IBM Library Server Print Preview

DOCNUM = SC26-7041-03
DATETIME = 04/27/98 08:39:45
BLDVERS = 1.3.0
TITLE = DFSORT/VSE Installation and Tuning Guide
AUTHOR =

PATH = /home/webapps/epubs/htdocs/book

COVER Book Cover

DFSORT/VSE

Installation and Tuning Guide

Version 3 Release 4

Document Number SC26-7041-03

Program Number
5746-SM3

NOTICES Notices

Note!
Before using this information and the product it supports, be sure to read the general information under "Notices" in topic FRONT_1.

EDITION Edition Notice

Fourth Edition (May 1998)

This edition replaces and makes obsolete the previous edition, SC26-7041-02. The technical changes for this edition are summarized under "Summary of Changes," and are indicated by a vertical bar to the left of a change. A vertical bar to the left of a figure caption indicates that the figure has changed. Editorial changes that have no technical significance are not noted.

This edition applies to Version 3 Release 4 of DFSORT/VSE, Program Number 5746-SM3, and to any subsequent releases until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

A form for reader's comments is provided at the back of this publication. Comments may be addressed to:

- International Business Machines Corporation
- RCF Processing Department
- G26/026
- 5600 Cottle Road
- San Jose, CA, 95193-0001
- U.S.A.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

CONTENTS Table of Contents

FIGURES Figures

FRONT_1 Notices

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 product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to :

- IBM Director of Licensing
- IBM Corporation
- 500 Columbus Avenue
- Thornwood, NY 10594
- U.S.A.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Subtopics:

FRONT_1.1 Programming Interface Information

This book primarily documents information that is NOT intended to be used as a Programming Interface of DFSORT/VSE.

This book also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of DFSORT/VSE. This information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

```
 ------------------ Programming Interface information -------------------


|--------------- End of Programming Interface information ---------------|
```

FRONT_1.2 Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

DFSORT	ESA/390	LE/VSE
ECKD	IBM	VSE/ESA
ESA/370	Language Environment	VM/ESA

The following terms are trademarks of other companies:

CA-DYNAM	Computer Associates International, Inc.
CA-SORT	Computer Associates International, Inc.
X/Open	X/Open Company Limited

PREFACE Preface

This book provides information about installing, customizing, and tuning DFSORT/VSE:

- Installing and Customizing

  Planning is the task of making fundamental decisions about the options a program offers. These decisions guide, set limits for, and identify requirements for the tasks of installation, customization, administration, application programming, and diagnosis. Installation is the task of making a program ready to do useful work. This task includes adding the materials on the IBM distribution tape to your system, initializing the program, and applying PTFs to the program. When you install a product, you are carrying out decisions you made in the planning step. Customization, an optional step, gives you the opportunity to tailor the program to the needs of the users at your site.

This book is designed to complement the *DFSORT/VSE Program Directory*. This book covers the planning and customization tasks; the *DFSORT/VSE Program Directory* covers the installation task. Read all of Part 1, "Installation and Customization" before installing DFSORT/VSE with the step-by-step instructions in the *DFSORT/VSE Program Directory*.

The *DFSORT/VSE Program Directory* is shipped in the same package as the DFSORT/VSE installation tape. It describes all of the installation materials and gives installation instructions specific to the product release level. If any differences exist between this book and the *DFSORT/VSE Program Directory*, use the information in the *DFSORT/VSE Program Directory*.

- Tuning

Sorting is one of the most frequently used functions at most data processing sites. This book provides information about tuning DFSORT/VSE, offering suggestions for reducing its use of system resources and achieving better turnaround time for the many applications that use DFSORT/VSE.

This book is intended for system programmers and system administrators who are responsible for installing and customizing DFSORT/VSE, and systems engineers, performance analysts, and application programmers who have some experience with the DFSORT/VSE program product, 5746-SM3.

Subtopics:

PREFACE.1 About This Book

The information in this book is organized into the following sections:

- Part 1, "Installation and Customization"
  - Chapter 1, "Planning for Installation" in topic 1.1, provides the information needed to plan the installation of DFSORT/VSE, including discussions of hardware, software, and storage requirements for installation.
  - Chapter 2, "Installing DFSORT/VSE" in topic 1.2, describes how to install DFSORT/VSE and verify the installation.

- Chapter 3, "Planning for Customization" in topic 1.3, discusses items you need to think about before customizing DFSORT/VSE.

- Chapter 4, "Customizing DFSORT/VSE" in topic 1.4, provides the information and procedures needed to customize DFSORT/VSE.

- Part 2, "Tuning"

  - Chapter 5, "Introduction" in topic 2.1, outlines the importance of DFSORT/VSE at your site, its system resource usage, the potential benefits of tuning, and explains some basic terms and concepts used throughout the book.

  - Chapter 6, "Analyzing DFSORT/VSE Use" in topic 2.2, describes various techniques which enable you to understand the current use of DFSORT/VSE at your site before starting to tune.

  - Chapter 7, "Environment Considerations" in topic 2.3, describes DFSORT/VSE's relationship to software environments.

  - Chapter 8, "Installation and Run-Time Options" in topic 2.4, explains how installation options, run-time options, storage, and files affect the performance of DFSORT/VSE applications.

  - Chapter 9, "Application Considerations" in topic 2.5, describes how new and existing applications can make efficient and effective use of the DFSORT/VSE facilities.

- Appendix A, "Installation Messages" in topic A.0, lists messages that may be issued during installation.

- Appendix B, "Locales Supplied with C Run-Time Library" in topic B.0, lists the locales supplied with the LE/VSE 1.4 C Run-time library or VSE/ESA 2.2 C Run-time library.

- Appendix C, "The ILUMSGS Macro" in topic C.0, contains a copy of the ILUMSGS macro.

PREFACE.2 DFSORT/VSE Publications

DFSORT/VSE Installation and Tuning Guide is part of a more extensive DFSORT/VSE library. These books can help you use DFSORT/VSE more effectively. The books in the library are listed below.

Task	Publication	Order Number
Application programming	*DFSORT/VSE Application Programming Guide*	SC26-7040
Evaluating DFSORT/VSE	*DFSORT/VSE General Information*	GC26-7039
Learning to use DFSORT/VSE	*Getting Started with DFSORT/VSE*	SC26-7101
Diagnosing failures and interpreting messages	*DFSORT/VSE Messages, Codes and Diagnosis Guide*	SC26-7132
Quick reference	*DFSORT/VSE Reference Summary*	SX26-6008

You can order a complete set of DFSORT/VSE publications with the order number SBOF-6130, except for *DFSORT/VSE Licensed Program Specifications* (GC26-7038), which must be ordered separately.

Subtopics:

PREFACE.2.1 DFSORT/VSE Library Softcopy Information

A softcopy version of the DFSORT/VSE library is available on the CD-ROM shown in the table that follows. The *IBM Online Library VSE Collection* contains all of the DFSORT/VSE books for Releases 2, 3, and 4, with the exception of the *DFSORT/VSE Reference Summary*, and books from other VSE libraries.

Order Number	Title
SK2T-0060	*IBM Online Library VSE Collection*

PREFACE.3 DFSORT/VSE on the World Wide Web

For news, tips, and examples, visit the DFSORT/VSE home page at URL:

```
http://www.ibm.com/storage/dfsortvse/
```

PREFACE.4 Related Publications

The information presented in the following publications can help you to install and to use DFSORT/VSE.

Short Title	Publication	Order Number
Administration	*VSE/ESA Administration (for VSE/ESA Version 1 Release 3)*	SC33-6505
	VSE/ESA Administration (for VSE/ESA Version 2)	SC33-6605
Planning	*VSE/ESA Planning (for VSE/ESA Version 1 Release 3)*	SC33-6503
	VSE/ESA Planning (for VSE/ESA Version 2)	SC33-6603
Using COBOL and PL/I	*COBOL for VSE/ESA Programming Guide*	SC26-8072
	PL/I for VSE/ESA Programming Guide	SC26-8053

PREFACE.5 Referenced Publications

Within the text of this book, references are made to the following publications:

Short Title	Publication	Order Number
Application Programming Guide	*DFSORT/VSE Application Programming Guide*	SC26-7040
Messages, Codes and Diagnosis Guide	*DFSORT/VSE Messages, Codes and Diagnosis Guide*	SC26-7132

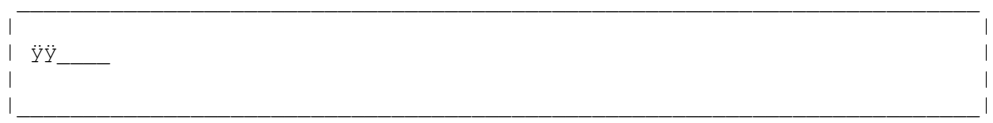
Getting Started with DFSORT/VSE	*Getting Started with DFSORT/VSE*	SC26-7101
VSE/ESA System Control Statements	*VSE/ESA System Control Statements (for VSE/ESA Version 1 Release 3)*	SC33-6513
	VSE/ESA System Control Statements (for VSE/ESA Version 2)	SC33-6613
VSE/ESA Installation	*VSE/ESA Installation and Service (for VSE/ESA Version 1 Release 3)*	SC33-6504
	VSE/ESA Installation (for VSE/ESA Version 2)	SC33-6604
VSE/ESA Extended Addressability	*VSE/ESA Extended Addressability (for VSE/ESA Version 1 Release 3)*	SC33-6524
	VSE/ESA Extended Addressability (for VSE/ESA Version 2)	SC33-6621
VSE/ESA Guide to System Functions	*VSE/ESA Guide to System Functions (for VSE/ESA Version 1 Release 3)*	SC33-6511
	VSE/ESA Guide to System Functions (for VSE/ESA Version 2)	SC33-6611
VSE/ESA Operation	*VSE/ESA Operation (for VSE/ESA Version 1 Release 3)*	SC33-6506
	VSE/ESA Operation (for VSE/ESA Version 2)	SC33-6606
VSE/POWER Administration and Operation	*VSE/POWER Administration and Operation (for VSE/ESA Version 1 Release 3)*	SC33-6571
	VSE/POWER Administration and Operation (for VSE/ESA Version 2)	SC33-6633
VSE/POWER Application Programming	*VSE/POWER Application Programming (for VSE/ESA Version 1 Release 3)*	SC33-6574
	VSE/POWER Application Programming (for VSE/ESA Version 2)	SC33-6636
COBOL Programming Guide	*COBOL for VSE/ESA Programming Guide*	SC26-8072
LE/VSE Library	*IBM Language Environment for VSE/ESA C Run-Time Programming Guide Release 4*	SC33-6688
	IBM Language Environment for VSE/ESA Installation and Customization Guide Release 4	SC33-6682

PREFACE.6 Notational Conventions

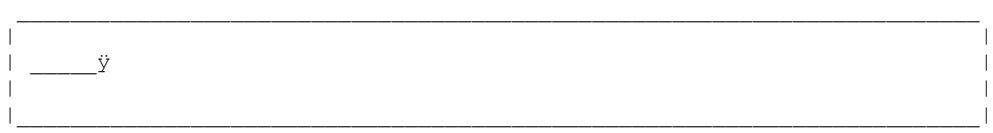
The syntax diagrams in this book are designed to make coding the DFSORT/VSE installation macro simple and unambiguous. The lines and arrows represent a path or flowchart that connects operators, parameters, and delimiters in the order and syntax in which they must appear in your completed statement. Construct a statement by tracing a path through the appropriate diagram that includes all the parameters you need, and code them in the order that the diagram requires you to follow. Any path through

the diagram gives you a correctly coded statement, if you observe these conventions:

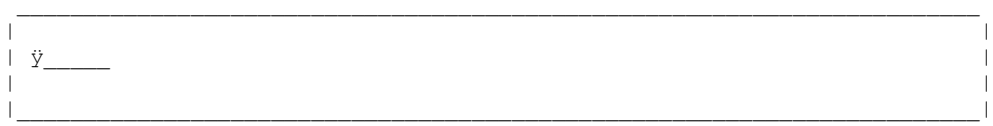
- Read the syntax diagrams from left to right and from top to bottom.
- Begin coding your statement at the spot marked with the double arrowhead.



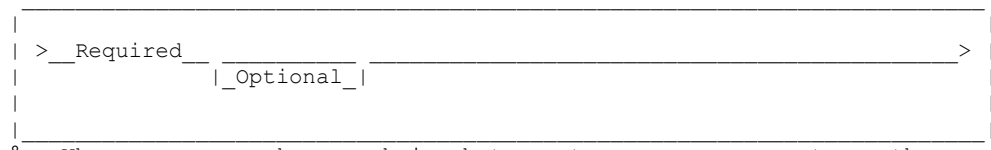
- A single arrowhead at the end of a line indicates that the diagram continues on the next line or at an indicated spot.



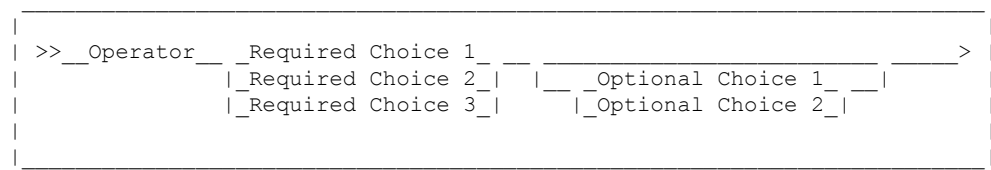
A continuation line begins with a single arrowhead.



- Strings in upper-case letters are operators and parameters, and must be coded exactly as shown. (The conventions require that at least one blank separates the initial operator from the succeeding parameters; no blanks are allowed between parameters.)
- Punctuation (parentheses, commas, and so on), must be coded exactly as shown.
- Strings in all lowercase letters represent information that you supply.
- Required parameters appear on the same horizontal line (the main path) as the operator, while optional parameters appear in a branch below the main path.

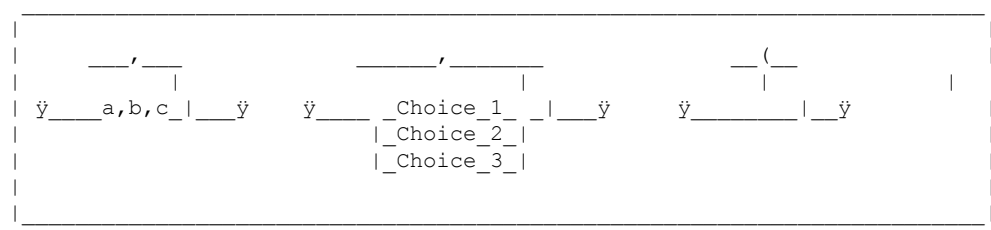


- Where you can make one choice between two or more parameters, the alternatives are stacked vertically.



If one choice within the stack lies on the main path (as in the example above, left), you *must* specify one of the alternatives. If the stack is placed below the main path (as in the example above, right), then selections are optional, and you can choose either one or none of them.

- The repeat symbol shows where you can return to an earlier position in the syntax diagram to specify a parameter more than once (see the leftmost example below), to specify more than one choice at a time from the same stack (see the center example below), or to nest parentheses (see the rightmost example below).



Do not interpret a repeat symbol to mean that you can specify incompatible parameters.

Use any punctuation or delimiters that appear within the repeat symbol to separate repeated items.

- A double arrowhead at the end of a line indicates the end of the syntax diagram.

FRONT_2 Summary of Changes

Subtopics:

FRONT_2.1 Fourth Edition, May 1998

Subtopics:

FRONT_2.1.1 New Programming Support for Release 4

DFSORT/VSE Version 3 Release 4 continues the strategy of providing performance improvements and productivity features. These improvements and features are described in more detail in the subsections that follow.

Subtopics:

FRONT_2.1.1.1 Performance

Performance enhancements for DFSORT/VSE Version 3 Release 4 include the following:

- Improved data processing methods for:
    - Dataspace and getvis sorting applications using work space
    - Merge and copy applications

- Improved input/output processing techniques for:
  - SAM output files
  - Non-VSAM input and output files
  - VSAM (and SAM ESDS accessed as VSAM) input and output files
  - Work files
- Improved ECKD disk device support for input, output, and work files by using the ECKD command set.
- New VSAMBSP installation option which allows users to control the number of buffers DFSORT/VSE can use for VSAM (or SAM ESDS accessed as VSAM) input and output file processing.
- Improved work file processing:
  - All work files are now closed at the end of an application.
  - Additional work file extents can now be used, if available, when end of extent is encountered regardless of whether STXIT is in effect.
  - All extents of an SD work file can now be used instead of only the first extent.

FRONT_2.1.1.2 Productivity

Additional Year 2000 Formats: New formats give users more flexibility in sorting, merging, or transforming two-digit year dates:

- Y2S interprets two-digit character year data according to the century window and allows special handling of indicators X'00' (binary zeros), X'40' (EBCDIC blanks), X'20' (ASCII blanks) and X'FF' (binary ones) in the year field.
- Y2B interprets two-digit binary year data according to the century window.

OUTREC Enhancements: The OUTREC control statement supports the following new features:

- Sophisticated editing capabilities such as hexadecimal display and control of the way numeric fields are presented with respect to length, leading or suppressed zeros, symbols (for example, the thousands separator and decimal point), leading

and trailing positive and negative signs, and so on. Twenty-six pre-defined editing masks are available for commonly used numeric editing patterns, encompassing many of the numeric notations used throughout the world. In addition, a virtually unlimited number of numeric editing patterns are available via user-defined editing masks.

- Selection of a character or hexadecimal string for output from a lookup table, based on a character, hexadecimal, or bit string as input (that is, lookup and change).

INCLUDE/OMIT Enhancements: The following INCLUDE/OMIT enhancements are supported:

- DFSORT/VSE can now handle a significantly larger number of INCLUDE and OMIT conditions.
- ALL and NONE allow users to include or omit all records.

ZDPRINT Option: With the new ZDPRINT installation and run-time options, users can choose to have summed (totalled) positive zoned decimal fields converted to printable numbers.

Online Message Explanations Support: New Online Message Explanations (OME) allow users to request an explanation of a DFSORT/VSE message. The message explanation is displayed on the console.

FRONT_2.1.1.3 Additional Enhancements

The following additional enhancements are supported:

- The IBM-supplied default has been changed from STXIT=YES to STXIT=MIN. The STXIT=MIN installation option and the MINSTXIT run-time option allow users to specify that DFSORT/VSE should use its STXIT routine for abend recovery processing, **not** restoring its STXIT every time control is returned from a user exit routine. Unlike STXIT=YES (or STXIT), STXIT=MIN (or MINSTXIT) does not degrade performance when COBOL or PL/I programs invoke DFSORT/VSE and use E15/E35 user exit routines to process records.
- The DIAGINF installation option provides a new DFSORT/VSE capability that allows users to request diagnostic information (diagnostic messages and a dump), regardless of the options in effect at run time.
- The new NRECOUT installation and run-time option allows users to specify the action DFSORT/VSE should perform

when it does not write any records to the output file. This gives users control over the action (continue or terminate), type of message (informational or error), and return code (0,4 or 16) when no records are written to the output file.

1.0 Part 1. Installation and Customization

DFSORT/VSE Installation and Customization is required for system programmers installing DFSORT/VSE. Besides providing information on storage amounts, prerequisite products, and so on, this book provides all of the information necessary for you to customize DFSORT/VSE for the needs of your site.

Subtopics:

1.1 Chapter 1. Planning for Installation

This chapter helps you to plan for the installation of DFSORT/VSE. It describes what you receive with DFSORT/VSE and the required systems and hardware.

Subtopics:

1.1.1 What You Receive with DFSORT/VSE

When you order DFSORT/VSE, you will receive the following:

- **Documentation**
  - *DFSORT/VSE Program Directory*
  - *DFSORT/VSE Licensed Program Specifications*
  - *DFSORT/VSE Application Programming Guide*

- *DFSORT/VSE General Information*
- *DFSORT/VSE Reference Summary*
- *DFSORT/VSE Messages, Codes and Diagnosis Guide*
- *DFSORT/VSE Installation and Tuning Guide*
- *Getting Started with DFSORT/VSE*

- **Distribution Media**

  - An unlabeled 9-track magnetic tape written at 6250 bpi, or
  - A 3480 cartridge, or
  - A 9346 cartridge, or
  - A 4mm Digital Audio Tape (DAT) cartridge

When you order DFSORT/VSE separately, DFSORT/VSE is delivered on a non-stacked tape. When you order DFSORT/VSE as an optional program with VSE/ESA, DFSORT/VSE is delivered on a stacked tape.

Figure 1 shows the layout of the non-stacked distribution tape.

Figure 1. Layout of the DFSORT/VSE Distribution Tape

File Number	Content
1	header file
2	DFSORT/VSE history file
3	DFSORT/VSE library file which contains phases, relocatable object modules, and source
4	null file (tapemark)
5	EOB (end of BACKUP information)
6	null file (tapemark)

If you order DFSORT/VSE as an optional program, refer to *VSE/ESA Installation* for information on using the stacked tape.

1.1.2 What You Need to Install DFSORT/VSE

Before you install the product, you must review certain requirements and considerations.

Subtopics:

1.1.2.1 System Requirements

DFSORT/VSE can be installed under all currently supported versions of the following operating systems:

- VSE/ESA Version 1.3 and above
- VSE/ESA Version 2

| **Note:** We recommend that you use VSE/ESA Version 1 Release 4 as the
| minimum level of operating system. Service was discontinued for CICS/VSE
| Version 2 Release 2, a major component of VSE/ESA Version 1 Release 3, in
| November, 1997. Service will be discontinued for VSE/ESA Version 1
| Release 3 in December, 1998.

VSE/ESA Version 1.4 and above is required to support locale processing with DFSORT/VSE.

You can install DFSORT/VSE any time after your operating system has been generated.

To install DFSORT/VSE, you must use the Maintain System History Program (MSHP).

1.1.2.2 Prerequisite Program Products

If you require locale processing, then you must install one of the following:

- VSE/ESA 1.4 through 2.1 and the Language Environment for VSE/ESA 1.4.0 (with the C-language run-time support installed), or
- VSE/ESA 2.2 (with the C-language run-time support installed from the VSE/ESA Extended Base Products tape), or

- VSE/ESA 2.3 or subsequent release.

1.1.2.3 Machine Requirements

You can install and run DFSORT/VSE on any processor that supports the VSE/ESA releases listed above.

1.1.2.4 Required Storage Devices

To *install* DFSORT/VSE, you need:

- At least one tape or cartridge device for loading the DFSORT/VSE distribution tape.
- Direct access storage for the DFSORT/VSE library. You can choose any of the devices supported by your operating system for program residence.

To *use* DFSORT/VSE, you need:

- Devices supported by SAM or VSAM for input/output
- Direct access devices supported by SAM or VSAM for work storage

---

1.1.2.5 Storage Requirements

| The minimum available partition program area requirement for DFSORT/VSE is
| 32 KB. DFSORT/VSE requires a minimum of 370 KB in the Shared Virtual Area
| (SVA). 16 KB is required in the 24-bit SVA and 354 KB may reside in the
| 31-bit SVA. Depending on the application, DFSORT/VSE might require more storage (in addition to any space needed for other programs). Therefore, although DFSORT/VSE requires a minimum of 32 KB, it is strongly recommended that you use more than the minimum amount of storage.

Auxiliary storage space is required for:

- System History file (if not already allocated)
- Library space

The actual storage requirements for library space are shown in Figure 2.

Figure 2. Space Requirements for DFSORT/VSE

Device	Required Space for DFSORT/VSE	Library Blocks
3390	59 tracks	1934
3380	63 tracks	1934
9345	70 tracks	1934
3375	78 tracks	1934
All FBA disks	3888 blocks	1934

1.1.2.6 Considerations Before Installing

By default, DFSORT/VSE is installed into the PRD2.PROD sublibrary. During initial installation, VSE/ESA allocates PRD2.PROD in VSE/VSAM managed space. This VSE/VSAM space is owned by the VSE master catalog. If the master catalog has sufficient space available, the PRD2 library will be extended automatically. In this case you do not need to be concerned about DASD space for the DFSORT/VSE library.

