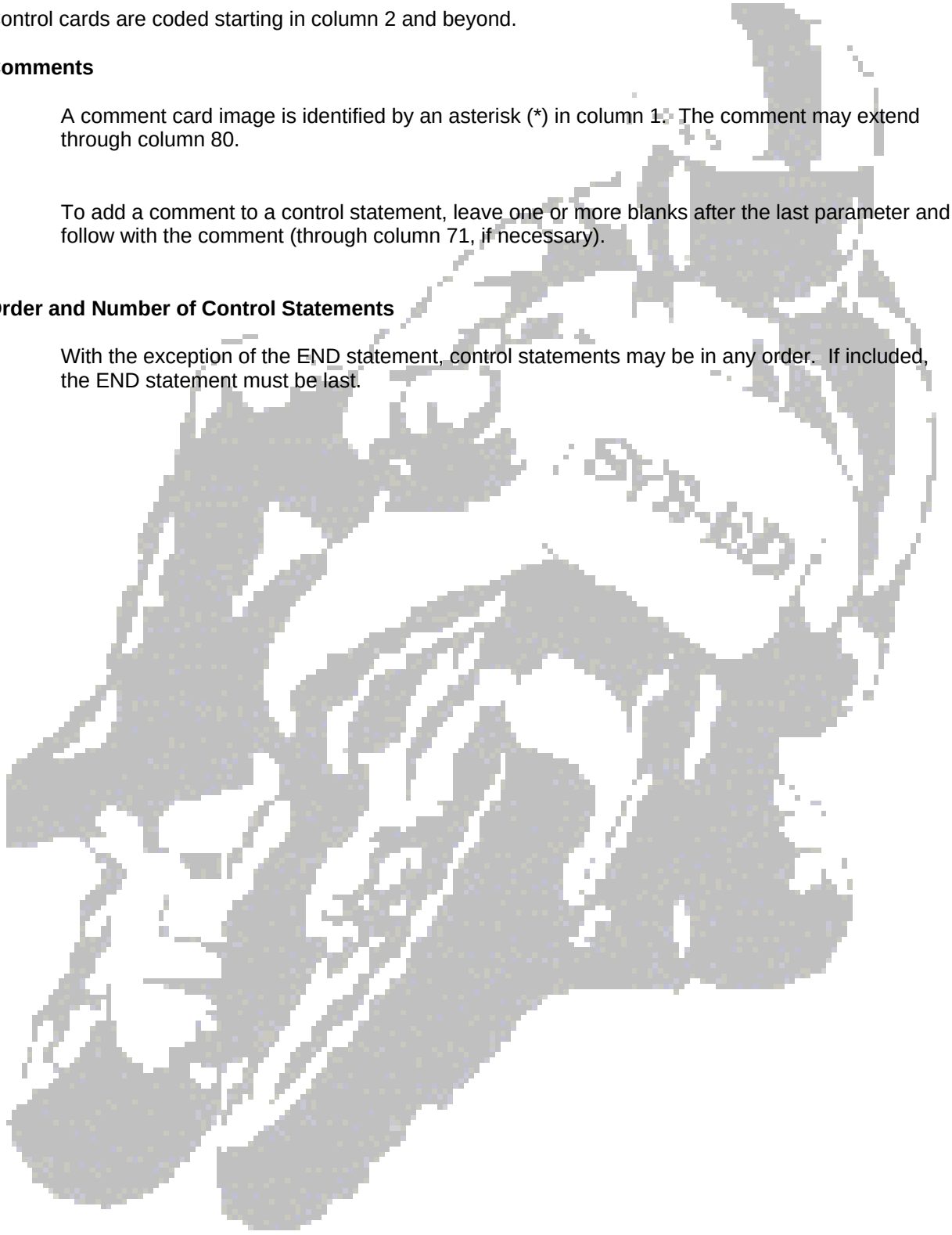
Comments

- A comment card image is identified by an asterisk (*) in column 1. The comment may extend through column 80.