If you have the Sort/Merge 5746-SM2 product or a previous release of DFSORT/VSE installed in the default production sublibrary (PRD2.PROD), and want to install DFSORT/VSE in the default production sublibrary, you will need to make changes in order to keep both products available on your system. DFSORT/VSE normally replaces the previous product when default installations are performed. In order to keep both products available, you need to install DFSORT/VSE in a library other than the default library. "Procedures for Installing DFSORT/VSE" in topic 1.2.1 outlines the steps for installing DFSORT/VSE in a library other than the default library. These steps are also included in the *DFSORT/VSE Program Directory*.

If you decide to install DFSORT/VSE into a sublibrary other than PRD2.PROD, verify that you have enough DASD space for DFSORT/VSE.

1.1.2.7 Installation Verification Job Requirements

DFSORT/VSE provides four sample jobs to verify that installation was successful. These jobs have the following requirements:

- ILUSAMPL requires at least 64 KB of virtual storage. It also requires scratch disk space: 40 tracks on a count-key-data (CKD) device or 200 blocks on a fixed block mode (FBA) device.
- ILUDATA creates sample input files that use about 20 KB bytes of storage in the master catalog space.
- ILUTOOL requires at least 100 KB of virtual storage.
- ILUCULT requires at least 2100 KB of virtual storage.

1.2 Chapter 2. Installing DFSORT/VSE

This chapter provides procedures for installing DFSORT/VSE.

Note: Installing DFSORT/VSE requires the use of Maintain System History Program (MSHP).

Use the information in this chapter together with the installation information in the *DFSORT/VSE Program Directory*.

Subtopics:

1.2.1 Procedures for Installing DFSORT/VSE

We recommend that you follow the instructions in *VSE/ESA Installation* using the VSE/ESA installation dialog to install DFSORT/VSE.

If you do not use the installation dialog, then follow the instructions provided in this chapter to install DFSORT/VSE. Installing DFSORT/VSE includes the following steps:

- Step 1. Read the *DFSORT/VSE Program Directory*
- Step 2. Allocate Space for the Library
- Step 3. Install DFSORT/VSE
- Step 4. Verify the DFSORT/VSE Installation

Subtopics:

1.2.1.1 Step 1. Read the DFSORT/VSE Program Directory

The *DFSORT/VSE Program Directory* contains detailed information about the installation materials and procedures. It also contains additions to this book. Review the *DFSORT/VSE Program Directory*, before proceeding with the installation of DFSORT/VSE.

In those cases where information in this book conflicts with the information in the *DFSORT/VSE Program Directory*, the *DFSORT/VSE Program Directory* supersedes this book.

1.2.1.2 Step 2. Allocate Space for the Library

By default DFSORT/VSE is installed into the PRD2.PROD sublibrary. If you decide to install DFSORT/VSE into a sublibrary other than PRD2.PROD then proceed with this step. If you decide to install DFSORT/VSE in the default sublibrary then proceed with "Step 3. Install DFSORT/VSE."

Decide where to allocate space for the DFSORT/VSE sublibrary. Identify, the disk volume to be used for the library and suitable areas of free space. To do this, list the volume table of contents (VTOC) of the disk or disks to be used. You can use a job similar to the one in Figure 3.

```
// JOB LVTOC    LIST VOLUME TABLE OF CONTENTS     01
// ASSGN SYS004,DISK,TEMP,VOL=volser,SHR          02
// ASSGN SYS005,SYSLST                            03
// EXEC LVTOC                                     04
/&                                                05
```

Figure 3. Job to List Volume Table of Contents

Line Explanation

02

SYS004 must be assigned to the physical address of the disk on which the necessary volume is mounted. Change the variable *volser* in the VOL operand to your volume serial number.

03

SYS005 must be assigned to the device where the VTOC output is to be routed.

The disk space for the DFSORT/VSE library can be allocated as described by the allocation job shown in the sample in Figure 4.

```
// JOB SORTDEF CREATE A LIBRARY FOR DFSORT/VSE              01
// OPTION LOG                                              02
// ASSGN SYS002,DISK,VOL=volser,SHR                        03
// DLBL SORT,'SORT.LIBRARY',99/365,SD                      04
// EXTENT SYS002,volser,,,rtrk,ntrk                        05
// EXEC LIBR                                               06
 DELETE LIB=SORT                                           07
 DEFINE LIB=SORT REPLACE=NO                                08
/*                                                         09
/&                                                         10
```

Figure 4. Job to Allocate the DFSORT/VSE Library Space

Line Explanation

03

The ASSGN job control statement assigns the logical unit name to the device. Change the variable *volser* to your volume serial number.

04

The DLBL job control statement contains file label information. Change the *filename* (*SORT* in the sample) and *file-id* ('*SORT.LIBRARY*' in the sample) to suit your DFSORT/VSE installation.

05

The EXTENT job control statement defines the disk file extents. The variable *ntrk* indicates the number of blocks or tracks required for DFSORT/VSE installation. Change this variable to suit the type of device being used for your installation. For the number of blocks or tracks required, see the information in "Storage Requirements" in topic 1.1.2.5 or the *DFSORT/VSE Program Directory*. The variable *rtrk* represents the start position of the extent. Change the variable *volser* to your volume serial number.

07

The Librarian job step includes a DELETE command so the job can be rerun. This means the following messages will be issued when the job runs for the first time:

```
L101I  LIBRARY SORT DOES NOT EXIST
L027I  ABNORMAL END DURING DELETE COMMAND PROCESSING
L113I  RETURN CODE OF DELETE IS 8
```

These messages may be ignored; the job continues to allocate the library.

08

The DEFINE command is used to create the library. Change the library name (*SORT* in the sample) to your library name.

The sample shown in Figure 4 in topic 1.2.1.2 defines a library in SAM space. For a library in SAM space, it is necessary to add label information for your library to the standard label area (SLA), as shown in Figure 5.

```
// JOB SORTSTD  ADD AN INFORMATION LABEL TO SLA                01
// OPTION STDLABEL=ADD                                         02
// DLBL SORT,'SORT.LIBRARY',99/365,SD                          03
// EXTENT ,volser,,,rtrk,ntrk                                  04
/*                                                             05
/&                                                             06
```

Figure 5. Job to Add an Information Label to Standard Label Area

Line Explanation

02

The STDLABEL=ADD operand of the OPTION job control statement causes label information to be stored permanently for all subsequent jobs in any partition.

03-04

The same label information for your library as in Figure 4 in topic 1.2.1.2.

To provide label information for your library automatically during system startup, modify procedure STDLABUS.PROC through the STDLABUS skeleton. You must include statements similar to those shown in Figure 5, in procedure STDLABUS (except for // JOB and /&). This portion of the installation must be done by a system administrator.

A library always consists of one or more sublibraries. Programs, procedures and other data are stored as members in the sublibraries. Sublibraries vary in size. They are dynamically extended as required until the space assigned to the library is exhausted.

At least one sublibrary must be defined within the library before data members of any kind can be cataloged.

A sublibrary may contain any or all member types used at an installation. This allows you to store all members that belong to one application in one sublibrary.

To create the sublibrary named SM4 in your library SORT, the job stream shown in Figure 6 in topic 1.2.1.2 is required. It is assumed that the label information is in the standard label area.

```
// JOB SORTDEFS CREATE A SUBLIBRARY FOR DFSORT/VSE       01
// OPTION LOG                                           02
// EXEC LIBR                                            03
 DEFINE SUBLIB=SORT.SM4                                 04
/*                                                      05
/&                                                      06
```

Figure 6. Job to Create the DFSORT/VSE Sublibrary

Line Explanation

04
The DEFINE command is used to create the sublibrary. Change the sublibrary name (SORT.SM4 in the sample) to your sublibrary name.

With the *File and Catalog Management* dialog of the Interactive Interface you can define the library in VSAM-managed space.

1.2.1.3 Step 3. Install DFSORT/VSE

The installation job stream for installing DFSORT/VSE uses the MSHP system history file that already exists as part of the VSE/ESA system. This system history file may already be defined in the system standard labels; if not, make sure that DLBL and EXTENT job control statements for the system history file are included in the job stream.

The default sublibrary for installing DFSORT/VSE is PRD2.PROD. If Sort/Merge 5746-SM2 or a previous release of

DFSORT/VSE is already installed in PRD2.PROD, the default installation of DFSORT/VSE will cause the old product to be replaced. If you need to save the old product, install DFSORT/VSE in a different sublibrary.

The DFSORT/VSE product tape may only contain the DFSORT/VSE product, or it could be a stacked tape containing one or more optional program products. The job shown in Figure 7 will handle both types of tapes.

Create and tailor the installation job shown in Figure 7, mount the distribution tape, and run the installation job.

```
// JOB SORTINST INSTALL DFSORT/VSE                      01
// OPTION LOG                                           02
// ASSGN SYS006,181                                     03
// MTC REW,SYS006                                       04
// EXEC MSHP,PARM='PIDSTACKED'                          05
 INSTALL PRODUCT FROMTAPE ID='DFSORT/VSE.3.4.0' -       06
    PROD INTO=PRD2.PROD                                 07
/*                                                      08
// EXEC LIBR                                            09
 LISTDIR SUBLIB=PRD2.PROD  -                            10
     OUTPUT=NORMAL -                                    11
     UNIT=SYSLST                                        12
/*                                                      13
// MTC  RUN,SYS006                                      14
/&                                                      15
```

Figure 7. Job to Install DFSORT/VSE

Line Explanation

03

The ASSGN job control statement assigns logical unit SYS006 as the distribution tape unit. Replace '181' with the address of your tape unit. Alternatively, you can use the generic tape assignment

```
// ASSGN SYS006,TAPE
```

05-07

The MSHP statements to install DFSORT/VSE into a sublibrary identified on the INTO operand of the INSTALL statement. If you are installing DFSORT/VSE into a sublibrary other than PRD2.PROD, then the name of the sublibrary in the INTO operand must be changed to reflect your chosen sublibrary. For more information about the install options, see *VSE/ESA System Control Statements*.

09-12

This step lists the directory entries of the library where DFSORT/VSE was installed. The directory list can be used to check that all DFSORT/VSE phases have been installed. Remove this step if a directory list is not required. If you are installing DFSORT/VSE into a sublibrary other than the installation default (PRD2.PROD), then the SUBLIB operand

(line 10) must be changed to reflect your chosen sublibrary. Entries for DFSORT/VSE have a three character prefix of ILU to distinguish them from other products. Four exceptions to this naming convention are ICETOOL.OBJ, ICETOOL.PHASE, $SVASORT.PHASE, and SORT.PHASE.

There is no DLBL job control statement for the system history file; you would typically have a permanent system standard label for this, with IJSYSHF as the file name. (IJSYSHF is the default file name that MSHP looks for in a label statement.)

If this install job has to be rerun, remember to restore the system history file (which should have been backed up before you ran this install job). Then rerun the library allocation step, if necessary.

1.2.1.4 Step 4. Verify the DFSORT/VSE Installation

To verify that DFSORT/VSE is correctly installed, run the following samples:

- ILUSAMPL, to generate a job to invoke DFSORT/VSE directly. ILUSAMPL is included in the DFSORT/VSE installation sublibrary in macro format with the name ILUSAMPL.A. (See "Tailor and Execute the ILUSAMPL Sample" in topic 1.2.1.4.1).
- Job ILUDATA, to invoke DFSORT/VSE from assembler. It generates three sample input files SORT.SAMPIN, SORT.SAMPADD, and SORT.BRANCH. For additional information on the sample files, see *Getting Started with DFSORT/VSE*.
- Job ILUTOOL, to invoke ICETOOL directly. It performs multiple operations, which include listing your installation defaults, copying and sorting files, displaying statistics and printing reports (see "ILUTOOL Verification" in topic 1.2.1.4.3).
- Job ILUCULT, if you have enabled cultural environment support, to verify that data is sorted correctly using LOCALE=DA_DK (see "ILUCULT Verification" in topic 1.2.2).

The ILUDATA, ILUTOOL, and ILUCULT jobs reside in the DFSORT/VSE installation sublibrary with names ILUDATA.Z, ILUTOOL.Z, and ILUCULT.Z, respectively.

Subtopics:

1.2.1.4.1 Tailor and Execute the ILUSAMPL Sample

You have to run the sample program in two stages. In the first stage, you supply the installation-dependent information and assemble the ILUSAMPL macro. The output from this assembly is a job stream. In the second stage, you run the generated job stream. Figure 8 shows an overview of this process.

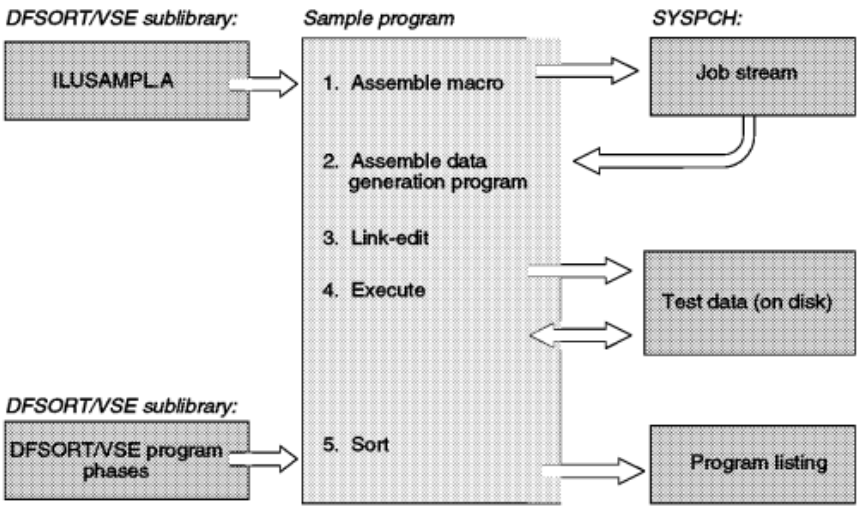


Figure 8. Overview of Steps for Running the Sample Program

To tailor the program you need to assemble the macro, specifying the device type, start of extent information, and the address of the scratch device.

Syntax for the ILUSAMPL macro is follows:

```
 _____________________________________________________________________
|                                                                     |
| >>__ILUSAMPL__DEVTYPE=aaaa__,__EXTENT=bbbbb__,__ADDR=cuu_____________><  |
|                                                                     |
|_____________________________________________________________________|
```

aaaa Device type for the input, output, and work file. Must be one of the following devices: 3390, 3380, 9345, 3375, FBA.

bbbbb Start of the extent for the input, output, and work file.

For CKD: One to five characters giving the sequential number of the track, relative to zero, at which the file is to begin. 40 tracks will be allocated starting at that address.

For FBA: One to ten characters giving the relative data block from the beginning of the disk. 200 blocks will be allocated starting at that address.

cuu Unit address for the input, output, and work file.

A sample of the job used to assemble the macro is shown in Figure 9 in topic 1.2.1.4.1.

```
// JOB  SAMPLPRB
// OPTION  DECK
// EXEC  ASSEMBLY
   COPY  ILUSAMPL
   ILUSAMPL DEVTYPE=3380,EXTENT=100,ADDR=236
   END
/*
/&
```

Figure 9. Sample of the Job to Assemble the ILUSAMPL Macro

This sample specifies an input, output, and work file on a 3380, unit address X'236', starting at track 100.

Output from this macro assembly is a job stream on the SYSPCH system logical unit (Punch Queue).

Copy the job stream you created on SYSPCH to a primary library and submit it for execution using ICCF. The job stream containing the following four steps is then executed:

1. An assembly step for the data generation program
2. A link-editing step for the data generation program
3. An execution step for the data generation program
4. An execution step for sorting the file generated in step 3

An example of the sample job stream and the output listing is shown in Figure 10 in topic 1.2.1.4.1.

```
|   Assembly of file generator
|   // JOB 5746-SM3 SAMPLE PROGRAM
|   // OPTION LINK,NODECK,NOXREF
|   // EXEC ASSEMBLY
|   Link-Edit of file generator
|   // EXEC LNKEDT
|   Generation of sort input file
|   // ASSGN SYS010,X'236'
|   // DLBL OUTFILE,,0
|   // EXTENT SYS010,,,,752,20
|   // EXEC
|   Sort execution
|   // ASSGN SYS015,SYS010
|   // ASSGN SYS020,SYS010
|   // DLBL SORTOUT,,0
|   // EXTENT SYS015,,,,752,20
|   // DLBL SORTIN1,'OUTFILE',0
|   // EXTENT SYS010,,,,752,20
|   // DLBL SORTWK1,,0
|   // EXTENT SYS020,,,,772,20
|   // EXEC SORT,SIZE=64K


|   ILU000I A0 --- CONTROL STATEMENTS/MESSAGES --- DFSORT/VSE 5746-SM3 RELEASE 4.0  DATE 05/29/1998
|               SORT FIELDS=(12,4,A),FORMAT=BI,WORK=1
|               RECORD TYPE=F,LENGTH=15
|               INPFIL BLKSIZE=75
|               OUTFIL BLKSIZE=75
|               OPTION PRINT=ALL,ROUTE=LST,DUMP,DIAG


|   ILU851I C0 27
|   ILU020I C0 STORAGE USED = 65416 BYTES
|   ILU808I C0 MODULE STATUS: SVA
|   ILU021I C0 FIXABLE STORAGE = 0 BYTES
|   ILU843I C0 PH1 2 2
|   ILU843I C0 PH2 0 3
|   ILU843I C0 PH3 2 2
|   ILU842I C0 1380 7538
|   ILU849I C0 39 39 6
|   ILU841I C0 9 10
|   ILU845I C0 15
|   ILU806I C0 PH0: TIME A: 1 SECONDS, TIME B: 0 SECONDS
|   ILU863I F0 5
|   ILU806I F0 PH1: TIME A: 1 SECONDS, TIME B: 0 SECONDS
|   ILU867I F0 PH3 2
|   ILU864I J0 PH3 5 1 0
|   ILU806I J0 PAR: TIME A: 0 SECONDS, TIME B: 0 SECONDS
|   ILU806I J0 PH3: TIME A: 1 SECONDS, TIME B: 0 SECONDS
|   ILU332I J0 WORK SPACE USED: 3 TRACKS ON 3390
|   ILU334I J0 SORT CAPACITY APPROXIMATELY 76000 RECORDS
|   ILU323I J0 SORT  COMPLETE, IN 2000, OUT 2000
|   1S55I  LAST RETURN CODE WAS 0000
|   EOJ 5746-SM3  MAX.RETURN CODE=0000               DATE 05/29/1998,CLOCK 08/31/23,DURATION   00/00/51
```

Figure 10. Sample of a Job Stream and Output Listing

A sample of the log output that results from executing of the sample program is shown in Figure 11.

```
BG 0000 // JOB 5746-SM3 SAMPLE PROGRAM
        DATE 05/29/98,CLOCK 08/30/32

BG 0000 ILU323I J1 5746-SM3  SORT  COMPLETE, IN 2000, OUT 2000
BG 0000 EOJ 5746-SM3  MAX.RETURN CODE=0000
        DATE 05/29/98,CLOCK 08/31/23,DURATION   00/00/51
```

Figure 11. Sample of a Job Log Output

Verify that the sample program ran correctly by:

1. Checking that the condition code for each step is 0.
2. Checking that message ILU323I is present in the output listing and the job log output.

1.2.1.4.2 ILUDATA Verification

Verify that the ILUDATA job ran correctly by:

1. Checking that the condition code for each step is 0. The IDCAMS job step includes DELETE operators so the job can be rerun. This means the DELETE return code will be 8 when the job runs for the first time.
2. Comparing the output of step ILUDATA to Figure 12.

```
|               .
|               .
|               .
|   *   EXEC ILUDATA
|   //  EXEC
|   ILU000I A0 --- CONTROL STATEMENTS/MESSAGES --- DFSORT/VSE 5746-SM3 RELEASE 4.0  DATE 05/29/1998


|              MERGE   FIELDS=COPY
|              RECORD  TYPE=F,LENGTH=173
|              INPFIL  EXIT
|              OPTION  ROUTE=LST,STORAGE=(50K,VIRT),DIAG,DUMP,FILNM=(SAMP1)
|              MODS    PH3=(,,E32)
|   ILU233I D1 OUTFIL BLOCK SIZE = 173 BYTES
|   ILU851I C0 13
|   ILU020I C0 STORAGE USED = 51200 BYTES
|   ILU808I C0 MODULE STATUS: SVA
|   ILU806I C0 PH0: TIME A: 0 SECONDS, TIME B: 0 SECONDS
|   ILU806I J0 MO : TIME A: 3 SECONDS, TIME B: 0 SECONDS
|   ILU321I J0 MERGE COMPLETE, INSERT 20, DELETE 0, IN 0, OUT 20
|   ILU000I A0 --- CONTROL STATEMENTS/MESSAGES --- DFSORT/VSE 5746-SM3 RELEASE 4.0  DATE 05/29/1998


|              MERGE   FIELDS=COPY
|              RECORD  TYPE=F,LENGTH=173
|              INPFIL  EXIT
|              OPTION  ROUTE=LST,STORAGE=(50K,VIRT),DIAG,DUMP,FILNM=(SAMP2)
|              MODS    PH3=(,,E32)
|   ILU233I D1 OUTFIL BLOCK SIZE = 173 BYTES
|   ILU851I C0 13
|   ILU020I C0 STORAGE USED = 51200 BYTES
|   ILU808I C0 MODULE STATUS: SVA
|   ILU806I C0 PH0: TIME A: 0 SECONDS, TIME B: 0 SECONDS
|   ILU806I J0 MO : TIME A: 1 SECONDS, TIME B: 0 SECONDS
|   ILU321I J0 MERGE COMPLETE, INSERT 22, DELETE 0, IN 0, OUT 22
|   ILU000I A0 --- CONTROL STATEMENTS/MESSAGES --- DFSORT/VSE 5746-SM3 RELEASE 4.0  DATE 05/29/1998


|              MERGE   FIELDS=COPY
|              RECORD  TYPE=F,LENGTH=033
|              INPFIL  EXIT
|              OPTION  ROUTE=LST,STORAGE=(50K,VIRT),DIAG,DUMP,FILNM=(SAMP3)
|              MODS    PH3=(,,E32)
|   ILU233I D1 OUTFIL BLOCK SIZE = 33 BYTES
|   ILU851I C0 13
|   ILU020I C0 STORAGE USED = 51200 BYTES
|   ILU808I C0 MODULE STATUS: SVA
|   ILU806I C0 PH0: TIME A: 0 SECONDS, TIME B: 0 SECONDS
|   ILU806I J0 MO : TIME A: 1 SECONDS, TIME B: 0 SECONDS
|   ILU321I J0 MERGE COMPLETE, INSERT 12, DELETE 0, IN 0, OUT 12
|   *$$$$ DFSORT/VSE  SAMPDATA: 3 SAMPLE DATA SETS WERE CREATED
|   1S55I  LAST RETURN CODE WAS 0000
|   EOJ ILUDATA   MAX.RETURN CODE=0000                DATE 05/29/98,CLOCK 06/25/51,DURATION   00/00/47
|               .
|               .
|               .
```

Figure 12. Expected Output from ILUDATA

1.2.1.4.3 ILUTOOL Verification

Verify that the ILUTOOL job ran correctly by:

1. Checking that the condition code for each step is 0.

2. Checking that the output of the DEFAULTS report correctly shows your DFSORT/VSE installation defaults as described in "ILUINST Defaults" in topic 1.4.1.3.

3. Comparing the output of the other reports produced by the job to Figure 13.

```
            .
            .
            .
      * Print profit, employees, and city for each Colorado branch
      DISPLAY FROM(CODASD) LIST(LST) ON(28,6,PD) ON(18,4,ZD) ON(1,15,CH)
(28,6,PD)          (18,4,ZD)          (1,15,CH)
+000000000005200   +000000000000020   Aspen
+000000000007351   +000000000000032   Boulder
+000000000006288   +000000000000033   Denver
-000000000002863   +000000000000022   Fort Collins
+000000000005027   +000000000000019   Vail
            .
            .
            .
      * Print a report for the Colorado branches
      DISPLAY FROM(CODASD) LIST(LST) -
         DATE TITLE('Colorado Branches Report') PAGE -
         HEADER('City') HEADER('Profit') HEADER('Employees') -
         ON(1,15,CH) ON(28,6,PD) ON(18,4,ZD) BLANK -
         TOTAL('Total') AVERAGE('Average') MINIMUM('Lowest')
05/29/98        Colorado Branches Report        - 1 -

City                         Profit          Employees
---------------   ----------------   ----------------
Aspen                          5200                 20
Boulder                        7351                 32
Denver                         6288                 33
Fort Collins                  -2863                 22
Vail                           5027                 19


Total                         21003                126


Average                        4200                 25


Lowest                        -2863                 19
            .
            .
            .
      * Print the count of books in use from each publisher
      OCCUR FROM(BKIN) LIST(LST) BLANK -
```

```
         TITLE('Books from Publishers') DATE(DMY.) -
         HEADER('Publisher') HEADER('Books in use') -
         ON(106,4,CH) ON(VALCNT)
Books from Publishers        05.29.98


Publisher      Books in use
---------   ---------------
COR                       7
FERN                      4
VALD                      5
WETH                      4
              .
              .
              .
```

Figure 13. Expected Reports from ILUTOOL

1.2.2 ILUCULT Verification

Verify that the ILUCULT job ran correctly by:

1. Checking that the condition code for each step is 0.
2. Comparing the DFSORT/VSE output to Figure 14.

Note: Before running the job, make sure that the partition size is at least 2100 KB.

```
              .
              .
              .
// EXEC  ,SIZE=56K

ILU000I A0 --- CONTROL STATEMENTS/MESSAGES --- DFSORT/VSE 5746-SM3 RELEASE 4.0  DATE 05/29/1998

           SORT    FIELDS=(1,1,CH,A),WORK=0
           RECORD  TYPE=F,LENGTH=32
           INPFIL  EXIT
           OPTION  STORAGE=48K,SORTOUT=LST,LOCALE=DA_DK
```

```
           MODS    PH1=(,,E15)
ILU701I B0 LOCALE PROCESSING WAS USED WITH ACTIVE LOCALE DA_DK
ILU233I D1 OUTFIL BLOCK SIZE = 32 BYTES
ILU020I C0 STORAGE USED = 49152 BYTES
ILU012I C0 MODULE STATUS: PARTITION (POSSIBLY PERFORMANCE DEGRADATION)
a
A
b
B
c
C
d
D
e
E
ILU334I J0 SORT CAPACITY APPROXIMATELY 430 RECORDS
ILU321I J0 SORT  COMPLETE, INSERT 10, DELETE 0, IN 0, OUT 10
1S55I  LAST RETURN CODE WAS 0000
EOJ ILUCULT   MAX.RETURN CODE=0000                DATE 05/29/1998, CLOCK 11/12/25, DURATION 00/00/12
             .
             .
             .
```

Figure 14. Expected Output from ILUCULT

1.3 Chapter 3. Planning for Customization

This chapter describes how you can use supplied services and facilities to customize DFSORT/VSE to the needs of your site. Customization is the task of enhancing or extending a program by using built-in facilities. This chapter describes planning considerations for:

- Changing the DFSORT/VSE Installation Defaults
- Modifying Message Text
- | Modifying DFSORT/VSE Online Message Explanations
- Placing DFSORT/VSE Phases in the SVA
- Enabling DFSORT/VSE Cultural Environment Support
- | Enabling Online Message Explanations Support

Subtopics:

1.3.1 Changing DFSORT/VSE Installation Defaults

The IBM-supplied installation defaults provided on the installation tape are given in Chapter 4, "Customizing DFSORT/VSE" in topic 1.4. These defaults are in effect if you install DFSORT/VSE without specifying your own defaults.

Many of the defaults can be changed at any time after installing DFSORT/VSE. You might want to first install DFSORT/VSE with the given defaults, then tailor them to your requirements.

You can change the IBM-supplied defaults by using ILUINST macro operands.

Some of the defaults can be overridden by the application programmer at run time by use of DFSORT/VSE control statements. For full override details, see *Application Programming Guide*.

1.3.2 Modifying Message Text

During installation, you can modify the wording of the DFSORT/VSE message text or translate them into another language.

If you want to issue messages in a language other than English or to change the wording of some messages, you can modify the message text in the ILUMSGS macro. To apply your changes to the ILUMSGS macro, you must reassemble and relink-edit the ILUMSGA module (to modify DFSORT/VSE messages), or the ILUMUTL module (to modify ICETOOL messages), or both.

Note: If you modify the message texts in ILUMSGS, you need to modify the corresponding message texts in the online message file. See "Modifying DFSORT/VSE Online Message Explanations"for more information.

| 1.3.3 Modifying DFSORT/VSE Online Message Explanations

| If you modify the DFSORT/VSE message text, we recommend that you modify
| the corresponding message texts in the DFSORT/VSE online message file.
| You can modify ILUSOME.Z, ILUSOMJ.Z, or ILUSOMG.Z and run the sample job
| in Figure 29 in topic 1.4.6, Figure 30 in topic 1.4.6.1, or Figure 31 in
| topic 1.4.6.2. The sample job loads the DFSORT/VSE online message file
| into the online message file.

| **Note:** Take extra care when you modify the messages in ILUSOMx.Z.
| Unexpected results can occur if you modify the online message file
| incorrectly.

1.3.4 Placing DFSORT/VSE Phases in the SVA

SVA-eligible phases of DFSORT/VSE can be placed into 24-bit and 31-bit Shared Virtual Area (SVA).

The SVA contains reenterable phases, which may be used by multiple tasks or programs concurrently executing in the system.

By sharing the information the following benefits may be gained:

- Reduce virtual storage requirements
- Reduce I/O activity to the product library, or the page data sets, or both
- Reduce elapsed time for applications

If DFSORT/VSE is installed resident, you must ensure that enough storage is available in the SVA. For further details, see "Placing DFSORT/VSE Phases into the SVA" in topic 1.4.4.

1.3.5 Enabling DFSORT/VSE Cultural Environment Support

DFSORT/VSE's cultural environment support allows you to modify DFSORT/VSE's collating behavior according to your cultural environment. Your cultural environment is defined to DFSORT/VSE using the X/Open locale model. The cultural environment is established by selecting the active locale using the LOCALE option at installation- or run-time.

The active locale defines collating rules, which affect the behavior of locale-sensitive functions. In particular, the collating rules of the active locale affect DFSORT/VSE's SORT, MERGE, INCLUDE, and OMIT processing. This provides sorting, merging, inclusion, and omission for single- or multi-byte character data based on defined collating rules that retain the cultural and local characteristics of a language.

For details of locale processing, see *Application Programming Guide* and the discussion of the LOCALE operand in "ILUINST Operands" in topic 1.4.1.4.

To use locale processing, you must enable DFSORT/VSE's cultural environment support. For further details, see "Establishing DFSORT/VSE Cultural Environment Support" in topic 1.4.5.

| 1.3.6 Enabling Online Message Explanations Support

| The file VSE.MESSAGES.ONLINE contains explanations to messages which may
| be issued by the VSE/ESA base system functions. Using the EXPLAIN
| function of the system console, you can request that an explanation of a
| message is displayed on the VSE/ESA console. DFSORT/VSE allows you to add
| the DFSORT/VSE explanations to this file for online display. The
| DFSORT/VSE message explanations that are displayed are the same message
| Explanation, System Action, and Programmer Response as shown in *Messages,
| Codes, and Diagnosis Guide*.

| To use DFSORT/VSE's message explanations, you must enable DFSORT/VSE's
| OME. For further details, see "Establishing DFSORT/VSE Online Message
| Explanations" in topic 1.4.6.

1.4 Chapter 4. Customizing DFSORT/VSE

This chapter provides the information and procedures needed to customize DFSORT/VSE under the VSE/ESA operating system.

Customizing DFSORT/VSE includes the following steps:

- Step 1. Reviewing DFSORT/VSE Installation Defaults
- Step 2. Changing DFSORT/VSE Installation Defaults
- Step 3. Modifying ILUMSGS Macro Message Text
- Step 4. Placing DFSORT/VSE Phases into the SVA
- Step 5. Establishing DFSORT/VSE Cultural Environment Support
- | Step 6. Establishing DFSORT/VSE Online Message Explanations

Subtopics:

1.4.1 Reviewing DFSORT/VSE Installation Defaults

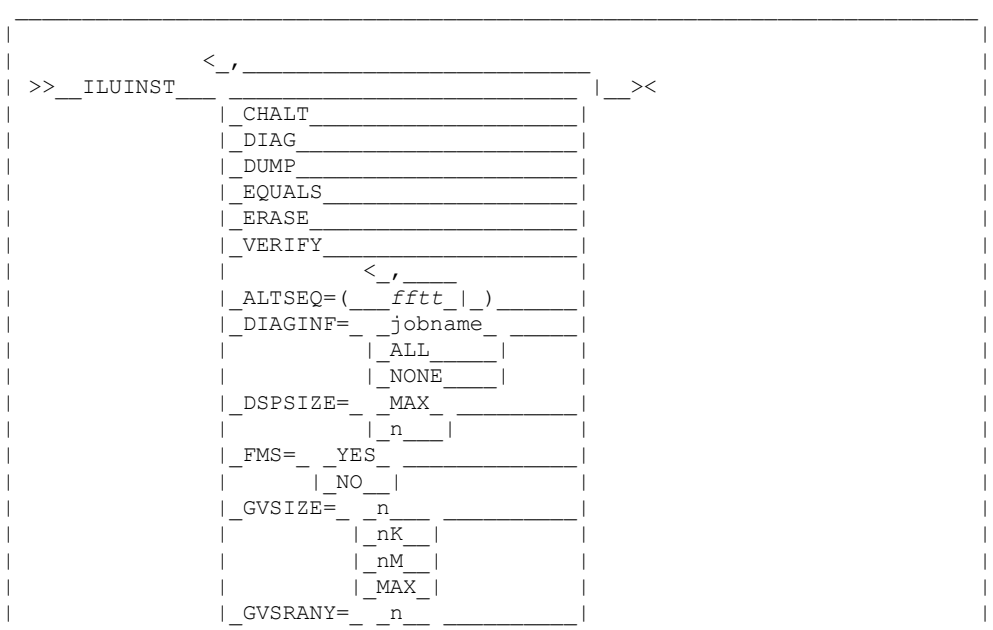
This section explains how to code the ILUINST macro to change IBM-supplied defaults and shows a sample of coding the macro. This section also lists the IBM-supplied defaults and explains all of the ILUINST parameters in detail.

The rules for coding the ILUINST macro are the same as those for coding IBM assembler language macro statements. ILUINST must be preceded and followed by at least one blank space, and operands must be separated by commas. See "Notational Conventions" in topic PREFACE.6 for information about notational conventions used in this book. For more complete rules, refer to your assembler language book.

Subtopics:

1.4.1.1 ILUINST Syntax Diagram

Figure 15 in topic 1.4.1.1 is the syntax diagram for the ILUINST macro. ILUINST has positional and keyword operands. All operands are optional-one or more of the operands can be used, but the individual operands cannot be repeated.



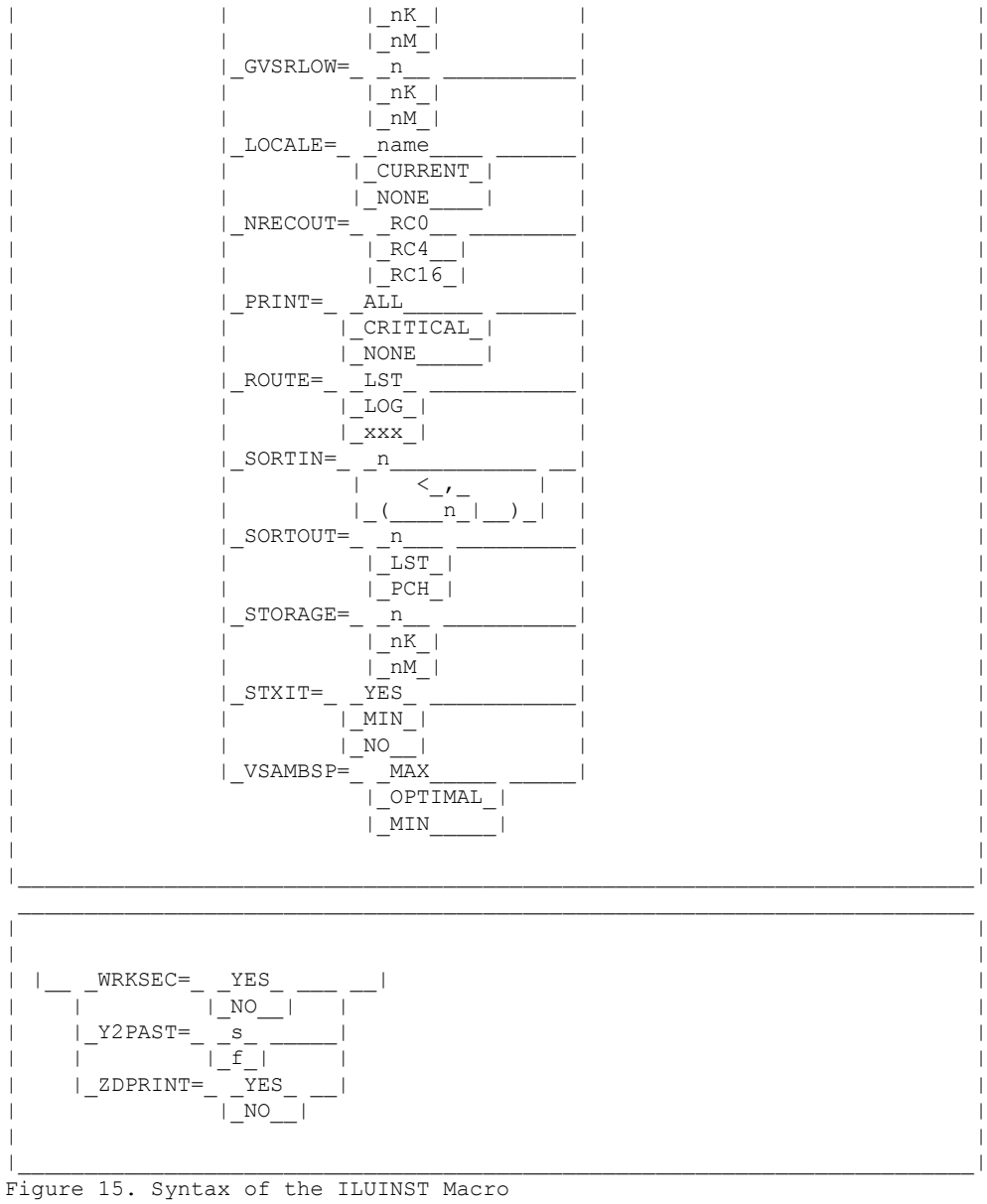


Figure 15. Syntax of the ILUINST Macro

1.4.1.2 ILUINST Macro Sample

Figure 16 shows you a sample of coding the ILUINST macro to change the IBM-supplied defaults.

```
column
1             16                          72 73    80
```

```
ILUINST     EQUALS,                    * 00010000          01
            ERASE,                     * 00010100          02
            GVSIZE=MAX,                * 00010200          03
            GVSRANY=48K,               * 00010300          04
            GVSRLOW=96K,               * 00010400          05
            PRINT=CRITICAL,            * 00010500          06
            STORAGE=512K                 00010600          07
```

Figure 16. Coding the ILUINST Macro

Line Explanation

01

DFSORT/VSE preserves the order of equally collating records from input to output.

02

DFSORT/VSE erases work files if they have been used.

03

DFSORT/VSE dynamically determines the maximum amount of partition GETVIS area to be used for getvis sorting.

04

DFSORT/VSE reserves 49152 bytes (48 KB) of 31-bit GETVIS area for the operating system and user application when the GVSIZE=MAX option is in effect.

05

DFSORT/VSE reserves 98304 bytes (96 KB) of 24-bit GETVIS area for the operating system and user application when the GVSIZE=MAX option is in effect.

06

DFSORT/VSE only prints critical error messages.

07

DFSORT/VSE makes use of all available partition program area up to a limit of 524288 bytes (512 KB).

No other standard defaults are overridden in this sample. See Figure 18 in topic 1.4.1.3 in "ILUINST Defaults" for the IBM-supplied defaults for all of the ILUINST parameters.

1.4.1.3 ILUINST Defaults

You can use the job shown in Figure 17 to list the ILUINST installation defaults currently in effect at your site. Figure 18 in topic 1.4.1.3 lists the IBM-supplied defaults for all of the ILUINST parameters. The IBM-supplied defaults are used unless you change them.

```
// JOB DEFAULTS   LISTING INSTALLATION DEFAULTS
// LIBDEF *,SEARCH=PRD2.PROD
// EXEC ICETOOL,SIZE=100K
DEFAULTS LIST(LST)
/*
/&
```

Figure 17. ICETOOL DEFAULTS Job

The values for each parameter are shown as they are set in the ILUINST installation defaults module loaded from the SEARCH sublibrary. If DFSORT/VSE has been installed in a sublibrary other than the installation default library, change PRD2.PROD to your sublibrary.

```
DFSORT/VSE INSTALLATION (ILUINST) DEFAULTS          - 1 -

 * ONLY SHOWN IF DIFFERENT FROM THE SPECIFIED INSTALLATION DEFAULT

PARAMETER      INSTALLATION DEFAULT                IBM-SUPPLIED DEFAULT *
----------     --------------------------------    --------------------------------
CHALT          NOCHALT
DIAG           NODIAG
DUMP           NODUMP
EQUALS         NOEQUALS
ERASE          NOERASE
VERIFY         NOVERIFY
ALTSEQ         *** SEE BELOW ***
DIAGINF        NONE
DSPSIZE        0
FMS            NO
GVSIZE         0
```

```
    GVSRANY         32K
    GVSRLOW         64K
    LOCALE          NONE
|   NRECOUT         RC0
    PRINT           ALL
    ROUTE           LST
    SORTIN          (2,3,4,5,6,7,8,9,10)
    SORTOUT         1
    STORAGE         0
|   STXIT           MIN
|   VSAMBSP         OPTIMAL
    WRKSEC          YES
    Y2PAST          80
|   ZDPRINT         NO


    ILUINST ALTSEQ TABLE:  SAME AS IBM-SUPPLIED ALTSEQ TABLE BELOW


    IBM-SUPPLIED ALTSEQ TABLE (IN HEXADECIMAL):
           00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
         -------------------------------------------------
      00 | 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F |
      10 | 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F |
      20 | 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F |
      30 | 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F |
      40 | 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F |
      50 | 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F |
      60 | 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F |
      70 | 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F |
      80 | 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F |
      90 | 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F |
      A0 | A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF |
      B0 | B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF |
      C0 | C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF |
      D0 | D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF |
      E0 | E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF |
      F0 | F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF |
         -------------------------------------------------
```

Figure 18. Output from the ICETOOL DEFAULTS Job Showing IBM-supplied Defaults

If the installation default value for a parameter is different from the IBM-supplied default value, the IBM-supplied default value is shown in the next column. Figure 19 in topic 1.4.1.3 is a sample of what the output from the ICETOOL DEFAULTS job looks like if some of the defaults have been changed from the IBM-supplied defaults.

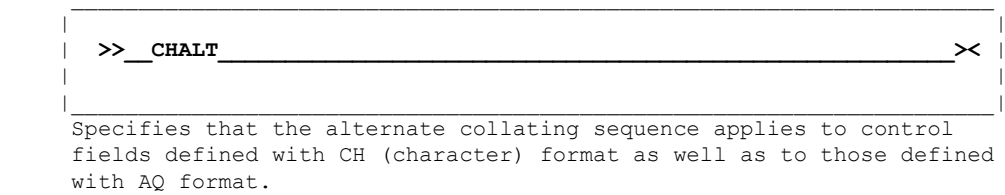
```
    DFSORT/VSE INSTALLATION (ILUINST) DEFAULTS          - 1 -


     * ONLY SHOWN IF DIFFERENT FROM THE SPECIFIED INSTALLATION DEFAULT

    PARAMETER      INSTALLATION DEFAULT               IBM-SUPPLIED DEFAULT *
    ----------     --------------------------------   --------------------------------
```

```
    CHALT          NOCHALT
    DIAG           NODIAG
    DUMP           NODUMP
    EQUALS         EQUALS                                NOEQUALS
    ERASE          ERASE                                 NOERASE
    VERIFY         NOVERIFY
    ALTSEQ         *** SEE BELOW ***
|   DIAGINF        NONE
    DSPSIZE        0
    FMS            NO
    GVSIZE         MAX                                   0
       .
       .
       .
```

Figure 19. Output from the ICETOOL DEFAULTS Job Showing that Some of the Defaults Have Been Changed

1.4.1.4 ILUINST Operands

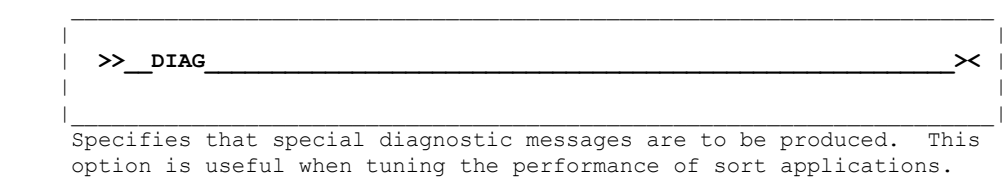
CHALT



Specifies that the alternate collating sequence applies to control fields defined with CH (character) format as well as to those defined with AQ format.

Default: Alternate collating sequence does not apply to character format fields.

DIAG



Specifies that special diagnostic messages are to be produced. This option is useful when tuning the performance of sort applications.

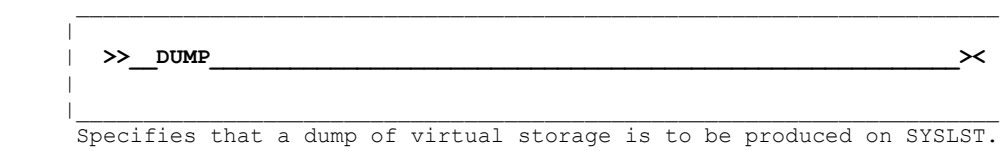
Default: Diagnostic messages are not produced.

Notes:

| 1. If DIAG is specified, PRINT=NONE or PRINT=CRITICAL is ignored.

| 2. If DIAGINF is in effect for all jobs or for a specific job, DIAG
| is forced, regardless of the options in effect at run time.

DUMP



Specifies that a dump of virtual storage is to be produced on SYSLST.

| When DFSORT/VSE's STXIT is in effect, a dump will be produced for any

| DFSORT/VSE abnormal termination. When DFSORT/VSE's STXIT is not in

| effect, a dump will be produced for DFSORT/VSE detected errors.

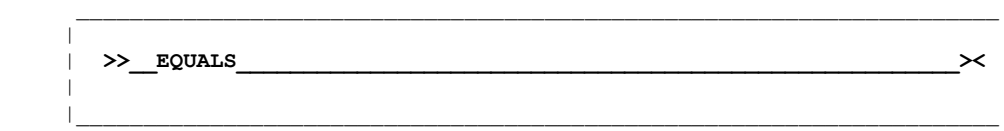
Default: A dump is not produced if DFSORT/VSE terminates abnormally.