- To add a comment to a control statement, leave one or more blanks after the last parameter and follow with the comment (through column 71, if necessary).

Order and Number of Control Statements

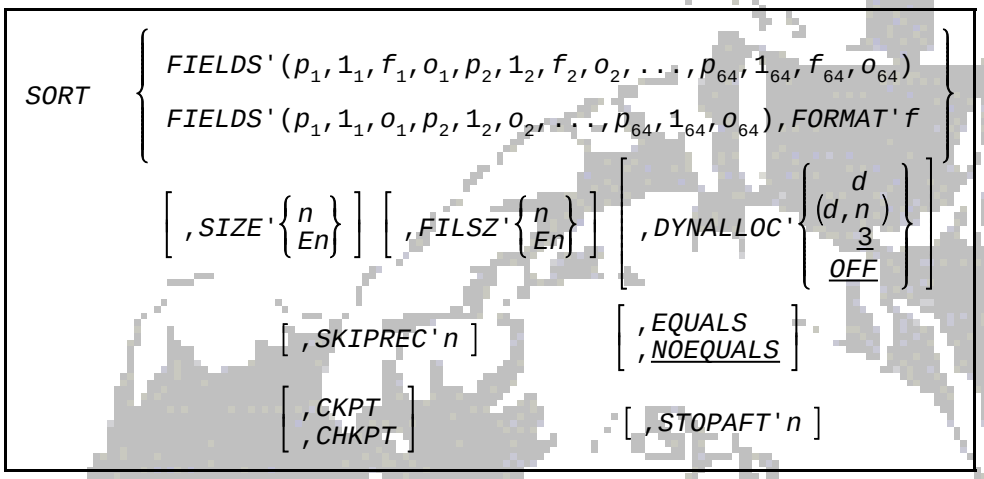
- With the exception of the END statement, control statements may be in any order. If included, the END statement must be last.



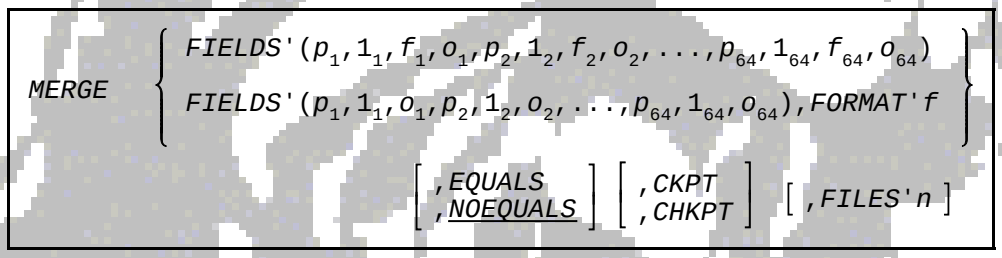
6 The SORT/MERGE Control Statement

The SORT/MERGE control statement is required for every SyncSort execution. It defines the job as a sort, merge or copy application.

SORT/MERGE Control Statement Format for a Sort



SORT/M
ERGE Control Statement Format for a Merge



SORT/MERGE Control Statement Format for a Copy

6.1 FIELDS Parameter

The FIELDS parameter is required and serves to define a copy application (FIELDS=COPY) or to describe the control fields for a sort or merge. As many as 64 control fields may be specified, listed in order of greatest to least priority. The most significant control field is listed first, followed by less and less important fields.

When all control fields have the same data format, the value for the f variable may be supplied once across all fields by way of the FORMAT=f parameter. When the BI field does not begin on a byte boundary, the bit numbers (0-7) must be coded.

The first four bytes of a variable length record are reserved for the Record Description Word (RDW). Therefore, the first byte of the data portion of the record is byte 5 and the 4th bit of the 21st byte of data is specified as 25.3.