Notes:

| 1. If PRINT=NONE is specified, DUMP is ignored.

| 2. If DIAGINF is in effect for all jobs or for a specific job, DUMP
| is forced, regardless of the options in effect at run time.

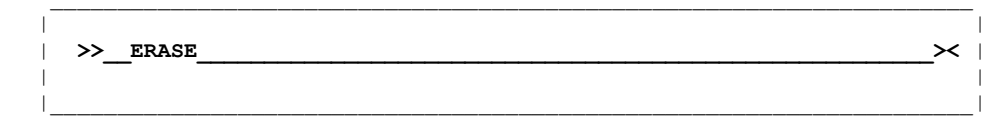
EQUALS



Specifies that the original sequence of records that collate identically for sort or merge applications should be preserved from input to output.

Default: The order need not be preserved.

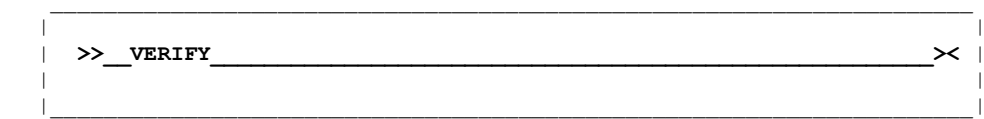
ERASE



Specifies that the work files must be erased if they have been used. For CKD devices, this is done by means of an end-of-file record written on every used track of each work file. Used parts of FBA work files are filled with zeros.

Default: Work files are not erased.

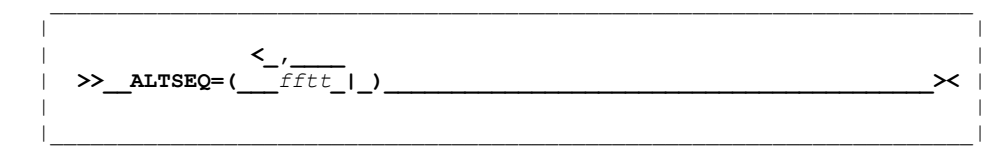
VERIFY



Specifies that, when a direct-access device is used to store the output file, each block is checked to ensure that it was written correctly. VERIFY is ignored for VSAM output files.

Default: Output blocks are not verified.

ALTSEQ



Specifies a modification to the standard EBCDIC collating sequence, to apply when the user defines a control field with the alternate sequence format (AQ). The data itself does not change, but only the order in which it is collated.

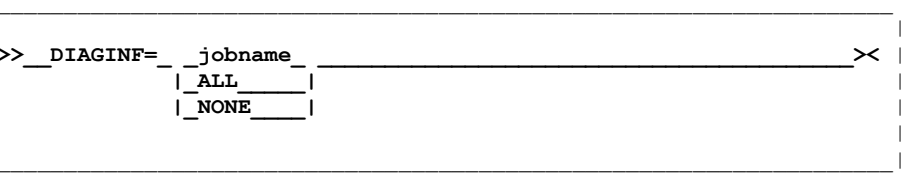
fftt specifies the characters whose positions are to be changed. You can change the positions of up to 50 characters.

ff specifies, in hexadecimal, the character whose position is to be changed.

tt specifies, in hexadecimal, the new position in the collating sequence the character is to occupy.

Default: The standard EBCDIC collating sequence is used.

| **DIAGINF**



| Specifies whether DFSORT/VSE should always produce diagnostic
| information for all jobs or for a specific job. Diagnostic
| information consists of diagnostic messages, and a dump if an abend
| occurs.

| **jobname** specifies that DFSORT/VSE should always produce diagnostic
| information for the specified job, regardless of the options
| in effect for that job. The following options will be used
| at run time for the specified job:

```
|                 DIAG, DUMP, PRINT=ALL, ROUTE=LST
```

| These options cause all diagnostic messages and a dump (if
| an abend occurs) to be produced and routed to SYSLST for the
| specified job. For complete information, see the
| descriptions of DIAG, DUMP, PRINT and ROUTE.

| Up to 8 characters can be specified for the *jobname* value.

| **ALL** specifies that DFSORT/VSE should always produce diagnostic
| information for all jobs, regardless of the options in

| effect for a particular job. The following options will be
| used at run time for all jobs:

```
|                 DIAG, DUMP, PRINT=ALL, ROUTE=LST
```

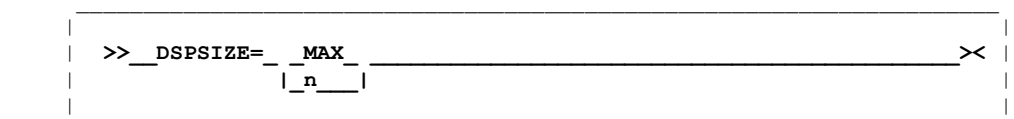
| These options cause all diagnostic messages and a dump (if
| an abend occurs) to be produced and routed to SYSLST for all
| jobs. For complete information, see the descriptions of
| DIAG, DUMP, PRINT and ROUTE.

| **NONE** specifies that DFSORT/VSE should produce diagnostic
| information according to the options in effect at run time.

| *Default:* DIAGINF=NONE.

| **Note:** DIAGINF=jobname and DIAGINF=ALL should be used only when
| necessary to obtain diagnostic information.

DSPSIZE



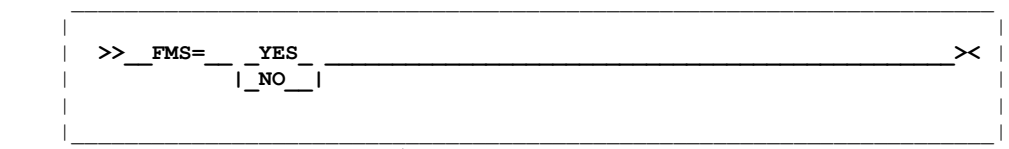
Specifies the maximum amount of data space to be used for dataspace sorting.

MAX specifies that DFSORT/VSE dynamically determines the maximum amount of data space to be used for dataspace sorting. In this case, DFSORT/VSE bases its data space usage on the information from the SYSDEF job control statement for creation of a single data space, the amount of processor storage, and the paging activity of the system.

n specifies the amount, in megabytes, of data space to be used for dataspace sorting. n must be a value between 0 and 2048. The specified value of n will be decreased by DFSORT/VSE, if required to avoid processor storage overcommitment. If n is 0 or the requested storage is not available, dataspace sorting will not be used.

Default: DSPSIZE=0.

FMS



Specifies whether DFSORT/VSE will attempt to interact with the File Management System installed at your site.

YES specifies that DFSORT/VSE will attempt to take advantage of the benefits provided by the File Management System installed at your site, such as:

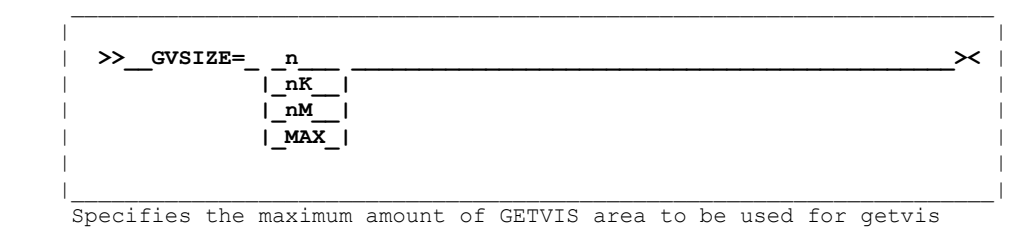
- Dynamic logical and physical device assignment
- Dynamic primary and secondary extent allocation

NO specifies that DFSORT/VSE will not attempt to interact with a File Management System.

Default: FMS=NO.

Note: If you use CA-DYNAM as your file management system, you may need to set CA-DYNAM's "Interface With CA-SORT Enabled" option to "YES".

GVSIZE



Specifies the maximum amount of GETVIS area to be used for getvis sorting.

n specifies that n bytes of GETVIS area are to be used for getvis sorting.

nK specifies that n times 1024 bytes (1 KB) of GETVIS area are to be used for getvis sorting.

nM specifies that n times 1048576 bytes (1 MB) of GETVIS area are to be used for getvis sorting.

MAX specifies that DFSORT/VSE dynamically determines the maximum amount of GETVIS area to be used for getvis sorting.

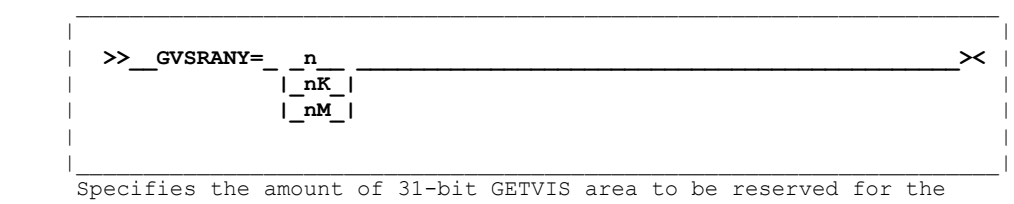
In this case, DFSORT/VSE bases its GETVIS area usage on the size of the available GETVIS area, the amount of reserved 24-bit GETVIS area and 31-bit GETVIS area for the operating system and user applications, the amount of processor storage, and the paging activity of the system.

Default: GVSIZE=0.

Notes:

1. The specified amount of GETVIS area must be 0 or a value between 32K and 2047M.
2. The specified value of n will be decreased by DFSORT/VSE, if required, to avoid processor storage overcommitment.
3. If n is 0, getvis sorting will not be used.

GVSRANY



Specifies the amount of 31-bit GETVIS area to be reserved for the operating system and user applications when the GVSIZE=MAX option is in effect.

n specifies that n bytes of GETVIS area are to be reserved.

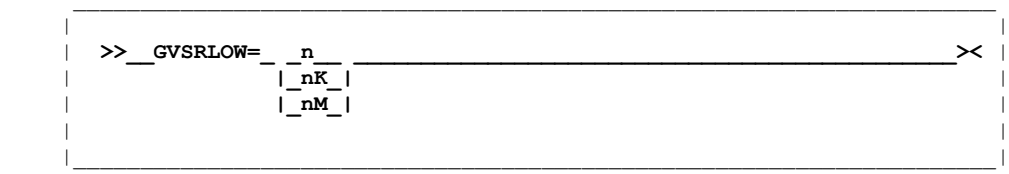
nK specifies that n times 1024 bytes (1 KB) of GETVIS area are to be reserved.

nM specifies that n times 1048576 bytes (1 MB) of GETVIS area are to be reserved.

Default: GVSRANY=32K.

Note: The specified amount of 31-bit GETVIS area must be a value between 32K and 2047M.

GVSRLOW



Specifies the amount of 24-bit GETVIS area to be reserved for the operating system and user applications when the GVSIZE=MAX option is in effect.

n specifies that n bytes of GETVIS area are to be reserved.

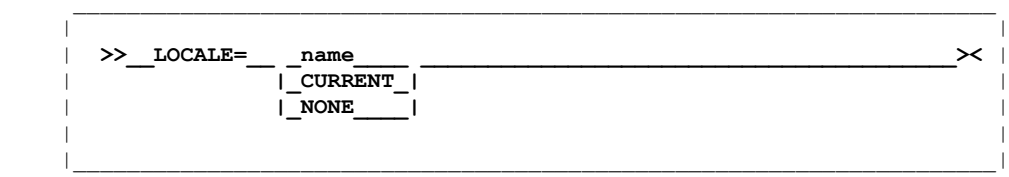
nK specifies that n times 1024 bytes (1 KB) of GETVIS area are to be reserved.

nM specifies that n times 1048576 bytes (1 MB) of GETVIS area are to be reserved.

Default: GVSRLOW=64K.

Note: The specified amount of 24-bit GETVIS area must be a value between 32K and 16M.

LOCALE



Specifies whether locale processing is to be used and if so, designates the active locale.

DFSORT/VSE's collating behavior can be modified according to your cultural environment. Your cultural environment is defined to DFSORT/VSE using the X/Open locale model. A locale is a collection of data grouped into categories that describe the information about your cultural environment.

The collate category of a locale is a collection of sequence declarations that define the relative order between collating elements (single-character and multi-character collating elements). The sequence declarations define the collating rules.

If locale processing is to be used, the active locale will affect the behavior of DFSORT/VSE's SORT, MERGE, INCLUDE, and OMIT functions. For SORT and MERGE, the active locale will only be used to process character (CH) control fields. For INCLUDE and OMIT, the active locale will only be used to process character (CH) compare fields, and character and hexadecimal constants compared to character (CH) compare fields.

name specifies that locale processing is to be used and designates the name of the locale to be made active during DFSORT/VSE processing.

The locales are designated using a descriptive name. For example, to set the active locale to represent the French language and the cultural conventions of Canada, specify LOCALE=FR_CA. Up to 32 characters can be specified for the descriptive locale name. The locale names themselves are not case-sensitive. See *LE/VSE Library* for a complete description of the naming conventions for locales.

IBM-supplied and user defined locales can be used. Examples of some IBM-supplied locales can be found in Appendix B, "Locales Supplied with C Run-Time Library" in topic B.0.

The state of the active locale prior to DFSORT/VSE being entered will be restored when DFSORT/VSE completes processing.

CURRENT specifies that locale processing is to be used, and the current locale active when DFSORT/VSE is entered will remain the active locale during DFSORT/VSE processing.

NONE specifies that locale processing is not to be used. For CH fields, DFSORT/VSE will use the default or alternate collation sequence as specified. DFSORT/VSE will use binary encoding for collating and comparing.

Default: LOCALE=NONE.

Notes:

1. LOCALE=name or LOCALE=CURRENT can cause DFSORT/VSE to run differently than it does with the IBM-supplied default LOCALE=NONE. In particular, locale processing may require more resources (for example, storage and work space), handle fewer control fields, and degrade performance. As a result, you may only want to set LOCALE=name or LOCALE=CURRENT on an individual application, rather than installation-wide, basis.

2. If you require locale processing, then you must install one of the following:

VSE/ESA 1.4 through 2.1 and the Language Environment for VSE/ESA 1.4.0 (with the C-language run-time support installed),

or

VSE/ESA 2.2 (with the C-language run-time support installed from the VSE/ESA Extended Base Products tape) or subsequent release.

When locale processing is in effect, you need additional partition GETVIS area for dynamically loadable LE/VSE phases. The size of that area depends on how LE/VSE was installed at your site, but 2 MB should be enough in most cases. The

storage needed by LE/VSE will be reduced if LE/VSE phases CEEBINIT, CEEPLPKA, and CEEV003 have been placed into the SVA. See *LE/VSE Library* for more information.

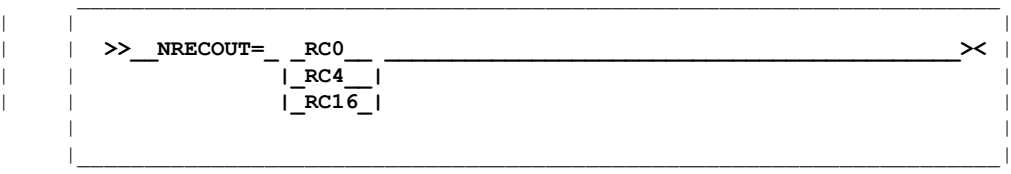
3. To use an IBM-supplied locale, DFSORT/VSE must have access to the sublibrary containing the LE/VSE dynamically loadable routines and compiled locale modules. For example, this sublibrary might be called PRD2.SCEEBASE. If you are not sure of where the compiled locale modules are installed at your site, contact your system administrator. To use a user-defined locale, DFSORT/VSE must have access to the sublibrary containing the locale.

4. If locale processing is requested:

- CHALT must not be used
- INREC must not be used

5. Locale processing for SORT, MERGE, INCLUDE, and OMIT functions can improve performance relative to applications which perform pre- or postprocessing of data to produce the desired collating results. However, locale processing should only be used when required since it can show degraded performance relative to collation using character encoding values.

| **NRECOUT**



| Specifies the actions to be taken by DFSORT/VSE when it does not write
| any records to the output file.

| **RC0** specifies that DFSORT/VSE should issue message ILU400I, set a
| return code of 0 and continue processing.

| **RC4** specifies that DFSORT/VSE should issue message ILU400I, set a
| return code of 4 and continue processing.

| **RC16** specifies that DFSORT/VSE should issue message ILU401A, set a
| return code of 16 and terminate processing.

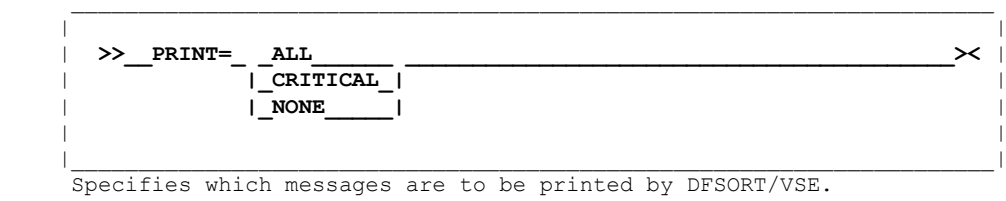
| *Default:* NRECOUT=RC0.

| **Notes:**

| 1. The return code of 0 or 4 set when DFSORT/VSE does not write any
| records to the output file can be overridden by a higher return
| code set for some other reason.

| 2. If OUTFIL EXIT is specified, NRECOUT is ignored.

PRINT



ALL specifies that all standard DFSORT/VSE messages are to be printed, including error and end of job messages, various size calculations messages, and other informational messages.

NONE specifies that DFSORT/VSE messages are not to be printed.

CRITICAL specifies that only critical error messages are to be printed; that is, error messages signaling conditions that can cause DFSORT/VSE to terminate.

Default: PRINT=ALL.

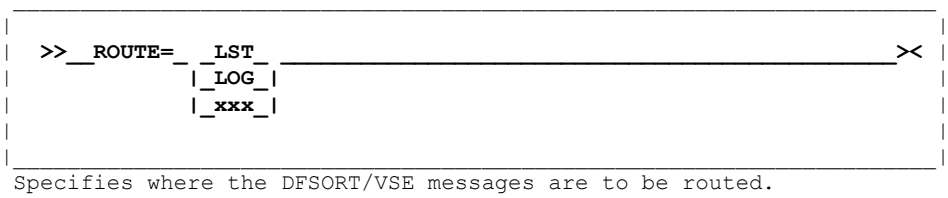
Notes:

1. If PRINT=NONE is specified, DUMP is ignored.

2. If DIAG is specified, PRINT=NONE or PRINT=CRITICAL is ignored.

| 3. If DIAGINF is in effect for all jobs or for a specific job,
| PRINT=ALL is forced, regardless of the options in effect at run
| time.

ROUTE



Specifies where the DFSORT/VSE messages are to be routed.

LST specifies that all DFSORT/VSE messages are to be routed to SYSLST, and critical messages to the system console.

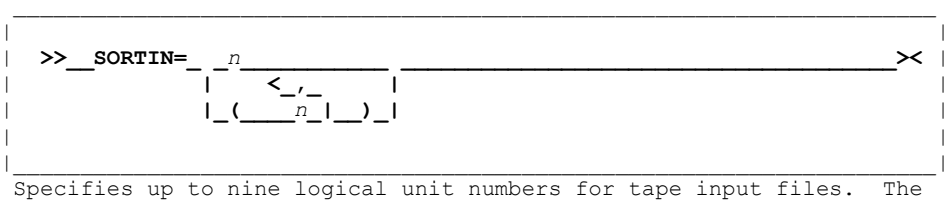
LOG specifies that all DFSORT/VSE messages are to be routed to the system console.

xxx specifies that DFSORT/VSE messages are to be routed to SYSxxx, where xxx can be any valid logical unit number between 000 and 221. Critical messages will also be routed to SYSLOG but the system dump (if any) is produced on SYSLST. ROUTE=xxx is only in effect when DFSORT/VSE is invoked from another program. When DFSORT/VSE is invoked directly and ROUTE=xxx is specified, ROUTE=LST is used.

Default: ROUTE=LST.

| **Note:** If DIAGINF is in effect for all jobs or for a specific job,
| ROUTE=LST is forced, regardless of the options in effect at run time.

SORTIN

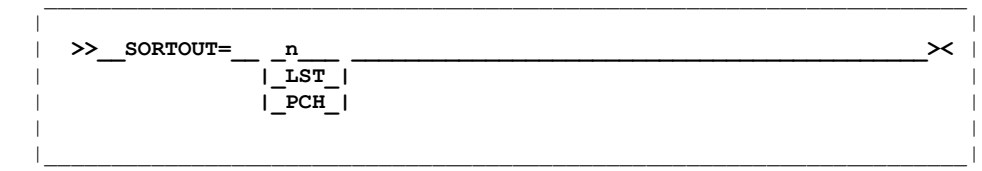


Specifies up to nine logical unit numbers for tape input files. The

```
value for a logical unit number can be any number from 1 through 221,
or a comma (for the default logical unit number).
```

Default: (002,...,010).

SORTOUT



Specifies the output device.

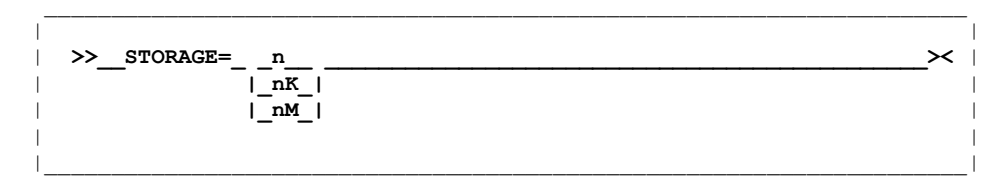
n specifies the logical unit number of the output device. The value for the logical unit number can be any number from 1 through 221. The logical unit number is only needed for tape, printer, and punch devices.

LST specifies that output is to be written to SYSLST.

PCH specifies that output is to be written to SYSPCH.

Default: 1.

STORAGE



Specifies the amount of virtual storage of the partition program area to be used, if available, by DFSORT/VSE.

n specifies that n bytes are to be used.

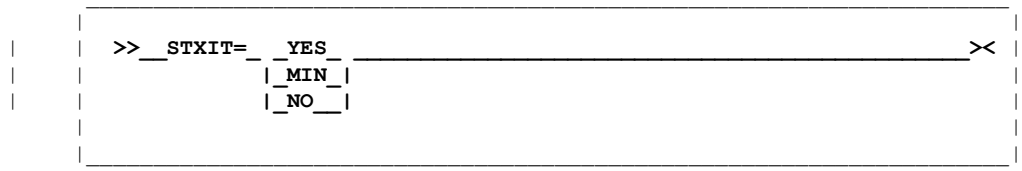
nK specifies that n times 1024 bytes (1 KB) are to be used.

nM specifies that n times 1048576 bytes (1 MB) are to be used.

Default: STORAGE=0.

Note: The amount of virtual storage must be a value between 32K and 16M.

STXIT



Specifies whether DFSORT/VSE should use its STXIT routine for abend recovery processing.

YES specifies that DFSORT/VSE should use its STXIT routine for abend recovery processing, restoring its STXIT every time control is returned from a user exit routine.

DFSORT/VSE will save the caller's STXIT, establish its STXIT, restore its STXIT each time a user exit routine (if any) returns control to DFSORT/VSE, and restore the caller's STXIT before returning control to the caller.

If an abend occurs, DFSORT/VSE will return control to the caller with a return code of 16 (unsuccessful completion) at the end of its abend recovery processing.

Performance can be degraded when COBOL or PL/I programs invoke DFSORT/VSE and use E15 or E35 user exit routines to process records.

MIN specifies that DFSORT/VSE should use its STXIT routine for abend recovery processing and will **not** restore its STXIT every time control is returned from a user exit routine.

DFSORT/VSE will save the caller's STXIT, establish its STXIT,

| and restore the caller's STXIT before returning control to the
| caller.

| If an abend occurs, DFSORT/VSE will return control to the caller
| with a return code of 16 (unsuccessful completion) at the end of
| its abend recovery processing.

| DFSORT/VSE will not restore its STXIT when it receives control
| from the user exit routine (if any), so if the user exit routine
| established its own STXIT, it will remain in effect even though
| DFSORT/VSE has control. Then if an abend occurs, control will
| be returned to the user exit routine's STXIT routine and
| therefore DFSORT/VSE will not be able to do its abend recovery
| processing.

| Unlike STXIT=YES, STXIT=MIN will not degrade performance when
| COBOL or PL/I programs invoke DFSORT/VSE and use E15/E35 user
| exit routines to process records.

| **NO** specifies that DFSORT/VSE should not use its STXIT routine.
| DFSORT/VSE will not be able to do its abend recovery processing.

| *Default:* STXIT=MIN.

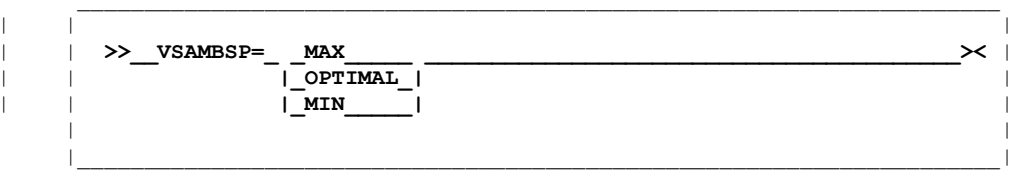
| **Notes:**

| 1. If you have COBOL or PL/I programs that invoke DFSORT/VSE and use
| E15/E35 user exit routines to process records, we recommend using
| STXIT=MIN.

| 2. If you do not have COBOL or PL/I programs that invoke DFSORT/VSE
| and use E15/E35 user exit routines to process records, we
| recommend using STXIT=YES.

| 3. If an E31 user exit routine is used and both MINSTXIT and CKPT are
| in effect, DFSORT/VSE's STXIT is restored when the user exit
| routine returns control to DFSORT/VSE.

| **VSAMBSP**



| Specifies the number of VSAM buffers DFSORT/VSE can use for VSAM (or
| SAM ESDS accessed as VSAM) input and output file processing.
| Additional buffers can improve performance.

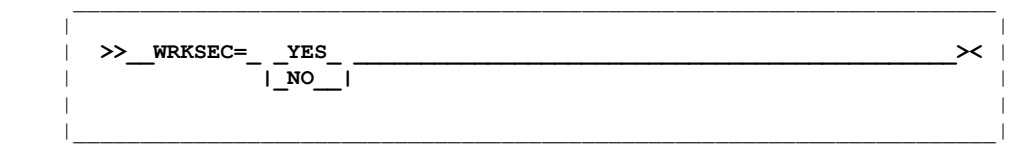
| **MAX** specifies that DFSORT/VSE can use the maximum number of
| buffers. MAX can provide the most significant performance
| improvements, but can result in excessive system paging
| activity for storage-constrained systems.

| **OPTIMAL** specifies that DFSORT/VSE can use the optimal number of
| buffers. OPTIMAL can provide substantial performance
| improvements with minimal impact on system paging activity.

| **MIN** specifies that DFSORT/VSE is to use the minimum number of
| buffers.

| *Default:* VSAMBSP=OPTIMAL.

WRKSEC



| Specifies whether secondary allocation of SAM ESDS or SD work files is

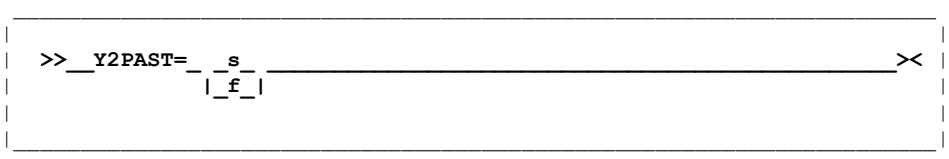
```
|     to be used.
```

| **YES** specifies that secondary allocation of SAM ESDS or SD work files
| is to be used.

| **NO** specifies that secondary allocation of SAM ESDS or SD work files
| is not to be used.

Default: WRKSEC=YES.

Y2PAST




```
|     Specifies the sliding (s) or fixed (f) century window.  The century

|     window is used with DFSORT/VSE's Y2C, Y2Z, Y2S, Y2P, Y2D, and Y2B

|     formats to correctly interpret two-digit year data values as
      four-digit year data values.
```

s specifies the number of years DFSORT/VSE is to subtract from the current year to set the beginning of the sliding century window. Since the Y2PAST value is subtracted from the current year, the century window slides as the current year changes. s must be a value between 0 and 100.

f specifies the beginning of the fixed century window. f must be a value between 1000 and 3000.

Default: 80.

Example 1

```
Y2PAST=81
```

If the current year is 1996, the century window will be 1915 to 2014. This century window results in the following interpretation of Y2 format data by DFSORT/VSE:

```
Two-digit
year data     Interpretation
---------     --------------
       00               2000
       14               2014
       15               1915
       61               1961
       62               1962
       99               1999
```

```
Note that when the current year changes to 1997, the century window
for Y2PAST=81 will be 1916 to 2015;  the century window slides as the
current year changes.
```

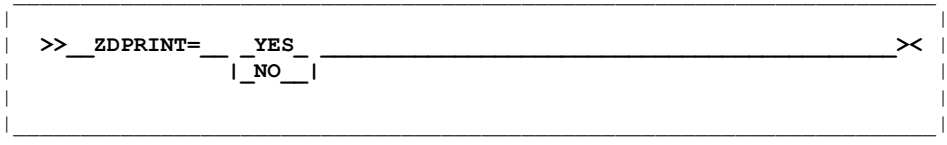
Example 2

```
Y2PAST=1962
```

The century window will be 1962 to 2061. This century window results in the following interpretation of Y2 format data by DFSORT/VSE:

```
Two-digit
year data     Interpretation
---------     --------------
       00               2000
       14               2014
       15               2015
       61               2061
       62               1962
       99               1999
```

| ZDPRINT




```
|     Specifies whether positive zoned decimal (ZD) fields resulting from
|     summation must be converted to printable numbers (that is, whether the
|     zone of the last digit should be changed from a hexadecimal C to a
|     hexadecimal F).
```

| **YES** means convert positive ZD summation results to printable numbers.
| For example, change hexadecimal F3F2C5 (prints as 32E) to F3F2F5
| (prints as 325).

| **NO** means do not convert positive ZD summation results to printable
| numbers.

| *Default:* ZDPRINT=NO.

1.4.2 Changing DFSORT/VSE Installation Defaults

If you are satisfied with the IBM-supplied installation defaults, you do not have to use the ILUINST macro or run this step. However, if you want to change one or more of the defaults, you must assemble ILUINST, specifying only those options you want changed.

You must include your ILUINST macro in the job stream shown in Figure 20 in topic 1.4.2 to change the defaults:

```
// JOB  5746-SM3 INSTALLATION DEFAULT          01
// LIBDEF *,SEARCH=(PRD2.PROD)                 02
// LIBDEF PHASE,CATALOG=PRD2.PROD              03
// OPTION CATAL,NODECK,NOXREF                  04
   PHASE   ILUSMANR,S,SVA                      05
   MODE    AMODE(ANY),RMODE(24)                06
   INCLUDE ILUSMANR,(ILUCTCHR)                 07
   INCLUDE ILUSMANR,(ILUSMANR)                 08
   INCLUDE ILUSMANR,(IJJCOD1N)                 09
   INCLUDE ILUSMANR,(ILULOADR)                 10
   INCLUDE ,(ILUINST)                          11
// EXEC ASSEMBLY                               12
   COPY ILUINST                                13
   ILUINST ...< your ILUINST macro >           14
   END                                         15
/*                                             16
// EXEC LNKEDT,PARM='MSHP'                     17
/&                                             18
```

Figure 20. Changing DFSORT/VSE Installation Defaults

Line Explanation

02
If DFSORT/VSE has been installed in a sublibrary other than the installation default library, change PRD2.PROD to your sublibrary. The SEARCH library points to the sublibrary where the source book ILUINST.A and object module ILUSMANR.OBJ are found.

03
If DFSORT/VSE has been installed in a sublibrary other than the installation default library, change PRD2.PROD to your sublibrary where ILUSMANR.PHASE is located.

Once the changes have been made, submit the job for assembly and link-editing on your system. This JCL will assemble control section ILUINST, relink-edit the ILUSMANR module, and catalog a new ILUSMANR.PHASE in the sublibrary where DFSORT/VSE is installed.

The assembler may issue MNOTE messages when it expands the ILUINST macro. These messages are printed in the assembler listing (see Appendix A, "Installation Messages" in topic A.0). The entire ILUINST macro is analyzed before generation is completed or terminated. If you have errors, correct them, then restart this step.

If you use the ILUINST macro more than once, you should be aware that each time it is run, it replaces all DFSORT/VSE installation options. For any option that is not specified, the default will revert to that originally supplied with DFSORT/VSE.

To restore all the original DFSORT/VSE installation defaults, you need only relink-edit ILUSMANR phase.

1.4.3 Modifying ILUMSGS Macro Message Text

If you want to issue DFSORT/VSE messages in a language other than English or to change the wording of some messages, you can modify the message text in the ILUMSGS macro.

The format of the ILUMSGS macro is:

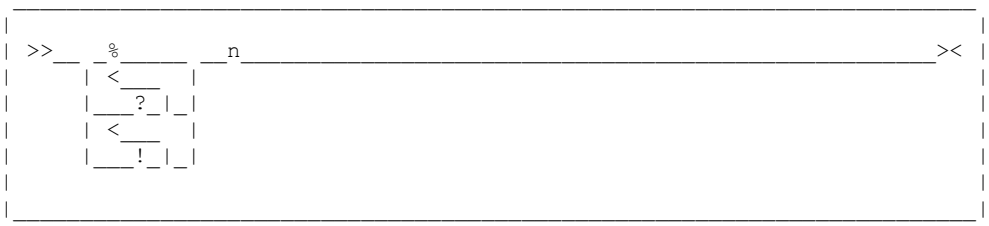
```
MACRO
ILUMSGS
```

```
ILUMSET 1,'text1...'
ILUMSET 2,'text2...'
      .
      .
      .
END
```

Each message macro consists of a macro call name (ILUMSET), a message number, and a text string. The text string consists of fixed characters and insert fields.

When DFSORT/VSE constructs a message to be printed, it replaces each insert field with actual data (for example, a specific number or keyword). When a message is translated into another language, the position and order of various insert fields may have to be changed to meet the syntax requirements of that language. Therefore, insert fields in each DFSORT/VSE message are given unique identifiers to allow their position and order to be changed as needed.

The insert fields supply variable information in the following forms:



Where

- % represents numeric data (one % for a number). Leading zeros are not printed.
- ? represents character data (one ? for each character of the maximum length field).
- ! represents hexadecimal information (one ! for each character of the maximum length field).
- n is an insert code (1, 2, ... 16) which uniquely identifies an insert field.

Subtopics:

1.4.3.1 Coding Rules for ILUMSGS

Code your updates to ILUMSGS, keeping these guidelines in mind:

- An ampersand (&) must be coded as two ampersands, following the assembler coding convention.
- An apostrophe (') must be coded as two apostrophes, following the assembler coding convention.
- Double-byte data may be used in a text string. The start of double-byte data is delimited by the shift-out control character (X'0E'), and the end by the shift-in control character (X'0F'). Shift-out and shift-in control characters must be paired. Either no bytes or an even number of bytes must appear between paired shift-out and shift-in control characters.
- Message macros must not be added or deleted.
- Macro call names (ILUMSET) and message numbers must not be changed.
- Fixed characters can be changed, added, deleted, or relocated.
- Insert field characters (?n, %n, !n) must not be changed, added, or deleted.
- Insert fields must not be made contiguous.
- Insert fields can be relocated.
- Each text string must not exceed 236 bytes.
- You can request that the message text be printed on several consecutive lines (message lines). Use the semicolon (;) to indicate the end of a message line. The semicolon will not be printed.

  For example, message text

```
'INPUT SEGMENT TOO LONG, SEGMENT LENGTH = %1, TOTAL LENGTH = %2, ????????3'
```

  will be printed as

```
ILU377A ik INPUT SEGMENT TOO LONG, SEGMENT LENGTH = xxxx,
           TOTAL LENGTH = yyyy, nnnnnnn
```

- Each message line must not exceed 109 bytes.
- Because the characters ?, %, and ! have special meanings, you must not use them as text characters.
- The special symbols ?n, %n, !n, and the semicolon (;) must not appear between shift-out and shift-in control characters.
- If continuation is necessary, use standard macro statement continuation rules.

The examples below illustrate message translation, including the method for changing the order of insert fields within the message text, when appropriate.

Figure 21 shows how the text of messages ILU266A and ILU656I can be translated into French.

```
ILU266A:

  Lx VALUE TOO LARGE FOR SORTINy

  LA VALEUR Lx EST EXCESSIVEMENT GRANDE POUR SORTINy

ILU656I:

  xxxxxxK EXTRA STORAGE REQUESTED, yyyyyyK AVAILABLE

  IL Y A yyyyyyK DE MEMOIRE, IL FAUT xxxxxxK DE PLUS
```

Figure 21. Suggested Translations for Two ILUMSGS Messages

Figure 22 shows the original text of these messages in the ILUMSGS macro.

```
.
.
.
ILUMSET 266,'L%1 VALUE TOO LARGE FOR SORTIN%2'
.
.
.
ILUMSET 656,'??????1K EXTRA STORAGE REQUESTED, ??????2K AVAILABLE'
.
.
.
```

Figure 22. The ILUMSGS Macro with Two Original Messages

Figure 23 shows the changes made in the ILUMSGS macro.

```
.
.
.
ILUMSET 266,'LA VALEUR L%1 EST EXCESSIVEMENT GRANDE POUR SORTIN%2'
.
.
.
ILUMSET 656,'IL Y A ???????2K DE MEMOIRE, IL FAUT ??????1K DE PLUS'
.
.
.
```

Figure 23. The ILUMSGS Macro with Two Messages Changed to French

---

1.4.3.2 Making the Change

To change the message text in the ILUMSGS macro, you can use Interactive Interface to do the following:

1. Punch the ILUMSGS macro from the sublibrary where DFSORT/VSE V3R4 is installed to your VSE/ICCF library, using the LIBRP macro.

2. Edit the messages provided in ILUMSGS as needed using the editor available under the Interactive Interface.

3. Use the LIBRC macro to catalog your changed ILUMSGS macro in a VSE sublibrary. If you want to save the IBM-supplied ILUMSGS macro shipped with the product tape, you have to catalog the changed ILUMSGS macro in your own sublibrary (for example, SORT.SM4), otherwise you can catalog the changed ILUMSGS macro in the sublibrary where DFSORT/VSE V3R4 is installed (for example, PRD2.PROD).

To apply the changes made to the ILUMSGS macro you must reassemble and relink-edit the ILUMSGA module for modifying DFSORT/VSE messages and the ILUMUTL module for modifying ICETOOL messages.

Figure 24 shows a sample of the job to assemble and link-edit the ILUMSGA and ILUMUTL modules.

```
// JOB APPLYMSG                                          01
// LIBDEF *,SEARCH=(SORT.SM4,PRD2.PROD)                  02
// LIBDEF PHASE,CATALOG=PRD2.PROD                        03
// OPTION CATAL,NODECK,NOXREF                            04
   PHASE ILUMSGA,S,SVA                                   05
   MODE  AMODE(31),RMODE(ANY)                            06
// EXEC ASMA90,SIZE=ASMA90                               07
* $$ SLI MEM=ILUMSGA.A,S=PRD2.PROD                       08
/*                                                       09
// EXEC LNKEDT,PARM='MSHP'                               10
   PHASE   ICETOOL,S                                     11
   MODE    AMODE(31),RMODE(24)                           12
   INCLUDE ICETOOL                                       13
   INCLUDE ,(ILUMUTL0)                                   14
   EXEC ASMA90,SIZE=ASMA90                               15
* $$ SLI MEM=ILUMUTL.A,S=PRD2.PROD                       16
/*                                                       17
// EXEC LNKEDT,PARM='MSHP'                               18
/&                                                       19
```

Figure 24. Job to Assemble and Link-edit the ILUMSGA and ILUMUTL Modules

Line Explanation

01

JOB job control statement. Introduces this application to the operating system.

02

Specifies the sublibrary (SORT.SM4) that contains the ILUMSGS macro, and the sublibrary (PRD2.PROD) where DFSORT/VSE was installed. What you specify in this line depends upon where you placed your copy of the ILUMSGS macro, and where you installed DFSORT/VSE.

03

Specifies the sublibrary (PRD2.PROD) in which the ILUMSGS and ICETOOL phases will be stored.

05-06

Linkage Editor control statements for ILUMSGA.

08

Specifies that the sublibrary member ILUMSGA.A is to be included in the input stream.

11-14

Linkage Editor control statements for ICETOOL.

16

Specifies that sublibrary member ILUMUTL.A is to be included in the input stream.

If you use double-byte data in message text, the DBCS assembler option must be specified.

If you violate any of the rules for modifying a message, you will invalidate that message, and an appropriate MNOTE message will be issued when you reassemble the message module.

1.4.3.3 Replacing ILUMSGS with ILUMSGSJ for Japanese Messages

DFSORT/VSE is shipped with English message text (in ILUMSGS) and Japanese message text (in ILUMSGSJ). If you want DFSORT/VSE to use the Japanese message text instead of the English message text, run the post-install job described in Figure 25. This post-install job does the following:

1. Renames ILUMSGS (English messages) to ILUMSGSE (English messages).
2. Renames ILUMSGSJ (Japanese messages) to ILUMSGS (Japanese messages).
3. Reassembles and link-edits the ILUMSGA and ILUMUTL modules using ILUMSGS (Japanese messages).

Figure 25 contains a sample of this post-install job.

```
// JOB ILUJMSG                                  01
// EXEC LIBR,PARM='MSHP'                        02
   AC S=PRD2.PROD                               03
   RENAME ILUMSGS.A  : ILUMSGSE.A               04
   RENAME ILUMSGSJ.A : ILUMSGS.A                05
/*                                              06
// LIBDEF *,SEARCH=PRD2.PROD                    07
// LIBDEF PHASE,CATALOG=PRD2.PROD               08
// OPTION CATAL,NODECK,NOXREF                   09
   PHASE  ILUMSGA,S,SVA                         10
   MODE   AMODE(31),RMODE(ANY)                  11
// EXEC   ASMA90,SIZE=ASMA90,PARM='DBCS'        12
* $$ SLI  MEM=ILUMSGA.A,S=PRD2.PROD             13
/*                                              14
// EXEC   LNKEDT,PARM='MSHP'                    15
   PHASE  ICETOOL,S                             16
   MODE   AMODE(31),RMODE(24)                   17
   INCLUDE ICETOOL                              18
   INCLUDE ,(ILUMUTL0)                          19
// EXEC   ASMA90,SIZE=ASMA90,PARM='DBCS'        20
* $$ SLI  MEM=ILUMUTL.A,S=PRD2.PROD             21
/*                                              22
```

```
|   // EXEC   LNKEDT,PARM='MSHP'                          23            |
|   /&                                                    24            |
|                                                                       |
|_______________________________________________________________________|
```

Figure 25. Job to Change English Message Text to Japanese Message Text

Note: You may need to set STDOPT HCTRAN=NO to prevent the DFSORT/VSE messages in the LISTLOG output from being translated to uppercase.

To change the Japanese message text back to English message text, run the job described in Figure 26. This job does the following:

1. Renames ILUMSGS (Japanese messages) to ILUMSGSJ (Japanese messages).
2. Renames ILUMSGSE (English messages) to ILUMSGS (English messages).
3. Reassembles and link-edits the ILUMSGA and ILUMUTL modules using ILUMSGS (English messages).

Figure 26 in topic 1.4.3.3 contains a sample of this job.

```
 _______________________________________________________________________
|                                                                       |
|                                                                       |
|   // JOB ILUJMSG                                       01            |
|   // EXEC LIBR,PARM='MSHP'                             02            |
|      AC S=PRD2.PROD                                    03            |
|      RENAME ILUMSGS.A  : ILUMSGSJ.A                    04            |
|      RENAME ILUMSGSE.A : ILUMSGS.A                     05            |
|   /*                                                   06            |
|   // LIBDEF *,SEARCH=PRD2.PROD                         07            |
|   // LIBDEF PHASE,CATALOG=PRD2.PROD                    08            |
|   // OPTION CATAL,NODECK,NOXREF                        09            |
|      PHASE  ILUMSGA,S,SVA                              10            |
|      MODE   AMODE(31),RMODE(ANY)                       11            |
|   // EXEC   ASMA90,SIZE=ASMA90                         12            |
|   * $$ SLI  MEM=ILUMSGA.A,S=PRD2.PROD                  13            |
|   /*                                                   14            |
|   // EXEC   LNKEDT,PARM='MSHP'                         15            |
|      PHASE  ICETOOL,S                                  16            |
|      MODE   AMODE(31),RMODE(24)                        17            |
|      INCLUDE ICETOOL                                   18            |
|      INCLUDE ,(ILUMUTL0)                               19            |
|   // EXEC   ASMA90,SIZE=ASMA90                         20            |
|   * $$ SLI  MEM=ILUMUTL.A,S=PRD2.PROD                  21            |
|   /*                                                   22            |
|   // EXEC   LNKEDT,PARM='MSHP'                         23            |
|   /&                                                   24            |
|                                                                       |
|_______________________________________________________________________|
```

Figure 26. Job to Change Japanese Message Text to English Message Text

1.4.4 Placing DFSORT/VSE Phases into the SVA

For improving the performance of DFSORT/VSE, SVA-eligible phases of DFSORT/VSE may be loaded into 24-bit and 31-bit SVA. Programs residing in the SVA are used concurrently by programs running in different partitions.

During IPL, VSE/ESA loads system phases into the SVA (according to internal load lists). In addition, during system startup VSE/ESA loads SVA-eligible phases through the SET SDL command into the SVA. This command maintains the SDL (system directory list) which includes the names of the phases that are to be loaded into the SVA. The SET SDL command can
| only be issued from the BG partition. You will need about 16 KB in the
| 24-bit SVA, 354 KB in the 31-bit SVA for SVA-eligible phases of DFSORT/VSE
| and 65 entries in the system directory list.

The operator can use the SET SDL command (from the BG partition) at any time after system startup to load phases into the SVA. Programs that are loaded into the SVA are stored in the virtual library area (VLA) of the respective SVA. Even if there are two SVAs in a system there is always only one SDL. It resides in the SVA (24-bit) and also addresses phases in the "high" VLA of the SVA (31-bit).

The SET SDL command loads phases with the attribute RMODE=ANY into the "high" VLA first. If the space is not sufficient, phases are stored in the VLA of the SVA (24-bit). *VSE/ESA Extended Addressability* provides details about the RMODE (residency mode) attribute and other items related to 31-bit addressing. *VSE/ESA System Control Statements* provides details about the storage allocation rules for 24-bit and 31-bit SVA.

To assist you in placing the DFSORT/VSE phases into the SVA, a load list named $SVASORT.PHASE is provided with the program package.

You can use the SET SDL command to specify a load list for DFSORT/VSE as shown by the job stream sample in Figure 27.

```
// JOB LSORTSVA  LOAD SVA-ELIGIBLE PHASES INTO SVA
// LIBDEF PHASE,SEARCH=PRD2.PROD
SET SDL
LIST=$SVASORT
/*
/&
```

Figure 27. Loading Phases into the SVA through a Load List

If DFSORT/VSE has been installed in a sublibrary other than the installation default library, change PRD2.PROD to your sublibrary.