Length values are specified in the same way: a number of bytes, optionally followed by a decimal point and a number of bits in the case of a BI control field. A length value of 0.5 refers to a control field 5 bits long.

The format code specifies the collating process relevant to the data in the associated control field(s). The order (o) value indicates how that collating sequence is referenced:

A	Ascending order.
D	Descending order.
E	As modified by an E61 exit. Ascending order.

Format Code Chart

Code	Data Format
*AC	EBCDIC characters are translated to their ASCII equivalents before sorting.
*AQ	Character. Records are sorted according to an alternate sequence specified either in the ALTSEQ control statement or as an installation default.
*ASL	Leading separate sign. An ASCII + or - precedes numeric fields. One digit per byte.
*AST	Trailing separate sign. An ASCII + or - trails numeric field. One digit per byte.
BI	Binary. Unsigned.
CH	Character. Unsigned.
*CSL or *LS	Leading separate sign.
*CST or *TS	Trailing separate sign.
FI	Fixed point. Signed
FL	Floating point. Normalized. Signed.
PD	Packed decimal. Signed.
ZD or *CTO	Zone decimal.

* Cannot be used with Tape Sort.
** 4084 for variable-length records.

6.2 The COPY Option

The coding constraints imposed by FIELD=COPY are the same, whether the operation name is SORT or MERGE. FIELDS=COPY does not process SORTINnn DD statements or the SUM control statement. SORTIN and SORTOUT DD statements are required in the JCL and, when necessary, may be coded with DUMMY data sets.

6.3 SIZE Parameter

SIZE=n gives the actual (SIZE=En, the estimated) decimal number of records that are to be expected from the input files(s). Unlike the FILSZ parameter, SIZE does not allow for any record additions or deletions made via an exit program or the INCLUDE/OMIT control statement.

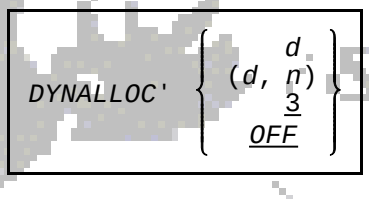
6.4 FILSZ Parameter

FILSZ=n gives the actual (FILSZ=En, the estimated) decimal number of records that are to be sorted or merged, taking into account any changes introduced by the INCLUDE/OMIT control statement, an E14, E15 or E32 exit routine. FILSZ will also take into account those fixed length records deleted by the SUM control statement in Phase 1. FILSZ=n instructs SyncSort to terminate unless exactly n records are to be sorted or merged.

If FILSZ has been specified in the PARM field as well as on the SORT/MERGE statement, the PARM specification will take precedence.

6.5 DYNALLOC Parameter

The MVS operating system is required in order to use the DYNALLOC feature, which dynamically allocates SORTWK data sets. DYNALLOC data sets supplement any SORTWKnn DD statements that are supplied to the sort.



6.6 SKIPREC Parameter

SKIPREC=n instructs the sort to skip a decimal number, n, of records before sorting or copying the input files, that is, before E15 exit and INCLUDE/OMIT control statement processing is begun.

6.7 EQUALS/NONEQUALS Parameter

EQUALS, when specified for a sort application, acts to preserve the original order of records with equal control fields--these records will be in the same order in the output file as they were in the input file.

When specified for a merge application, EQUALS guarantees that whenever equal keyed records appear in different SORTIN data sets, the record from the lowest numbered SORTINnn will be written first to the output file. Whether or not EQUALS is specified, the order of equal-keyed records within a SORTINnn file is preserved during a merge.

When used in conjunction with SUM, EQUALS indicates which of the equal-keyed records will be preserved, containing the sum: the record occurring first in the input file (for a sort) or drawn from the lowest numbered SORTINnn file (for a merge) will contain the totaled fields.