To load SVA phases of DFSORT/VSE automatically during system startup modify procedure $0JCL through skeleton SKJCL0. You must include statements such as those shown in the example in Figure 27 in procedure $0JCL (except for // JOB and /&). This portion of the installation must be done by a system administrator.

1.4.5 Establishing DFSORT/VSE Cultural Environment Support

To provide cultural environment support, DFSORT/VSE must have access to one of the following:

- VSE/ESA 1.4 through 2.1 and the Language Environment for VSE/ESA 1.4.0 (with the C-language run-time support installed), or
- VSE/ESA 2.2 (with the C-language run-time support installed from the VSE/ESA Extended Base Products tape) or subsequent release.

To enable DFSORT/VSE's cultural environment support, the following post-install job must to be run after DFSORT/VSE is installed. The post-install job is shipped on the product tape as ILULOCEN.Z and can be found in the sublibrary where DFSORT/VSE is installed. Use the LIBRP macro to punch the post-install job to your VSE/ICCF sublibrary and edit it as needed for your environment. After you edit the post-install job, submit it from the VSE/ICCF sublibrary. The job will link-edit DFSORT/VSE's ILUSLOC phase with LE/VSE.

The output from the post-install job will list the Language Environment for VSE/ESA parts that are link-edited with DFSORT/VSE. You may want to save the output from this post-install job because you will need to rerun the post-install job again if maintenance is applied to any of the Language Environment for VSE/ESA parts listed in the output.

```
// JOB ILULOCEN  ENABLE CULTURAL ENVIRONMENT SUPPORT    01
// LIBDEF *,SEARCH=(PRD2.PROD,PRD2.SCEEBASE)            02
// LIBDEF PHASE,CATALOG=PRD2.PROD                       03
// OPTION CATAL,NODECK,NOXREF                           04
   PHASE   ILUSLOC,S,SVA                                05
   MODE    AMODE(31),RMODE(ANY)                         06
   INCLUDE ILUSLOC                                      07
/*                                                      08
// EXEC LNKEDT,PARM='MSHP'                              09
/&                                                      10
```

Figure 28. Job to Enable the Cultural Environment Support

LINE EXPLANATION

01
JOB control statement. Introduces this application to the operating system.

02
Specifies sublibraries, where DFSORT/VSE and the C-language feature of either LE/VSE or VSE/ESA have been installed. Change this line, if either DFSORT/VSE has been installed in a sublibrary other than PRD2.PROD, or the C-language feature has been installed in a sublibrary other than PRD2.SCEEBASE.

03
Specifies the DFSORT/VSE installation sublibrary. Change this line, if DFSORT/VSE has been installed in a sublibrary other than PRD2.PROD.

04-09
Job control and Linkage Editor control statements to link-edit DFSORT/VSE module ILUSLOC with the C-language subroutines.

You will need to rerun the post-install job if any of the following occur:

- If you reinstall Language Environment for VSE/ESA 1.4.0 or a subsequent release
- If you reinstall DFSORT/VSE Version 3 Release 4
- If you apply maintenance to any of the Language Environment for VSE/ESA parts that are link-edited with DFSORT/VSE.

| 1.4.6 Establishing DFSORT/VSE Online Message Explanations

| If you want to be able to display DFSORT/VSE messages on the console, load
| the DFSORT/VSE message explanations into the online message file using the
| VSE/ESA IESMSGS utility. This utility is available with VSE/ESA V2.

| Figure 29 contains a sample job that shows how to invoke the IESMSGS
| utility.

```
|   // JOB ILUSOME
|   // OPTION PARTDUMP,NOSYSDMP
|   // DLBL NEWMSGS,'VSE.MESSAGES.ONLINE',,VSAM,CAT=IJSYSCT
|   // EXEC IESMSGS,SIZE=60K,PARM='EU'
|   * $$ SLI MEM=ILUSOME.Z S=PRD2.PROD
|   /*
|   /&
```

| Figure 29. Load the DFSORT/VSE English Message Explanations into the
| Online Message File.

| Be sure to turn EXPLAIN OFF before you invoke the IESMSGS utility.
| Otherwise the file will be in use and will not be updated. When IESMSGS
| completes, turn EXPLAIN ON and the DFSORT/VSE explanations will be
| available.

Subtopics:

- 1.4.6.1 Displaying Japanese Message Explanations
- 1.4.6.2 Displaying English Message Explanations on a German Console

| 1.4.6.1 Displaying Japanese Message Explanations

| DFSORT/VSE is shipped with English message explanations (ILUSOME.Z) and
| Japanese message explanations (ILUSOMJ.Z). If you want Japanese message
| explanations you need to load the DFSORT/VSE Japanese message explanations
| into the online message file. Use the LIBRP command to punch the
| ILUSOMJ.Z (from the sublibrary containing DFSORT/VSE) to the ICCF library.
| Edit this new member and add the necessary JCL to the job and submit the
| job. Use the JCL in Figure 30as a guideline.

```
|   // JOB ILUSOMJ
|   // OPTION PARTDUMP,NOSYSDMP
|   // DLBL NEWMSGS,'VSE.MESSAGES.ONLINE',,VSAM,CAT=IJSYSCT
|   // EXEC IESMSGS,SIZE=60K,PARM='JU'
|        (input data from ILUSOMJ.Z)
|   /*
|   /&
```

| Figure 30. Load the DFSORT/VSE Japanese Message Explanations into the
| Online Message File.

| **Notes:**

| 1. PARM=JU is only valid for a Japanese VSE/ESA V2 system.

| 2. To view the Japanese message explanations, you must have a console
| that is capable of displaying Japanese characters.

| 1.4.6.2 Displaying English Message Explanations on a German Console

| DFSORT/VSE is shipped with English message explanations that will display
| all characters correctly on a German console (ILUSOMG.Z). To use this
| version of the English message explanations, run the job in Figure 31 in
| topic 1.4.6.2to load the DFSORT/VSE English message explanations for a
| German console into the online message file.

```
|   // JOB ILUSOMG
|   // OPTION PARTDUMP,NOSYSDMP
|   // DLBL NEWMSGS,'VSE.MESSAGES.ONLINE',,VSAM,CAT=IJSYSCT
|   // EXEC IESMSGS,SIZE=60K,PARM='GU'
|   * $$ SLI MEM=ILUSOMG.Z S=PRD2.PROD
|   /*
```

```
|    /&
```

```
| Figure 31. Load the DFSORT/VSE English Message Explanations for German
|            console into the Online Message File.
```

| **Note:** PARM=GU is only valid for a German VSE/ESA system.

2.0 Part 2. Tuning

Part 2 of this document provides information about tuning DFSORT/VSE, offering suggestions for reducing its use of system resources and achieving better turnaround time for applications that use DFSORT/VSE.

This information is intended for systems engineers, performance analysts, system programmers, and application programmers.

Subtopics:

2.1 Chapter 5. Introduction

This chapter offers information and recommendations on tuning DFSORT/VSE. It offers advice to the system programmer on how to install DFSORT/VSE and set default parameters. For application programmers, it offers suggestions on how to improve the performance of individual DFSORT/VSE applications. It describes the main indicators used to measure VSE/ESA performance. Since sort applications tend to be more complex and time-consuming than merge or copy applications, this book concentrates on techniques for improving the performance of sort applications. However, many of these techniques are also appropriate for copy and merge applications.

The tuning of DFSORT/VSE and the way applications use it is important because sorting and copying are two of the most frequently used functions at VSE/ESA sites.

More efficient use of DFSORT/VSE can lead to:

- Reduced use of system resources
- Reduced job turnaround time
- Improved productivity

All of these benefits can result in cost savings.

The actual performance of a sort application is dependent on many factors including record length, file size, partition size, total processor and virtual storage available, type and number of auxiliary storage devices, and specific functions used.

Elapsed time results for sorting in a multitasking environment are dependent on application and workload characteristics. Therefore, the results might differ from user to user.

Subtopics:

2.1.1 Tuning

The purpose of tuning DFSORT/VSE is to use system resources more efficiently. This is important at most sites, since there are usually too many demands made for limited resources. Although DFSORT/VSE automatically optimizes many of its tuning decisions, there are additional actions which a site and its DFSORT/VSE users can take to further improve DFSORT/VSE performance. These actions depend on the priority given to various performance objectives.

Different sites and programmers define efficient performance in different ways. A site with a primarily batch environment or an application programmer with a single task to complete probably measures performance based on elapsed time. Alternatively, a site experiencing high CPU usage or a programmer who is charged based on CPU time is more likely to evaluate efficiency in terms of reduced CPU time.

While improved performance is the objective of tuning, often it is necessary to compromise. That is, improving the use of one system resource can have a negative impact on other resources, in much the same way that giving more resources to one application can make it perform much better at the expense of degrading the performance of other applications that are competing for the same resources.

The main trade-offs that you should consider are:

Processor Load

Accounting charges are often based on the number of CPU service units used by an application. The more CPU time used, the higher the charges. Reducing an application's CPU time not only reduces these charges, but also enables other applications competing for the same CPU resource to complete sooner.

Paging Activity

System paging activity reflects how the system is managing virtual storage. During execution, whenever a required instruction or some data is not present in processor storage, execution is interrupted by a page fault. The required page must then be read into processor storage from the page data set. System paging activity can increase elapsed time for all programs.

I/O Activity

Some accounting charges are based on the I/O performed by an application. This is frequently measured by start I/O (SIO) counts. While SIO counts might not represent a completely accurate usage of I/O resources, they are important to many users and sites, and steps can be taken to reduce them.

Elapsed Time

Elapsed time is likely to be most important for sites, where processing of particular applications has to be completed within a certain period. It is also important to users whose productivity depends on having their applications complete as soon as possible.

DASD Utilization

In environments where DASD is constrained, the amount of auxiliary storage required by an application is of paramount concern. For DFSORT/VSE applications, this usually involves the efficient use of output and work files.

2.1.2 Performance Indicators

This section describes the ways in which you can measure the performance factors listed above for DFSORT/VSE. Often, performance is evaluated as a combination of some or all of them. The priorities given to each depend upon site objectives.

Subtopics:

2.1.2.1 Processor Utilization

CPU fields are used to measure the amount of work performed by the processor, as opposed to work performed by the I/O and storage subsystems.

Task Time
This is the actual time used by a job or job step in the system.

Overhead Time
This is the time needed for activities that cannot be charged to a specific program or partition. For example, the time for calling a routine for error recovery, or the time from the start of the $JOBACCT routine to the processing of the EXEC job control statement. All SVC processing is counted as active processor time for the job or job step. Overhead time is distributed over active partitions in proportion to used CPU time.

For VSE/ESA systems, CPU time is defined as the sum of these fields.

2.1.2.2 System Paging

Unlike most of the other performance indicators, the system paging activity cannot be measured on a job by job basis. To be meaningful, paging activity measurements are usually associated with particular workloads, or groups of applications that are run simultaneously on a given processor. As such, system paging activity is more a measure of the performance and throughput of the entire system than of the performance of individual applications being run on the system.

2.1.2.3 I/O Activity

This represents the movement of data between the processor and DASD or tape devices. The effective use of I/O resources is important to a site and I/O activity is often an important component of site accounting methodologies.

Because performing input to and output from DASD and tape devices typically takes much longer than manipulating the data in real storage, the amount of I/O performed is a key component of an application's elapsed time. Therefore, reducing I/O generally improves an application's elapsed time.

I/O activity is primarily measured in the following ways:

SIOs
The number of start I/O (SIO) commands issued.

Device Connect Time
The amount of time a particular device is dedicated for the I/O transfer.

Channel Usage
The percentage of time a channel is busy initiating, transferring, or completing the movement of data between a device and the CPU.

While SIOs are often used as a measure of I/O activity, their counts can be extremely misleading. For example, one SIO can be used to transfer a few bytes or dozens of megabytes. For a complete analysis of total I/O performance, device connect time and channel usage should also be considered.

2.1.2.4 Elapsed Time

Elapsed time refers to the amount of "wall clock" time from initiation to termination of the application. For typical sorting applications, elapsed time is composed primarily of I/O time, with CPU time and I/O queueing time also contributing significantly.

The elapsed time for an application can differ from run to run, depending upon the amount of competition from other applications for system resources. Accurate elapsed time comparisons can be done only if the system has no other applications running.

2.1.2.5 DASD Utilization

In certain environments, DASD space usage is a more important characteristic of an application's performance than CPU time or elapsed time. Inefficient DASD usage is usually measured in terms of the amount of DASD space that is allocated, but not used (also referred to as wasted space). Frequently, large amounts of DASD space is wasted as a result of inefficient blocking.

2.1.3 Where to Find Performance Indicators

Performance indicators can be found in a number of different places. Where you choose to find them depends on your own knowledge of VSE/ESA, what tools are available, and how much effort you want to expend.

Sources you can use include:

- Log output
- DFSORT/VSE messages
- Job accounting
- Displaying system activity

Subtopics:

2.1.3.1 Log Output

The time-of-day clock provides a consistent measure of elapsed time suitable for time-of-day indication. The time-of-day and the date are automatically included with each JOB and /& job control statement that is printed on the SYSLST or SYSLOG file.

2.1.3.2 DFSORT/VSE Messages

If option DIAG is specified on the ILUINST macro or OPTION control statement, DFSORT/VSE issue messages quantifying its use of certain system resources. This is the simplest way of accessing such information. These messages are useful for tuning. A sample of DFSORT/VSE messages is shown in Figure 32. Explanations of these messages can be found in *Messages, Codes and Diagnosis Guide*.

```
|   ILU000I A0 --- CONTROL STATEMENTS/MESSAGES --- DFSORT/VSE 5746-SM3 RELEASE 4.0  DATE 05/29/1998
|               OPTION PRINT=ALL,ROUTE=LST,GVSIZE=MAX,DIAG
|               SORT FIELDS=(1,2,A),FORMAT=BI,WORK=0
|               RECORD TYPE=F,LENGTH=100
|               INPFIL BLKSIZE=8000
|               OUTFIL BLKSIZE=8000
|               END

|   ILU816I B3 GETVIS SORTING AREA OBTAINED = 1351680 BYTES BELOW, 0 BYTES ABOVE
|   ILU851I C0 35
|   ILU020I C0 STORAGE USED = 102280 BYTES
|   ILU808I C0 MODULE STATUS: SVA
|   ILU021I C0 FIXABLE STORAGE = 0 BYTES
|   ILU843I C0 PH1 0 2
|   ILU843I C0 PH2 0 0
|   ILU843I C0 PH3 2 0
```

```
|   ILU852I C0 3 3
|   ILU845I C0 102
|   ILU806I C0 PH0: TIME A: 0 SEC, TIME B: 0 SEC
|   ILU817I E0 IN-CORE SORT
|   ILU345I J0 GETVIS SORTING AREA USED = 116232 BYTES
|   ILU806I J0 PH1: TIME A: 1 SECONDS, TIME B: 0 SECONDS
|   ILU334I J0 SORT CAPACITY APPROXIMATELY 11500 RECORDS
|   ILU323I J0 SORT  COMPLETE, IN 1000, OUT 1000
|   1S55I  LAST RETURN CODE WAS 0000
|   EOJ GETVIS    MAX.RETURN CODE=0000         DATE 05/29/98,CLOCK 09/12/02,DURATION   00/00/36
```

Figure 32. Sample of DFSORT/VSE Messages.

2.1.3.3 Job Accounting

Job accounting is a system function that lists the system resources used by all jobs.

If accounting support is available, VSE/POWER automatically collects job accounting information for every partition under its control and stores this information, for every job step, in chronological order in the account file.

To have VSE/POWER job accounting support available, the system programmer specifies the account records that are needed in the POWER generation macro. VSE/POWER needs some additional processor, virtual storage, and disk space to accommodate the VSE/POWER account file. VSE/POWER produces various types of account records, and you can use ICETOOL or write an evaluation program of your own to process these records. For detailed information about these requirements, see *VSE/POWER Administration and Operation*.

Job accounting is a function that accumulates accounting information for each job step in a job accounting table to be used for charging the users of the system, for planning new applications, and for supervising system operation more efficiently. At the end of each job step or job, accounting information is accumulated in a table for that partition and can be processed by a user exit routine.

You may want to write a program with the name $JOBACCT to add information to the execution account record. See *VSE/ESA Guide to System Functions* for further details.

If you decide to use the job accounting facility, you must catalog your routine into the system sublibrary. At IPL, $JOBACCT is automatically loaded into the SVA if it is SVA eligible. If not, $JOBACCT is loaded into the corresponding partition at the end of each job step.

2.1.3.4 Displaying System Activity

The Interactive Interface provides two dialogs that display general information about current system activity. These are the *Display System Activity* and the *Display Channel and Device Activity* dialogs. Each dialog interactively displays general information about current system activity. This includes such things as:

- CPU usage
- I/O activity
- Paging activity

From the *VSE/ESA Function Selection* dialog, as a system administrator, you can access the *Display System Activity* dialog as follows:

```
3     (Operations)
6     (System Status)
1     (Display System Activity)
```

```
 ____________________     ____________________________________
|  Operator          |   |  Synonym                           |
|  Fast Path:   361  |   |  Default:    da     Yours:         |
|____________________|   |____________________________________|
```

The *Display System Activity* dialog shows current system activity. From this dialog, you can obtain further information about dynamic classes, dynamic partitions, and CPU activity.

The *Display Channel* and *Device Activity* dialog lets you monitor the use of I/O devices. You can select information about a single device address or about a range of device addresses. As system administrator you can access the dialog. Start with the *VSE/ESA Function Selection* dialog and select:

```
3     (Operations)
6     (System Status)
2     (Display Channel and Device Activity)
```

```
 ____________________     ____________________________________
|  Operator          |   |  Synonym                           |
|  Fast Path:   362  |   |  Default:    sio    Yours:         |
|____________________|   |____________________________________|
```

The *Display Channel and Device Activity* dialog shows current device activity. This dialog displays information from the time you selected the dialog. You can refresh the dialog by pressing **ENTER**. This updates the display to show the most current information.

For detailed information about the dialogs refer to *VSE/ESA Operation*.

2.2 Chapter 6. Analyzing DFSORT/VSE Use

Tuning activity is often started when a particular application is taking too long to complete or is using an excessive amount of system resources. But if such a "problem application" does not exist at your site, where do you start to look for ways to improve DFSORT/VSE's performance or balance its use of resources with other requirements?

This chapter outlines the actions available to you for tuning the use of DFSORT/VSE. It shows what information you need about DFSORT/VSE, where to find it, and what methods are available for collecting tuning data.

Subtopics:

2.2.1 Using the DIAG Option

The DIAG option can be used for all DFSORT/VSE applications (ILUINST macro), or for specific DFSORT/VSE applications (OPTION control statement), to investigate how well DFSORT/VSE is performing in its current environment and determine if improvements can be made.

If DIAG is in effect when DFSORT/VSE application is run, you receive additional messages about DFSORT/VSE's storage use, optimization parameters, data handling, and time required, as shown in the tuning table in Figure 33. The table also shows how to interpret the various items of information, and how they can be used to improve performance.

Figure 33. Tuning Table

Message Concerning:	**Interpretation:**
VIRTUAL STORAGE USAGE	
Whether DFSORT/VSE is executed using the SVA	Execution using the SVA

Virtual storage used by DFSORT/VSE	Determined by the partition allocation, the SIZE operand of the EXEC job control statement, or the STORAGE operand of the OPTION control statement.
Dataspace sorting	Ensure dataspace sorting is used (if appropriate) and determine how much data space is used.
Getvis sorting	Ensure getvis sorting is used (if appropriate) and determine how much GETVIS area is used.
Modal record length	When the input file consists of variable-length records, you can aid performance by careful specification of L5 on the LENGTH operand of the RECORD control statement. The L5 parameter value gives the record length that occurs most frequently in the input file (modal length). If you do not specify a value for L5, it is assumed to be equal to the average of the maximum and minimum (L2 and L4) record lengths in the input file. For optimum performance, both L4 and L5 should be specified. Specifying an inaccurate modal length may cause reduced performance.
WORK SPACE USAGE	
Allocation of work space	Use information on work space utilization to make sure you are not being over-generous or too sparing with work file space. If you have an application that is run regularly, and the amount of data is likely to grow, you will probably want to allow a generous amount of space. If you have a shortage of direct access space, you may want to trim the allocation to the minimum.
Sort capacity	Compare DFSORT/VSE's estimate of how many records it could handle with the number of records actually sorted.
Work space used	Compare the amount of space actually used with the amount allocated.
OPTIMIZATION PARAMETERS	
Block size for index and work buffers	With CKD devices, work block size can be up to a full track length. If it is less, performance might be improved if you increase virtual storage size. Be sure that you have enough real storage to support the increased virtual storage.
Merge order (M) in phases 2 and 3	A high merge order provides improved performance. If it is less than, for example, 8, performance might be improved if you increase it by increasing virtual storage size.
DATA MOVEMENT	
Number of data and index blocks moved	Increased storage allocation can lead to reduced data moved.
Number of physical and logical strings moved by the partitioning part of DFSORT/VSE (phase 2)	The amount of data movement in phase 2 can give you an idea of how well DFSORT/VSE is performing, as can the number of physical strings moved in partitioning. If, in partitioning, the number of logical strings is lower than the number of physical strings, your input file is not random. The amount of difference gives an indication of the degree of randomness.
TIME REQUIRED	

CPU and elapsed time for each of the phases	Enables you to study the effect of your tuning measures, if you have job accounting in your supervisor.

2.2.2 Using Job Accounting

If accounting support is available, VSE/POWER automatically collects job accounting information for every partition under its control and stores this information for every job step in chronological order in the account file. Types and layouts of account records are described in *VSE/POWER Application Programming*. By means of the PACCOUNT VSE/POWER command, you can request that this information be copied into a separate disk or tape file. You also can request that the account information be spooled as punched output and queued into the punch queue. You can use ICETOOL or write an evaluation program of your own to process these records.

VSE/ESA provides interface support for all partitions in the system. At the end of each job step or job, accounting information is accumulated in a table for that partition and can be processed by a user-written routine. This routine can extract data for such purposes as charging system usage and supervising system operation, or for planning and tuning new applications.

You can add information to every execution account record by writing a routine under the name of $JOBACCT that makes use of the PUTACCT macro. For guidance on how to write the routine, see *VSE/ESA Guide to System Functions*.

Job Control calls your $JOBACCT program at the end of every job or job step.

2.2.3 Using the Dialogs for Displaying System Status

Display System Activity and *Display Channel and Device Activity* dialogs dynamically provide system status information for daily operation. The information available through these dialogs can be used to:

- Observe system behavior.

  The data shows:

  - How much CPU and I/O resources are used by the system and by each partition.
  - How well disk I/O activity is balanced.

    - How rapidly each partition is progressing.

- Diagnose initial problems or performance.

  The information may indicate high paging rates or that the CPU is being used too heavily.

  When the system has poor response times although paging and CPU use are relatively low, the *Display Channel and Device Activity* dialog may show where I/O problems are located.
- Understand the effect of tuning actions.

  For example, you may use the **PRTY** command to:
    - Modify partition priorities.
    - Activate or deactivate the partition balancing.

  You can then monitor the new resource distribution.

  Review I/O activity and CPU use on a job basis to approximate the resources you need.
- Follow the progress of test runs for application programs.
- Follow the progress of important batch applications.
- Observe what the system is doing.

2.3 Chapter 7. Environment Considerations

While DFSORT/VSE automatically optimizes many of its tuning decisions to improve performance, the environment in which it runs affects its overall performance and, therefore, the programs and applications that use DFSORT/VSE. DFSORT/VSE is designed to take advantage of the major components of IBM computer systems.

The purpose of this chapter is to describe how DFSORT/VSE modifies its operation to take advantage of these components and the benefits it derives from them.

The majority of the discussion in this chapter is relevant only to sort applications because sorting uses the most system resources and is perhaps the most important to tune.