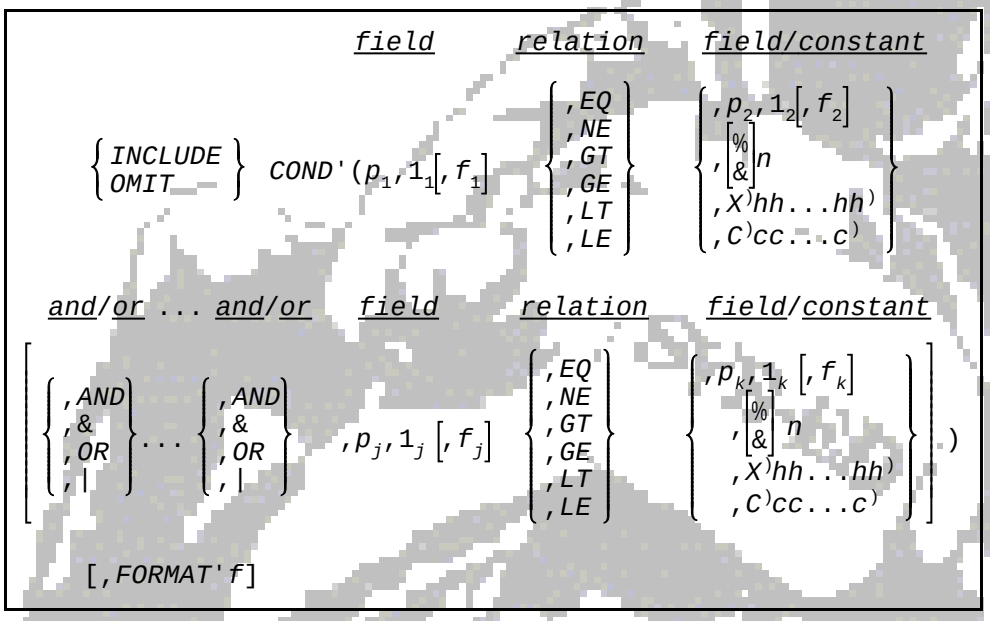
6.8 STOPAFT Parameter

When STOPAFT=n (a decimal number) is specified, at most n records will be sorted or copied. These will be the first n records after any input processing due to an E15 exit, an INCLUDE/OMIT statement or the SKIPREC parameter.

7 The INCLUDE/OMIT Control Statement

The INCLUDE/OMIT control statement is used to select records from a general input file based on comparisons testing the contents of various fields within the record.

These fields may be compared to a constant or to another field within the same record. The INCLUDE/OMIT control statement may appear only once per program execution, either as an INCLUDE or as an OMIT statement. This restriction does not apply to the INCLUDE/OMIT parameter on the OUTFIL statement.



7.1 COND Parameter

The COND parameter specifies the condition(s) under which records are to be included in or omitted from the sort/merge. Each condition is based on a comparison between a record field and a constant, or between two record fields. Any number of conditions may be set, provided these are joined by ANDs (&) and ORs (|). All fields must be located within the first 4092 bytes of the record. All references to fields within the data record, included in the INCLUDE/OMIT control statement, refer to the record after processing by an E15 exit.

Constant coding follows the usual conventions, with hexadecimal and character literals enclosed in single quotes (double an apostrophe within a character literal, as in C'D"ARTAGNAN') and the hexadecimal constant expressed in pairs of valid hexadecimal digits.

7.2 Field-to-Constant Comparison Format

Field-to-constant comparisons are of two kinds:

Decimal Comparisons	Compares PD, ZD, or FI fields to a decimal constant.
Non-Decimal Comparisons	Compares BI or CH fields to a hexadecimal or character literal.

When the field and the constant are of different lengths, the length of the constant is adjusted. Hexadecimal and character literals are padded/truncated on the right, decimal constants on the left. X'00' bytes are used to pad short constants except when a ZD field or C'cc...c' literal is involved. A ZD field calls for X'FO' padding for the decimal constant; a character literal is padded with X'40'.

7.3 Combining Comparisons Using AND's and OR's

The INCLUDE/OMIT control statement may combine several conditions using the Boolean connectives AND and OR in order to determine whether SyncSort should process the record.

Unless internal parentheses intervene, AND conditions are evaluated before OR conditions. Conditions enclosed within internal parentheses are evaluated first. Only one level of internal parentheses is permitted.

8 The INREC Control Statement

The INREC control statement is used to reformat input records on the way into the sort/merge.

It provides the following capabilities:

- Deletion or repetition of individual segments of the input record.
- Insertion of character strings between data fields.
- Insertion of binary zeros.
- Conversion of numeric data to printable format.
- Conversion of data to hexadecimal format.
- Realignment of data fields.

Since execution time is determined in part by the total number of bytes the sort/merge must handle, it is generally advisable to rely on INREC processing for procedures that shorten records--for example, deleting fields that are not relevant to the application.

```
                    INREC FIELDS'(field[,field]...)
&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
INREC fields:
      ⎧                                                                      ⎫
      ⎪    ⎧ X                                                               ⎪
      ⎪ [n]⎨ X'hhhh...hh)                                                    ⎪
      ⎪    ⎩ C'literal string)                                               ⎪
[c:]  ⎨    Z                                                                 ⎪
      ⎪                                                                      ⎪
      ⎪    ⎧ a                                                               ⎪
      ⎪ p1,11[,⎨ f (,[ Mm                                                    ⎪
      ⎪    ⎩   EDIT'(pattern)] ) [,SIGNS'(s1,s2,s3,s4)] [LENGTH'(n)]]        ⎪
      ⎪                                                                      ⎪
      ⎩ p1,[,11][,HEX]                                                       ⎭
```

8.1 Formatting the Reformatted Records

Column Alignment	Established by the (c:) subparameter. The numeric specified for this entry defines the position of the first column of the designated field within the reformatted record. SyncSort will automatically add the number of blanks necessary to achieve proper alignment. A maximum of 4092 bytes may be specified between any two column definitions.
Literal Fields	Included in the reformatted records may be hexadecimal characters, literal characters or blank spaces. These subparameters are specified in conjunction with a repetition factor, 'n', which may be any number between 1 and 256 inclusive. The literal field may be up to 256 characters long. The repetition factor times the literal field cannot exceed 4092 characters.
Hexadecimal Characters	Can be inserted in the reformatted records by specifying the nX'h' entry in the FIELDS parameter. The number of times the hexadecimal character(s) is to be repeated is defined by the number specified for 'n'. The X'h' entry should be coded to the immediate right of the number specified for 'n', indicating that 'n' copies of the hexadecimal character(s) 'hhh...' are to be inserted in the reformatted record. A maximum of 256 pairs of hexadecimal characters may be specified.
Literal Characters	May be inserted in the reformatted record by specifying the nC'c' entry in the FIELDS parameter. The number of times that the literal character(s) is to be repeated is defined by the number specified for 'n'. Therefore, the nC'c' entry is used to specify that 'n' copies of the literal character or characters, 'cc...' are to be inserted in the reformatted record. An apostrophe within a character literal must be specified as a pair of consecutive apostrophes (e.g. C'O"LEARY').
Blank or Spaces	Can be inserted in the reformatted record by coding the nX entry in the FIELDS parameter. The number of blanks or spaces is defined by the number specified for 'n'. The X entry represents spaces and must be coded to the immediate right of the number specified for 'n'. Since 'n' may only represent a number between 1 and 256 inclusive, if more than 256 spaces are desired, two or more nX values should be coded. For example, 256x,256x,61x will provide 573 spaces. The nX entry may not be used; the Record Descriptor Word must be accounted for.
Binary Zeros	Can be inserted in the reformatted record by coding the nZ entry in the FIELDS parameter. The number of bytes of binary zeros is defined by the number specified for 'n';'n' may be any number from 1 to 256 inclusive. The Z entry represents binary zeros and must be coded to the immediate right of the number specified for 'n'.