Within an IBM computer system, data resides across a storage hierarchy. Each level of this hierarchy is represented in hardware by a different component, and the amount of each component in a system is usually variable between some minimum and maximum levels. A typical representation of these components would be processor storage, DASD, and tape.

Subtopics:

2.3.1 Processor Storage

Of all the components that affect DFSORT/VSE's performance, processor (or real) storage is the most crucial. Sorting is a memory-intensive operation. Without enough processor storage to back the virtual storage needs of DFSORT/VSE, its performance (as well as the performance of other applications on the system) degrades significantly due to excessive system paging activity.

To sort efficiently, DFSORT/VSE needs large amounts of virtual storage. Its needs grow with the amount of data being sorted: a file four times as large as another requires roughly twice as much virtual storage to sort with the same degree of efficiency. If this virtual storage is not backed by central storage when DFSORT/VSE is running, there is a noticeable performance degradation on the system.

Processor storage plays an important role in the use of dataspace and getvis sorting. DFSORT/VSE bases its use of its very efficient sorting methods on the amount of processor storage it can use without causing excessive paging on the system. The monitoring of paging activity of the system is performed only when DSPSIZE=MAX or GVSIZE=MAX is in effect. Note that if you specify a value other than MAX for DSPSIZE or GVSIZE, excessive paging activity may occur. See "Sorting with Data Space" in topic 2.4.4 and "Sorting with GETVIS Area" in topic 2.4.5 for more information about dataspace and getvis sorting.

2.3.2 DASD

In most cases, DFSORT/VSE automatically adjusts itself to take advantage of the geometry of the type of DASD available to it. This is especially true for the work files, whose block sizes and distribution of data play crucial roles in the performance of DFSORT/VSE.

The location of input, output, and work files, as well as the speed of the DASDs has a significant effect on the performance of a sort application. For best results, these files should be placed on different storage subsystems to avoid channel, control unit path, and device contentions. To obtain the maximum performance benefit, these files (or at least the input and output files) should be placed on the fastest DASDs. In general, while DFSORT/VSE has no control over where its files are allocated, it does automatically optimize its access patterns based on file location to achieve the best possible performance.

In addition to file location, certain file characteristics, such as block size, are also very important when considering performance. As mentioned before, DFSORT/VSE automatically chooses an optimal block size for its work files.

2.3.3 Tape

Tape is at the bottom of the storage hierarchy because it is the least expensive on a per-byte basis. It has the highest capacity for data storage of any component in the storage hierarchy. But, it also has the slowest access time and the data must be accessed sequentially.

Based on these characteristics, tape is most useful for input and output files. Tape devices are not used for work files since a fast access rate is critical and work data tends to be accessed in a nonsequential manner.

2.3.4 Virtual Storage

Logically, DFSORT/VSE (like any other application) works in a partition within a virtual address space. A VSE/ESA partition is composed of a program area, which is always below 16 MB virtual, and GETVIS areas (24-bit and 31-bit) which can be used dynamically.

The size of the partition program area is determined by the user with installation defaults such as STORAGE, as well as run-time options such as SIZE (from the EXEC or SIZE job control statement) and STORAGE (from the OPTION control statement).

Certain functions require virtual storage dynamically during program execution. VSE/ESA maintains a GETVIS (Get Virtual Storage) area located within a partition for this purpose. In addition DFSORT/VSE uses GETVIS areas (24-bit and 31-bit) for getvis sorting to improve the performance of DFSORT/VSE applications. The size of the partition GETVIS area for getvis sorting is determined by the user with an installation default or run-time option GVSIZE.

In addition to address spaces (in which static and dynamic partitions reside) VSE/ESA supports data spaces. A data space is an area of virtual storage like an address space but it contains data only. It can have a maximum size of up to 2 GB. DFSORT/VSE uses a data space for dataspace sorting to improve the performance of DFSORT/VSE applications. The size of the data space for dataspace sorting is determined by the user with the installation default or run-time option DSPSIZE. Dataspace sorting is never used in conjunction with getvis sorting.

If you provide DFSORT/VSE with enough virtual storage, it can sort an input file entirely in virtual storage (an "incore" sort) without the need of work files. This is the preferred method of sorting files up to a few megabytes in size.

When an incore sort is not possible or practical, DFSORT/VSE processes a portion of the input file at a time and then combines all of these processed portions together into a final sorted file. DFSORT/VSE excels at allocating virtual storage effectively to minimize both the number of portions and the time spent combining the portions into an output file.

VSE/ESA uses SVA areas to keep reenterable phases. VSE/ESA supports two kinds of SVA areas: 24-bit SVA and 31-bit SVA (above 16 MB virtual). SVA-eligible phases of DFSORT/VSE may be loaded into 24-bit SVA and 31-bit SVA.

DFSORT/VSE also tries to move as many of its data areas above 16 MB virtual as it can to help provide virtual storage constraint relief for the system.

2.4 Chapter 8. Installation and Run-Time Options

Improving the performance of DFSORT/VSE consists of a number of activities, including:

- Tuning on a site-wide or system level
- Tuning of individual applications
- Designing efficient applications

Generally, the success of each depends upon the success of the others. For instance, suppose we have a DFSORT/VSE application which has been carefully designed and tuned, and would be a good candidate for dataspace sorting. It might be adversely affected by the site's decision to set the default DSPSIZE operand to 0. Such a setting would have the effect of turning off dataspace sorting (unless the programmer has specifically overridden it in the application).

Even the best system tuning may be wasted if applications use sorts unnecessarily. For example, an application that sorts two already sorted files into one could be changed to more efficiently use the DFSORT/VSE merge function. Likewise, an application that uses a sort to extract a subset of the records, but does not rearrange the records in any way, could be changed to more efficiently use the DFSORT/VSE copy function.

This chapter offers advice for system programmers and application developers who are responsible for installing and using DFSORT/VSE. It explains how to set installation and run-time options to improve the performance of DFSORT/VSE.

Subtopics:

2.4.1 Installing DFSORT/VSE

The way DFSORT/VSE is installed can affect its performance. This section explains some of the things you should consider.

Subtopics:

2.4.1.1 DFSORT/VSE in SVA

There are several performance benefits that can be gained by placing SVA-eligible phases of DFSORT/VSE in the shared virtual area (SVA):

- Programs residing in SVA can be used concurrently by two or more application programs running in different partitions. This enables processor storage to be used more efficiently and cuts down on system paging.

- When SVA-eligible phases of DFSORT/VSE are put in the SVA, they are not loaded into your partition. This saves unnecessary paging and time. This is especially noticeable for smaller DFSORT/VSE applications, which tend to make up the bulk of DFSORT/VSE applications at most sites.

| Since DFSORT/VSE is invoked so frequently and the 24-bit SVA requirements
| of DFSORT/VSE are very low, it is a prime candidate for installation in
| the SVA. Make sure the 31-bit SVA is large enough so that the 31-bit SVA
| eligible phases are not loaded into the 24-bit SVA.

2.4.1.2 Changing Installation Defaults

DFSORT/VSE is shipped with a set of installation defaults. These defaults set values for various DFSORT/VSE parameters. They can be changed on a site-wide level using the ILUINST macro. Most of the defaults can be overridden for specific applications by setting the appropriate run-time options.

The ILUINST macro can be used to modify the IBM-supplied DFSORT/VSE installation defaults. Many of these installation options have an impact on performance, including:

- Virtual storage limit
- Data space limit
- GETVIS area limit
- VSAM buffers
- Use of VERIFY, EQUALS, ERASE, ALTSEQ and LOCALE

Modifications to ILUINST should be carefully chosen to reflect how you want DFSORT/VSE to run at your site. You should

only specify ILUINST options if the supplied default values are not acceptable. See Chapter 4, "Customizing DFSORT/VSE" in topic 1.4 for complete details of ILUINST options.

The DFSORT/VSE installation defaults are based on customer feedback. They should only be changed on an exception basis.

Installation defaults that are inappropriate for an application should be overridden for that application with the corresponding run-time options. See *Application Programming Guide* for complete details of run-time options.

2.4.2 Site-Wide Performance Options

The following site-wide options influence the performance of DFSORT/VSE. They are explained in more detail in the following sections. Refer to *Application Programming Guide* for a complete list of run-time override options.

Figure 34. ILUINST Options Which Influence DFSORT/VSE Performance

Type	ILUINST Option	Description	Dependency	Run-time Option
Option that tailors partition program area	STORAGE	Upper limit for program area.	Limited by SIZE statement or SIZE operand of EXEC statement.	STORAGE
Option that affects use of dataspace sorting	DSPSIZE	Upper limit for data space size.	Data space limited by SYSDEF VSE/ESA command or job control statement.	DSPSIZE
Options that affect use of getvis sorting	GVSIZE	Upper limit for GETVIS area size	Limited by the partition GETVIS area size.	GVSIZE
	GVSRLOW	24-bit GETVIS area reserved for use by system and user application	Only used when GVSIZE=MAX in effect	GVSRLOW
	GVSRANY	31-bit GETVIS area reserved for use by system and user application	Only used when GVSIZE=MAX in effect	GVSRANY

Option that affects VSAM (or SAM ESDS accessed as VSAM) input and output file processing	VSAMBSP	The number of VSAM buffers to use.		None
Other options that can degrade performance	EQUALS	Whether input order is preserved for records with equal control fields		EQUALS NOEQUALS
	VERIFY	Whether output records are checked after being written		VERIFY NOVERIFY
	ERASE	Whether work files are erased, if they have been used		ERASE NOERASE
	ALTSEQ	Whether an altered EBCDIC collating sequence is used		ALTSEQ
	LOCALE	Whether locale processing is used		LOCALE

Note: If DFSORT/VSE is invoked from another application or passes control to user exit routines, the MINSTXIT or NOSTXIT option may improve sorting performance.

2.4.3 Virtual Storage

In general, the more virtual storage you make available to DFSORT/VSE the better performance you receive. The default amount of virtual storage available to DFSORT/VSE is defined when it is installed. DFSORT/VSE generally uses all the virtual storage available to it.

DFSORT/VSE sorts most efficiently when sufficient virtual storage is available to enable an optimal balance between placing data in virtual and auxiliary storage. When virtual storage is limited, DFSORT/VSE must expend more resources to transfer data

between virtual and auxiliary storage, which causes increased CPU time, elapsed time, and I/O usage.

Sufficient processor storage must be available to support virtual storage requirements of DFSORT/VSE. Supplying DFSORT/VSE with more virtual storage might not improve performance if the available system resources cannot accommodate the corresponding increase in virtual storage activity. If processor storage resources become overcommitted, excessive paging can result. This can cause the performance of both DFSORT/VSE and the system to degrade. It is important, therefore, to balance virtual storage resources supplied to DFSORT/VSE with the overall system resource requirements.

Subtopics:

2.4.3.1 DFSORT/VSE Options and Virtual Storage

The STORAGE, GVSIZE, and DSPSIZE options affect the use of virtual storage as follows:

- The STORAGE option defines how much partition program area can be used by DFSORT/VSE. This area is allocated statically below 16 MB virtual. When GVSIZE=0 and DSPSIZE=0, it is also used as the sorting area which holds the records currently being processed.
- The GVSIZE option enables getvis sorting and defines how much partition GETVIS area can be used for the DFSORT/VSE sorting area.
- The DSPSIZE option enables dataspace sorting and specifies that a data space can be used for the DFSORT/VSE sorting area.

Only one sorting area is allocated by DFSORT/VSE. Therefore, getvis and dataspace sorting are mutually exclusive. In the case of either getvis or dataspace sorting there is no reason to specify a large value for the STORAGE option.

In general, DFSORT/VSE uses dataspace sorting if it is enabled. However, if the requested amount of data space is not available, DFSORT/VSE will not use dataspace sorting.

If dataspace sorting is not used, DFSORT/VSE uses getvis sorting if it is enabled. However, if the requested amount of GETVIS area is not available, DFSORT/VSE will not use getvis sorting.

If neither dataspace sorting nor getvis sorting is used, DFSORT/VSE uses partition program area sorting.

Getvis and dataspace sorting provide similar performance, and both allow you to override the 16 MB limit for the sorting area. Getvis sorting uses only partition virtual storage, and is preferable when an application should not depend on availability of virtual storage for data space. Dataspace sorting uses global system resources, and is preferable when DFSORT/VSE is called from an application which uses partition virtual storage extensively.

2.4.3.2 File Size and Virtual Storage

The relationship between file size and the amount of available virtual storage is critical to the performance of DFSORT/VSE. Basically, there are four separate cases to consider.

- When virtual storage is larger than the file, DFSORT/VSE may be able to perform the sort entirely within virtual storage, without the need to store intermediate data. This is called an incore sort. Indeed, this is the preferred method for sorting small files, since it minimizes I/O usage as well as CPU and elapsed time.

- When virtual storage is smaller than the file, work files are needed to store intermediate data. If provided sufficient virtual storage, DFSORT/VSE can perform an efficient sort, with elapsed and CPU times close to those of an incore sort. I/O usage is increased, however, reflecting the need to write intermediate data to work files.

- When virtual storage is reduced further or the file size is increased, DFSORT/VSE is forced to make less efficient use of work files. DFSORT/VSE does what it can to maintain performance, but it is forced to use work files less efficiently as the ratio of file size to available storage increases. The loss of efficiency has an adverse effect on elapsed time and I/O usage.

- When virtual storage is very small or the file size is very large, DFSORT/VSE may require several additional passes over the data to perform the sort. This phenomenon is known as intermediate merging, and results in significant performance degradation.

All other factors being equal, the range of file sizes that DFSORT/VSE can sort efficiently (or sort without requiring intermediate merging) grows roughly as the square of the virtual storage size. That is, doubling the virtual storage enables the application to handle files four times as large, with the same degree of efficiency. Likewise, halving the virtual storage causes the application to handle files only one-fourth as large with the same efficiency.

2.4.3.3 Virtual Storage and Sorting with Data Space or GETVIS Area

Dataspace sorting and getvis sorting have a different set of guidelines regarding virtual storage.

In the case of ESA supervisor mode, a data space is created (by SYSDEF command or job control statement) and enabled (by the DSPSIZE operand). Dataspace sorting uses a separate piece of virtual storage to hold the records currently being processed.

In any supervisor mode, the GETVIS area is enabled (by the GVSIZE operand) and getvis sorting uses a separate piece of virtual storage from the partition program area to hold the records currently being processed.

2.4.3.4 Recommendations for STORAGE Option

By using an appropriate value for the DFSORT/VSE installation STORAGE option, you can ensure that the majority of applications have sufficient virtual storage. If the IBM-supplied value for the STORAGE option is inappropriate for your site, you should change it to a more appropriate value. For specific applications, the installation values can be overridden using the STORAGE run-time option.

DFSORT/VSE needs a minimum of 32 KB of virtual storage but the minimum requirement for a given application can be more than 32 KB. You can determine in advance how much virtual storage is needed by submitting an ANALYZE control statement along with the other control statements for your application. DFSORT/VSE will analyze all the control statements, make optimization calculations, issue all the usual messages (including those specifying how many records can be sorted with the given configuration; how much more virtual storage should be allocated, and if the allocation was insufficient), and then terminate without sorting or merging.

If user exit routines are preloaded when DFSORT/VSE is program invoked, the size of any user exit routines to be used should not be included in the virtual storage calculated for DFSORT/VSE.

Some storage areas, for example, I/O buffers for non-VSAM files, are allocated in the program area. Therefore for these storage areas, you need to reserve a reasonable amount of virtual storage in your program area, with the installation STORAGE option, even when getvis or dataspace sorting is used.

2.4.4 Sorting with Data Space

Dataspace sorting is a DFSORT/VSE capability that uses data space available on ESA supervisor mode. DFSORT/VSE is shipped with dataspace sorting turned off. You must turn on dataspace sorting (with the DSPSIZE operand of ILUINST or the OPTION control statement) for DFSORT/VSE to take advantage of it. Dataspace sorting improves the elapsed time and CPU time performance of sort applications. Dataspace sorting increases the probability of sort applications not using work files.

Subtopics:

2.4.4.1 Benefits

A data space is a large contiguous area of virtual storage that is backed by processor or auxiliary storage, whichever is necessary, as determined by the system. The benefits of dataspace sorting are twofold:

- Due to the potentially large size of a data space, dataspace sorting enables a greater percentage of sort applications to be processed completely within virtual storage, without the need to write intermediate data to DASD. Incore sorts are the most efficient of all sorts; consequently, dataspace sorting can dramatically reduce CPU and elapsed times, as well as I/O and channel usage.

- If an incore sort is not possible, the additional virtual storage available enables DFSORT/VSE to sort more data before writing to work files. This provides savings in CPU and elapsed times.

2.4.4.2 Operation

The use of dataspace sorting is affected by the SYSDEF command or job control statement, which determines system default values for data spaces. In addition, dataspace sorting can be controlled with the DSPSIZE operand of the OPTION control statement. DSPSIZE can be specified as an installation-wide default on the ILUINST macro, which can also be overridden at run-time with an OPTION control statement.

Dataspace sorting is subject to the same paging rules as virtual storage, which can affect system paging. To minimize the impact on the paging subsystem, you must have sufficient real (processor) storage available to avoid a possible paging bottleneck.

When DSPSIZE=MAX is in effect, DFSORT/VSE monitors the system paging activity, and reduces its use of data space in the case of excessive paging on the system.

2.4.4.3 Size Limitations

The DSPSIZE operand specifies the maximum amount of data space to be used with dataspace sorting. The maximum size of a data space allocated by DFSORT/VSE is determined by the DSPSIZE value (either the installation default value, or an overriding value specified at run-time) and the DSIZE operand of the SYSDEF command or job control statement.

When you specify DSPSIZE=MAX, DFSORT/VSE dynamically determines the maximum amount of data space to be used for dataspace sorting. In this case, DFSORT/VSE bases its data space usage on the size of the available data space, amount of processor storage, and the paging activity of the system.

When you specify DSPSIZE=n, n cannot exceed the total amount of virtual storage that may be allocated by all data spaces (DSIZE operand value), or the current available data space at run-time. DFSORT/VSE decreases the value of n, if required, to avoid processor storage overcommitment. If n exceeds the amount of data space available at run-time, dataspace sorting is not used.

The amount of data space storage used for each sort application is displayed in message ILU340I. If the input file size is known, you should choose a DSPSIZE operand value large enough to improve the performance of the sort application.

2.4.4.4 Recommendations for Sorting with Data Space

The appropriate value for the DSPSIZE operand depends on the size of the file being sorted. For an incore sort, it is recommended that you specify a DSPSIZE value as large as the size of the file being sorted. The appropriate value to use for n is best determined by experimentation; start with a low value and keep raising it by a few megabytes until DFSORT/VSE achieves an incore sort.

Use of dataspace sorting should provide significant CPU and elapsed time performance improvements for sort applications. When your tuning efforts emphasize SIO counts over CPU and elapsed times, you may wish to turn off dataspace sorting by specifying DSPSIZE=0.

If a number of large applications that use dataspace sorting start at the same time on a system, resource contention for processor storage could become a problem. If processor storage is overcommitted, paging rates increase significantly. If this situation is likely to occur frequently at your site, it is recommended that you place further restrictions on the use of dataspace sorting.

2.4.5 Sorting with GETVIS Area

Getvis sorting is a DFSORT/VSE capability that uses partition GETVIS areas (24-bit and 31-bit). DFSORT/VSE is shipped with getvis sorting turned off. You must turn on getvis sorting (with the GVSIZE operand of ILUINST or the OPTION control statement) for DFSORT/VSE to take advantage of it. Getvis sorting improves the elapsed time and CPU time performance of sort applications. Getvis sorting will increase the probability that DFSORT/VSE will not use work files.

Subtopics:

2.4.5.1 Benefits

A GETVIS area is a contiguous area of partition virtual storage that is backed by processor or auxiliary storage, whichever is necessary, as determined by the system. The benefits of getvis sorting are twofold:

- Due to the potentially large size of a GETVIS area, getvis sorting enables a greater percentage of sort applications to be processed completely within virtual storage, without the need to write intermediate data to DASD. Incore sorts are the most efficient of all sort applications; consequently, getvis sorting can reduce CPU and elapsed times, as well as I/O and channel usage.

- If an incore sort is not possible, the additional virtual storage available still enables DFSORT/VSE to sort larger pieces of data at a time, before writing them to the work files. This provides a savings in CPU and elapsed times.

2.4.5.2 Operation

The use of getvis sorting is affected by the size of the partition GETVIS area, which depends on the partition size. In addition, getvis sorting can be controlled with the GVSIZE operand of the OPTION control statement. GVSIZE can be specified as an installation-wide default on the ILUINST macro, which can also be overridden at run time with an OPTION control statement.

Getvis sorting is subject to the same paging rules as virtual storage, which can affect system paging. To minimize the impact on the paging subsystem, you must have sufficient real (processor) storage available to avoid a possible paging bottleneck.

When GVSIZE=MAX is in effect, DFSORT/VSE monitors the system paging activity, and reduces its use of GETVIS area in the case of excessive paging on the system.

2.4.5.3 Size Limitations

The GVSIZE operand specifies the maximum amount of GETVIS area to be used with getvis sorting. The maximum size of a GETVIS area allocated by DFSORT/VSE is determined by the GVSIZE value (either the installation default value, or an overriding value specified at run time) and the GETVIS area size.

When you specify GVSIZE=MAX, DFSORT/VSE dynamically determines the maximum amount of GETVIS area that is to be used for getvis sorting. In this case, DFSORT/VSE bases its GETVIS area usage on the size of the available GETVIS area, the amount of reserved 24-bit GETVIS area (GVSRLOW operand) and 31-bit GETVIS area (GVSRANY operand) for the operating system and user application, the amount of processor storage, and the paging activity of the system. But make sure that the amount of GETVIS area reserved by GVSRLOW and GVSRANY operands is enough for VSAM buffers, servicing system requests, other requests of DFSORT/VSE, the DFSORT/VSE invoking program, and user exit routines (if any).

When you specify GVSIZE=n, n bytes of GETVIS area will be used for getvis sorting. You should make sure that other requests for GETVIS area can be serviced. DFSORT/VSE decreases the value of n, if required, to avoid processor storage overcommitment. If the requested n bytes of GETVIS area is not available, the program area will be used for sorting, and message ILU185I will be issued. If n=0, getvis sorting will not be used.

The amount of the GETVIS area used for each sort application is displayed in message ILU345I. If the input file size is known, you should execute the sort application in a partition large enough to improve the performance of the sorting application.

2.4.5.4 Recommendations for Sorting with GETVIS Area

The appropriate value for the GVSIZE operand depends on the size of the partition GETVIS area (the size of the partition GETVIS area = the partition size - the program area size). For an incore sort, it is recommended that you use a partition large enough to allow a GVSIZE area the size of the file being sorted. The appropriate value to use for GVSIZE is best determined by experimentation; start with a low value and keep raising it by a few megabytes until DFSORT/VSE achieves an incore sort.

Use of getvis sorting should provide significant CPU and elapsed time performance improvements for sort applications. When your tuning efforts emphasize SIO counts over CPU and elapsed times, you may wish to turn off getvis sorting by specifying GVSIZE=0.

If a number of large applications that use getvis sorting start at the same time on a system, resource contention for processor storage could become a problem. If processor storage is overcommitted, paging rates increase significantly. If this situation is likely to occur frequently at your site, it is recommended that you place further restrictions on the use of getvis sorting.

2.4.6 Input and Output Files

The performance of DFSORT/VSE can be affected by your choice of block sizes, devices for input and output files, and the use of user exit routines.

Choosing an efficient block size can improve space utilization and I/O performance.

Subtopics:

2.4.6.1 Space Utilization

The amount of space on a track or cylinder occupied by user data depends on the block size specified for the file. Grouping logical records into blocks reduces the amount of space needed to store data. Because fewer physical records are needed to store the same number of logical records, the amount of DASD space for gaps between records and for count and key areas is reduced.