Hexadecimal Conversion	Conversion of any field within a record may be achieved by specifying p,l,HEX for both fixed-length records and the fixed-length portion of variable-length records or p,HEX for the variable-length portion of variable-length records. Starting in position p of the input record, for a length of l, each hex digit will be converted to its printable representation. In the reformatted record, the converted field will double in length.

8.2 Unedited Input Record Fields

Input record fields that do not require editing--but that are nonetheless necessary for an application--must be coded. These fields are specified using the position, length, and alignment subparameters.

8.3 Data Conversion and Editing

Data conversion is achieved through editing masks, the format, LENGTH, SIGNS, and EDIT subparameters of the INREC and OUTREC control statements, as well as the OUTREC, TRAILER1, TRAILER2, and SECTIONS parameters of the OUTFIL control statement. Data conversion is initiated by specifying the format (f) of the data in the appropriate subparameter.

If neither an editing mask nor the EDIT subparameter is specified, data is converted according to the default editing-mask pattern, M0. The LENGTH and SIGNS subparameters may be used to modify the default editing-mask pattern.

8.4 Edit Subparameter

The EDIT subparameter provides the user with the capability of creating individual patterns for editing converted numeric data.

An individual character pattern is composed of significant digit selectors, leading insignificant digit selectors, a sign replacement character, and all other characters that are to be reproduced in the actual output. The edit pattern may be up to 22 characters in length, with a maximum of 15 leading insignificant and/or significant digits.

```
                    EDxy'

where: x '   insignificant digit selector
       y '   significant digit selector
```

8.5 SIGNS Subparameter

```
                SIGNS'(s1,s2,s3,s4)

where: s1 '  leading positive sign indicator
       s2 '  leading negative sign indicator
       s3 '  trailing positive sign indicator
       s4 '  trailing negative sign indicator
```

9 The OUTREC Control Statement

The OUTREC control statement is used to reformat records as they leave the sort/merge.

All references to fields within the data record included in the OUTREC control statement refer to the record after processing by an E15 exit, the INREC control statement, and the SUM control statement.

Sort applications with OUTREC require a minimum of one disk file for sort work and are not compatible with an incore sort.

It provides the following capabilities:

- Deletion or repetition of individual segments of the input record.
- Insertion of character strings between data fields.
- Insertion of binary zeroes.
- Conversion of numeric data to printable format.
- Conversion of data to hexadecimal format.
- Realignment of data fields.

OUTREC reformatting is virtually the same as INREC reformatting, and the OUTREC control statement is coded in the same format.

```
                    OUTREC  FIELDS'(field[,field]...)
&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
OUTREC  fields:
        {        { X              }                                                   }
        {        { X'hhhh...hh)   }                                                   }
        { [n]    { C'cccc...cc)   }                                                   }
        {        { Z              }                                                   }
 [c:]   {                                                                             }
        {        [   { a                                                         ]    }
        { p1,11  [ , { f (,[ Mm                ] ) [,SIGNS'(s1,s2,s3,s4)] [LENGTH'(n)] ]    }
        {        [   {       EDIT'(pattern)                                      ]    }
        { p1,[,11][,HEX]                                                              }
```

10 The OUTFIL Control Statement

The OUTFIL control statement describes the output file or files to the sort. It is required when:

1. Creating multiple output files.
2. Using the report writing facility.
3. Reformatting records after E35 processing.

OUTFIL requires a minimum of one disk SORTWK file; this minimum, however, does not hold for MERGE or SORT/MERGE FIELDS=COPY applications. OUTFIL is not compatible with an incore sort or MAXSORT.

```
OUTFIL  [FILES'(fileid [,fileid]...)] [{,INCLUDE} ,{ALL                 }]
                                      [{,OMIT   }  {(logical expression)}]
                                      [            {NONE                }]

        [,OUTREC'(field [,field]...)]
        [,HEADER1'(f1[,f2]...)]        [,HEADER2'(f1[,f2]...)]
        [,TRAILER1'(f1[,f2]...)]       [,TRAILER2'(f1[,f2]...)]
        [,LINES'n          ]
        [,LINES'ANSI       ]           [,SECTIONS'(f1[,f2]...)]
        [,LINES'(ANSI,n)   ]
                                       [,NODETAIL]
```

10.1 Creating Multiple Output Files with the OUTFIL Control Statement

OUTFIL can be used to create multiple output files without making multiple passes through the input data. Whether the data in each of these multiple output files is the same or different is determined by OUTFIL's FILES, INCLUDE/OMIT, and OUTREC parameters.

```
FILES'(fileid [,fileid]...)

                  {OUT}
where: fileid '   {x  }
                  {xx }
```

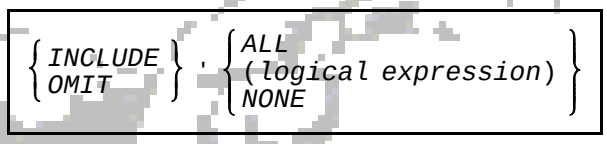
The FILES parameter connects the OUTFIL control statement with one or more output files. When creating multiple output files, this parameter is required to specify which output file or files the OUTFIL statement is describing.

10.2 INCLUDE/OMIT Parameter

The INCLUDE/OMIT control statement, COND parameter, provides the detailed format of logical expression.

The FORMAT=f parameter, which is permitted on the INCLUDE/OMIT control statement, is not permitted on the INCLUDE/OMIT parameter. The formats of fields must be specified on a field-by-field basis.

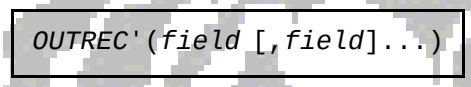
The INCLUDE/OMIT parameter indicates which records are to be included in or omitted from each output file. The default for this parameter is to include ALL records leaving the sort/merge in the output file(s) referred to by the OUTFIL control statement. Otherwise, one or more logical conditions, under which records are to be included in or omitted from the output file(s), are specified. But when no records are to be included in the output file(s), such as, when running a test, either INCLUDE=NONE or OMIT=ALL must be specified.



10.3 OUTREC Parameter

The OUTREC parameter provides for the reformatting, after any processing by an E35 exit, of records that are to be included in the output file(s).

If however, no additional reformatting is required, then this parameter should be omitted.



11 The RECORD Control Statement

The RECORD control statement is used to optimize sort performance for variable length records or to override DCB information drawn from the data set label of the input file(s).

The RECORD control statement is required whenever:

- Initiating the sort/merge from a program passing either a 24-bit or 31-bit, extended, parameter list and using an E15 or E32 exit routine.
- An E15 or E35 exit routine changes the record length.
- SORTIN (or SORTINnn) is a VSAM data set and SORTOUT is a VSAM or variable length data set.

$$RECORD\ \ TYPE'\ \left\{\begin{matrix}F\\V\end{matrix}\right\}\left[\ ,LENGTH'(1_1,1_2,1_3,1_4,1_5,1_6,1_7)\ \right]$$

11.1 TYPE Parameter

This required parameter specifies whether the records are fixed (F) or variable (V) length. TYPE=FB or VB may be specified, but the B is ignored.

11.2 LENGTH Parameter

Three length values (1_1-1_3) can be specified for fixed length records and non-sorting applications; seven, length values for sorting variable length records.

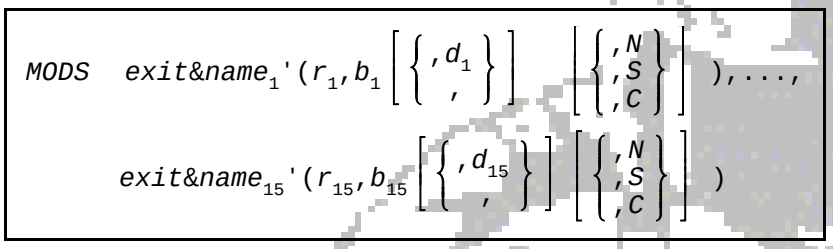
The last two 1-values, 1_6 and 1_7, only apply to the Disk Sort and MAXSORT techniques.

1_1	Refers to the maximum length of the records in the input data set.
1_2	Gives the maximum record length after E15 exit processing.
1_3	Gives the maximum record length after E35 exit processing.

12 The MODS Control Statement

The MODS control statement is used to activate program exits.

Whenever an exit is activated, SyncSort transfers control to the routine designated for that particular exit in the sort/merge.



12.1 Exit-Name Parameter

There are 16 program exits available through the MODS statement. Each has a unique exit name.

EXIT NAME	Sort Phase 1	Sort Phase 2	Sort, Merge, or Copy Phase 3	All Phases
	E11, E14, E15, E16, E17, E18	E21, E25, E27	E31, E32, E35, E37, E38, E39	E61

This parameter has these additonal values:

Value	Explanation
r	Gives the name of the routine to be included at this exit point.
b	Gives either the exact or the estimated decimal number of bytes the exit routine requires in main storage.
d	Gives the DD statement name that defines the library where the exit routine resides.
N	Code N when linkediting is not required.
S	Code S when linkediting is required, and the exit routine can be linkedited separately from other exit routines for the same phase.
C	Code C when the exit is COBOL routine.

13 The SUM Control Statement

The optional SUM control statement affects the processing of equal keyed records by indicating which numeric fields (if any) are to be treated as summary fields.

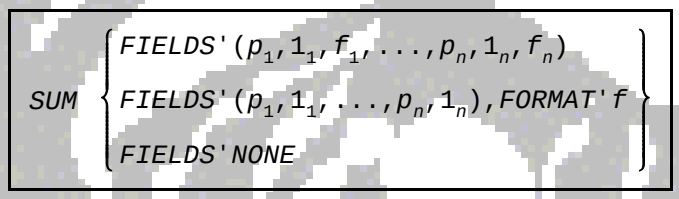
Equal-keyed records are processed pair by pair. Provided the data in their corresponding summary fields total to a number which can be obtained in that field, one record will be preserved containing these totals, and one record will be deleted. Non-summary fields remain unchanged and are picked up from the record which receives the sum.

Assuming that arithmetic overflow does not occur (guaranteed with SUM FIELDS=NONE), there will be only one record per sort key in the SORTOUT data set; those fields designated as summary fields contain the arithmetic sums of the data found in all the sorted/merged records having the same key.

Unless EQUALS is specified on the SORT/MERGE control statement or as a PARM, it will not be possible to predict which record will be retained, containing these totals, and which will be deleted.

With EQUALS, the record occurring first in SORTIN (for a sort) or drawn from the lowest-numbered SORTINnn file (for a merge) receives the sum. When the records are found in the same SORTINnn file, the record occurring first is preserved whether or not EQUALS is specified.

An overflow condition never destroys data. If SUMing a particular record pair would cause arithmetic overflow, the pair is not summarized and neither record is deleted. The SUM process continues, however, so that any subsequent pair of records which can be summarized without overflow will receive SUM processing.



13.1 FIELDS Parameter

The FIELDS parameter defines the numeric fields to be summed when the control fields of two or more records are equal. FIELDS=NONE permits the user to reduce the output file to one record per sort key without doing any summarization.

Summary fields are referenced by position (p) and length (l); all fields must be located within the first 4096 bytes of the record.

Five numeric format codes (f) are permitted: BI, FI, PD, ZD, and FL.

The FORMAT=f option is used when all of the fields specified in the control statement have the same format code.

14 The END Control Statement

The END control statement signals the end of the SyncSort control statements in the input job stream.