Larger block sizes increase DASD space utilization when compared to files with a smaller block size.

2.4.6.2 I/O Performance

Although small block sizes permit more concurrent channel operations, they reduce the net data transfer rate (the actual amount of data transferred per second). This can impact the elapsed time of a DFSORT/VSE application performing a large amount of I/O. Small block size transfer also requires more CPU involvement and can therefore increase CPU time.

Large block sizes enable a higher net data transfer rate for input and output files and reduce the amount of processor time.

2.4.6.3 Recommendations

When selecting a block size for input or output files, consider these factors:

- Smaller file sizes generally result in less efficient use of DASD space.
- DFSORT/VSE applications that process files with small block sizes will generate higher SIO counts and probably consume more CPU time.

Select a block size as large as possible that is appropriate for the track capacity of the device you are using, adjusted to accommodate the size of the logical records in the file. Even higher utilization can be obtained by selecting optimum block sizes based on the attributes of your files.

For large files, files on tape, or files that you sort often, you might want to choose a larger block size. In general, the larger your input and output file block sizes, the better DFSORT/VSE performs. With FBA devices, a large CI (control interval) value gives better performance than a small one.

2.5 Chapter 9. Application Considerations

Most sites have many applications involving sorting. DFSORT/VSE is often used by these applications, either directly by invoking DFSORT/VSE with JCL or indirectly by invoking DFSORT/VSE from a program. In particular, the COBOL SORT statement and the PL/I sort routines result in calls to DFSORT/VSE. See the appropriate language books for complete details.

This chapter describes techniques for:

- Improving the performance of applications which use DFSORT/VSE.
- Improving your productivity in designing or modifying such applications.

In order to illustrate the various techniques available, the following methods for implementing a sorting application are described:

- Method 1. COBOL program with inline code
- Method 2. COBOL program with INPUT/OUTPUT PROCEDUREs
- Method 3. COBOL program and DFSORT/VSE control statements
- Method 4. DFSORT/VSE with control statements

Unlike the sample application, most real applications cannot be implemented using all of the methods described. However, an understanding of the methods available and their merits can help you select the best method for your application when you have a choice.

COBOL features, such as SORT statement INPUT/OUTPUT PROCEDUREs and DFSORT/VSE features, can be combined in more ways than just those listed above. For example, you can use a COBOL calling program with INPUT/OUTPUT PROCEDUREs and DFSORT/VSE control statements. Although it would be impractical to cover all of the possible combinations, the methods described here were chosen to cover each feature at least once. This should help you understand how to combine the features in different ways when necessary.

| If you are not familiar with any of the COBOL/VSE features described in
| this chapter, refer to *COBOL Programming Guide* for details. If you are
| not familiar with any of the DFSORT/VSE features described in this
| chapter, refer to *Application Programming Guide* for details. For excellent tutorials on DFSORT/VSE control statements, see *Getting Started with DFSORT/VSE*.

| **Note:** Although this chapter focuses on COBOL/VSE for purposes of illustration and performance comparisons, most of the techniques described
| can also be used with assembler and other program products that can call
| DFSORT/VSE such as DOS/VS COBOL, VS COBOL II, DOS PL/I Optimizing Compiler, and IBM PL/I for VSE/ESA (with Language Environment for VSE/ESA).

Subtopics:

2.5.1 Sample Sorting Application

The application described in this chapter does the following:

1. Preprocessing: Deletes all input records which have zero in a particular field (referred to as the OMIT field). The OMIT field is in columns 30-34 of the records.

2. Sorting: Sorts the remaining records in descending order by a particular zoned decimal format field (referred to as the KEY field). The KEY field is in columns 5-24 of the records.

3. Postprocessing: Writes one output record with each key.

| 2.5.2 COBOL/VSE Interfaces to DFSORT/VSE

Understanding the interfaces between DFSORT/VSE and languages such as
| COBOL/VSE can help you design more efficient applications. Knowing which interfaces are used enables you to:

- Obtain more information about these interfaces in the DFSORT/VSE books.
- Determine the best way to take advantage of these interfaces.

Subtopics:

2.5.2.1 Invoking DFSORT/VSE from COBOL

In order to invoke DFSORT/VSE from COBOL, you must code a COBOL SORT statement. This statement has the following variations which affect the way in which DFSORT/VSE and COBOL pass information back and forth:

- USING
- GIVING
- INPUT PROCEDURE
- OUTPUT PROCEDURE.

| In addition, the COBOL/VSE compile-time options FASTSRT and NOFASTSRT also affect the interfaces between COBOL and DFSORT/VSE.

The interfaces that result from the COBOL SORT statement and the FASTSRT/NOFASTSRT compile-time options are described in "Processing with FASTSRT" in topic 2.5.2.2 and "Processing with NOFASTSRT" in topic 2.5.2.3.

DFSORT/VSE can get the control statements either from a SYSIPT file or from a parameter list. The COBOL SORT or MERGE statement results in a call to DFSORT/VSE with a parameter list that contains the following DFSORT/VSE control statements: SORT or MERGE, RECORD, MODS, INPFIL, OUTFIL, OPTION. Optionally, COBOL allows you to pass information to DFSORT/VSE through control statements read from a SYSIPT file or contained in a VSE Librarian member.

You can specify that run-time control statements are to be read by using the SORT-CONTROL special register. You can assign to SORT-CONTROL the value "SYSIPT" (if control statements are to be read from a SYSIPT file) or the name of a VSE Librarian member. If you provide the name of a VSE Librarian member, the member must have a member type of "C", it must be cataloged in a sublibrary available at run-time, and the member name must not be IGZSRTCD.

The control statements you can provide at run-time are:

1. | DFSORT/VSE SORT or MERGE control statement (used to replace the SORT
   | or MERGE statement generated by the compiler).

2. SMS=nnnnn

   where nnnnn is the length, in bytes, of the most frequent record size (use only if the SD is variable). SMS=nnnnn is equivalent to operand LENGTH=(,,,,nnnnn) of the DFSORT/VSE RECORD control statement.

3. | DFSORT/VSE OPTION control statement except FILNM= operand.

4. | Other DFSORT/VSE control statements except RECORD, MODS, INPFIL, and
   | OUTFIL.

Control statements must be coded in the order listed above, and must be between columns 2 and 71. You can continue a statement by ending a line with a comma and starting the next line with a new keyword. Labels and remarks are not allowed in these statements.

2.5.2.2 Processing with FASTSRT

| COBOL/VSE's FASTSRT compile-time option is a special feature of the language that can improve performance for qualifying applications which use the COBOL SORT statement. You should always specify the FASTSRT
| option. COBOL/VSE decides automatically at compile-time whether FASTSRT can actually be used. For example, FASTSRT cannot be used for input
| processing when an INPUT PROCEDURE is specified. See *COBOL Programming*
| *Guide* for the complete list of FASTSRT requirements and restrictions.

| When FASTSRT is in effect for input processing, COBOL/VSE passes the name of the input file to DFSORT/VSE. DFSORT/VSE uses this file name to read the input file directly.

| When FASTSRT is in effect for output processing, COBOL/VSE passes the name of the output file to DFSORT/VSE. DFSORT/VSE uses this file name to write the output file directly.

Input or output processing by DFSORT/VSE rather than COBOL can result in reductions in CPU time, SIOs, and elapsed time.

Notes:

| 1. VS COBOL II also has the FASTSRT compile-time option. DOS/VS COBOL
| has no equivalent to FASTSRT.

2. PL/I's PLISRTA subroutine is equivalent to using FASTSRT for both input and output processing. PLISRTB is equivalent to using FASTSRT for output processing and PLISRTC is equivalent to using FASTSRT for input processing.

2.5.2.3 Processing with NOFASTSRT

When NOFASTSRT is in effect for input processing, the USING or INPUT PROCEDURE causes COBOL to generate a DFSORT/VSE E15 user exit routine and pass its address to DFSORT/VSE. When an INPUT PROCEDURE is used, COBOL incorporates the INPUT PROCEDURE code into the E15 routine it generates.

DFSORT/VSE does not read the input file directly, but instead obtains all the input records from the E15 routine. The E15 routine generated by COBOL reads the input file and passes one record to DFSORT/VSE each time it is called.

When NOFASTSRT is in effect for output processing, the GIVING or OUTPUT PROCEDURE causes COBOL to generate a DFSORT/VSE E35 user exit routine and pass its address to DFSORT/VSE. When an OUTPUT PROCEDURE is used, COBOL incorporates the OUTPUT PROCEDURE code into the E35 routine it generates.

DFSORT/VSE does not write the output file directly, but instead passes all the output records to the E35 routine. DFSORT/VSE calls the E35 routine generated by COBOL once for each record so the E35 routine can write the record to the output file.

Input or output processing by COBOL rather than DFSORT/VSE can result in degraded performance if installation option STXIT=YES is in effect. However, you can avoid STXIT-related performance problems by changing the installation option to STXIT=MIN or by passing DFSORT/VSE run-time option MINSTXIT. You can also turn off STXIT by passing DFSORT/VSE run-time option NOSTXIT but this is not recommended. See "Example 11. COBOL, DFSORT/VSE and Two-digit Year Date Example" in *Application Programming Guide* for information on how to pass the OPTION control statement to DFSORT/VSE.

Notes:

1. DOS/VS COBOL's input and output processing is equivalent to using NOFASTSRT.

2. PL/I's PLISRTD subroutine is equivalent to using NOFASTSRT for both input and output processing. PLISRTC is equivalent to using NOFASTSRT for output processing and PLISRTB is equivalent to using NOFASTSRT for input processing.

2.5.3 Method 1: COBOL Program with Inline Code

Method 1 takes the straightforward approach of doing preprocessing, sorting, and postprocessing using separate sequential inline blocks of code.

FASTSRT is in effect for input and output processing.

Subtopics:

2.5.3.1 Operation (FASTSRT in Effect)

The COBOL program does each operation inline.

The preprocessing code reads the input file and writes records with a nonzero OMIT field to the first temporary file.

The SORT statement results in a call to DFSORT/VSE with a parameter list that contains the following DFSORT/VSE control statements:

```
SORT    FIELDS=(0005,0020,CH,D),WORK=1
RECORD  TYPE=F,LENGTH=(000160,,000160)
INPFIL  BLKSIZE=00027840
OUTFIL  BLKSIZE=00027840
OPTION  FILNM=(TEMP2,TEMP1)
```

DFSORT/VSE reads the first temporary file, sorts it as requested by the SORT statement passed from the calling program, and writes it to the second temporary file.

The postprocessing code reads the second temporary file and writes one record with each key to the output file.

2.5.3.2 COBOL Calling Program

```
*-----------------------------------------------------------------
*  METHOD 1: INLINE COBOL PROCESSING
*-----------------------------------------------------------------
*  1. PRE-PROCESS:
*     THE PROGRAM DELETES RECORDS WITH A ZZZZZ OMIT FIELD BEFORE
*     SORTING. THE OMIT FIELD IS IN COLUMNS 30-34.
*  2. SORT
*     THE PROGRAM CALLS DFSORT/VSE TO SORT THE RECORDS IN
*     DESCENDING ORDER. THE KEY IS IN COLUMNS 5-24.
*  3. POST-PROCESS:
*     THE PROGRAM WRITES ONE RECORD WITH EACH KEY AFTER SORTING.
*  INPUT/OUTPUT: 1. READS INDS AND WRITES TO TEMP1.
*                2. DFSORT/VSE READS TEMP1 AND WRITES TEMP2.
*                3. READS FROM TEMP2 AND WRITES TO OUTDS.
*-----------------------------------------------------------------
 ID DIVISION.
```

```
 PROGRAM-ID. CASE1.
ENVIRONMENT DIVISION.
 INPUT-OUTPUT SECTION.
   FILE-CONTROL.
      SELECT INDS    ASSIGN TO INDS.
      SELECT TEMP1   ASSIGN TO TEMP1.
      SELECT TEMP2   ASSIGN TO TEMP2.
      SELECT OUTDS   ASSIGN TO OUTDS.
      SELECT SORT-FILE  ASSIGN TO SORTFIL.
DATA DIVISION.
  FILE SECTION.
   FD INDS RECORD CONTAINS 160 CHARACTERS
      LABEL RECORD STANDARD BLOCK 27840
      DATA RECORDS ARE INDS-RECORD.
   01 INDS-RECORD.
     05      FILLER        PIC X(4).
     05      INDS-KEY      PIC X(20).
     05      FILLER        PIC X(5).
     05      INDS-OMIT     PIC X(5).
     05      FILLER        PIC X(126).
    FD OUTDS RECORD CONTAINS 160 CHARACTERS
      LABEL RECORD STANDARD BLOCK 27840
      DATA RECORDS ARE OUTDS-RECORD.
   01 OUTDS-RECORD.
     05 FILLER          PIC X(160).
   FD TEMP1 RECORD CONTAINS 160 CHARACTERS
      LABEL RECORD STANDARD BLOCK 27840
      DATA RECORDS ARE TEMP1-RECORD.
   01 TEMP1-RECORD.
     05 FILLER          PIC X(160).
   FD TEMP2 RECORD CONTAINS 160 CHARACTERS
      LABEL RECORD STANDARD BLOCK 27840
      DATA RECORDS ARE TEMP2-RECORD.
   01 TEMP2-RECORD.
     05      FILLER        PIC X(4).
     05      TEMP2-KEY     PIC X(20).
     05      FILLER        PIC X(136).
   SD SORT-FILE RECORD CONTAINS 160 CHARACTERS
      DATA RECORD IS SORT-RECORD.
   01 SORT-RECORD.
     05       FILLER        PIC X(4).
     05       SORT-KEY      PIC X(20).
     05       FILLER        PIC X(136).
  WORKING-STORAGE SECTION.
   01 FLAGS.
    05 INDS-EOF     PIC X VALUE SPACE.
       88 SFLAG     VALUE "Y".
    05 TEMP-EOF     PIC X VALUE SPACE.
        88 TFLAG     VALUE "Y".
   01 PSTART     PIC 9(1) VALUE 0.
   01 SAVE-KEY   PIC X(20).
PROCEDURE DIVISION.
*-----------------------------------------------------------------
* PREPROCESSING:
*     READ INDS AND DELETE RECORDS WITH A 'ZZZZZ' OMIT FIELD.
*     WRITE THE REMAINING RECORDS TO TEMP1.
*-----------------------------------------------------------------
      OPEN INPUT INDS
      OPEN OUTPUT TEMP1
      READ INDS AT END SET SFLAG TO TRUE
      END-READ
      PERFORM UNTIL SFLAG
        IF INDS-OMIT NOT = "ZZZZZ"
          WRITE TEMP1-RECORD FROM INDS-RECORD
        END-IF
        READ INDS AT END SET SFLAG TO TRUE
        END-READ
      END-PERFORM
      CLOSE INDS TEMP1
*-----------------------------------------------------------------
* SORT:
*     CALL DFSORT/VSE TO SORT THE RECORDS FROM TEMP1 IN DESCENDING
*     ORDER AND WRITE THEM TO TEMP2.
```

```
      *-----------------------------------------------------------------
            SORT SORT-FILE
            ON DESCENDING KEY SORT-KEY
            USING TEMP1
            GIVING TEMP2.
            IF SORT-RETURN > 0
              DISPLAY "SORT FAILED"
            ELSE
      *-----------------------------------------------------------------
      * POSTPROCESSING:
      *     READ TEMP1 AND WRITE ONE RECORD WITH EACH KEY TO OUTDS.
      *-----------------------------------------------------------------
              OPEN INPUT TEMP2
              OPEN OUTPUT OUTDS
              READ TEMP2 AT END SET TFLAG TO TRUE
              END-READ
              PERFORM UNTIL TFLAG
                IF PSTART = 0
      *-----------------------------------------------------------------
      *     FIRST RECORD - SAVE KEY AND WRITE RECORD TO OUTDS.
      *-----------------------------------------------------------------
                  MOVE TEMP2-KEY TO SAVE-KEY
                  WRITE OUTDS-RECORD FROM TEMP2-RECORD
                  MOVE 1 TO PSTART
                ELSE
                  IF TEMP2-KEY NOT = SAVE-KEY
      *-----------------------------------------------------------------
      *     RECORD HAS NEW KEY - SAVE KEY AND WRITE RECORD TO OUTDS.
      *-----------------------------------------------------------------
                    MOVE TEMP2-KEY TO SAVE-KEY
                    WRITE OUTDS-RECORD FROM TEMP2-RECORD
                  END-IF
                END-IF
                READ TEMP2 AT END SET TFLAG TO TRUE
                END-READ
              END-PERFORM
              CLOSE OUTDS TEMP2
            END-IF
            STOP RUN.
```

Figure 35. COBOL Calling Program for Method 1

2.5.3.3 Performance

In all, three read/write passes are made over the data (one pass each for preprocessing, sorting, and postprocessing). The other methods we will describe make only a single pass over the data, providing significant performance improvements.

Using the same temporary file for the USING and GIVING would require less code, but would also result in NOFASTSRT being in effect for input, output, or both input and output processing (depending on the release of

| COBOL/VSE). Method 1's use of different temporary files for the USING and GIVING enables FASTSRT to be in effect for input and output processing, resulting in significant performance gains as shown in Figure 36.

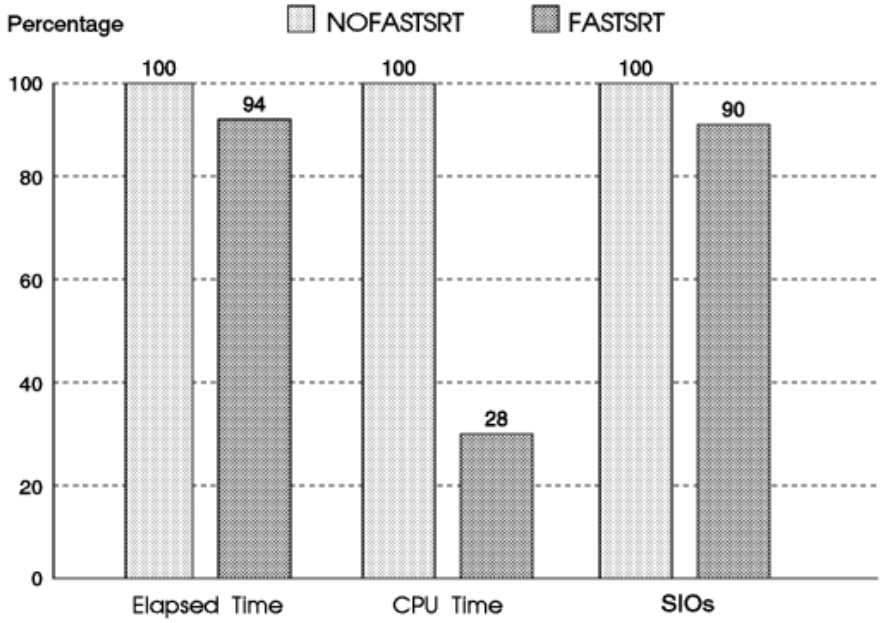


Figure 36. Benefits of FASTSRT for Method 1

2.5.4 Method 2: COBOL Program with INPUT/OUTPUT PROCEDUREs

Method 2 eliminates the inline preprocessing and postprocessing code by using a SORT statement INPUT PROCEDURE for preprocessing and a SORT statement OUTPUT PROCEDURE for postprocessing. Whereas inline code results in multiple passes over the data, INPUT/OUTPUT PROCEDUREs enable a single pass over the data.

NOFASTSRT is in effect for input and output processing due to the use of INPUT and OUTPUT PROCEDUREs.

An INPUT PROCEDURE or OUTPUT PROCEDURE can add, delete, alter, edit, or otherwise modify the records.

The INPUT PROCEDURE and OUTPUT PROCEDURE are actually special forms of E15 and E35 exits which are called by DFSORT/VSE during its processing, but controlled by the COBOL calling program.

The INPUT PROCEDURE is responsible for supplying all of the input records to be sorted to DFSORT/VSE, while the OUTPUT PROCEDURE is responsible for disposing of all the records after they are sorted.

Subtopics:

2.5.4.1 COBOL Calling Program

```
*-----------------------------------------------------------------
*  METHOD 2: COBOL INPUT AND OUTPUT PROCEDURES.
*-----------------------------------------------------------------
*  1. PRE-PROCESS:
*     THE PROGRAM USES A SORT INPUT PROCEDURE TO DELETE RECORDS
*     WITH A ZZZZZ OMIT FIELD BEFORE SORTING. THE OMIT FIELD IS
*     IN COLUMNS 30-34.
*  2. SORT
*     THE PROGRAM CALLS DFSORT/VSE TO SORT THE RECORDS IN DESCENDING
*     ORDER. THE KEY IS IN COLUMNS 5-24.
*  3. POST-PROCESS:
*     THE PROGRAM USES A SORT OUTPUT PROCEDURE TO WRITE ONE
*     ONE RECORD WITH EACH KEY AFTER SORTING.
*  INPUT/OUTPUT: READS INDS AND WRITES OUTDS.
*                DFSORT/VSE PASSES RECORDS TO THE PROCEDURES.
*-----------------------------------------------------------------
 ID DIVISION.
  PROGRAM-ID. CASE2.
 ENVIRONMENT DIVISION.
  INPUT-OUTPUT SECTION.
    FILE-CONTROL.
       SELECT INDS    ASSIGN TO INDS.
       SELECT OUTDS   ASSIGN TO OUTDS.
       SELECT SORT-FILE  ASSIGN TO SORTFIL.
 DATA DIVISION.
   FILE SECTION.
    FD INDS RECORD CONTAINS 160 CHARACTERS
       LABEL RECORD STANDARD BLOCK 27840
       DATA RECORDS ARE INDS-RECORD.
    01 INDS-RECORD.
      05      FILLER        PIC X(4).
      05      INDS-KEY      PIC X(20).
      05      FILLER        PIC X(5).
      05      INDS-OMIT     PIC X(5).
      05      FILLER        PIC X(126).
    FD OUTDS RECORD CONTAINS 160 CHARACTERS
       LABEL RECORD STANDARD BLOCK 27840
       DATA RECORDS ARE OUTDS-RECORD.
    01 OUTDS-RECORD.
      05 FILLER          PIC X(160).
    SD SORT-FILE RECORD CONTAINS 160 CHARACTERS
       DATA RECORD SORT-RECORD.
```

```
    01 SORT-RECORD.
      05       FILLER        PIC X(4).
      05       SORT-KEY      PIC X(20).
      05       FILLER        PIC X(136).
   WORKING-STORAGE SECTION.
    01 FLAGS.
      05 INDS-EOF     PIC X VALUE SPACE.
         88 SFLAG     VALUE "Y".
      05 TEMP-EOF     PIC X VALUE SPACE.
         88 TFLAG     VALUE "Y".
    01 PSTART     PIC 9(1) VALUE 0.
    01 SAVE-KEY   PIC X(20).
    01 TEMP-RECORD.
      05      FILLER        PIC X(4).
      05      TEMP-KEY     PIC X(20).
      05      FILLER        PIC X(136).
 PROCEDURE DIVISION.
   MASTER SECTION.
*-----------------------------------------------------------------
*     CALL DFSORT/VSE TO SORT THE RECORDS IN DESCENDING ORDER.
*-----------------------------------------------------------------
      SORT SORT-FILE
        ON DESCENDING KEY SORT-KEY
        INPUT PROCEDURE INPUT-PROC
        OUTPUT PROCEDURE OUTPUT-PROC.
        IF SORT-RETURN > 0
          DISPLAY "SORT FAILED".
      STOP RUN.
*-----------------------------------------------------------------
*     SORT INPUT PROCEDURE:
*       READ INDS.
*       DELETE ALL RECORDS WITH A "ZZZZZ" OMIT FIELD.
*       SEND ALL OTHER RECORDS TO DFSORT/VSE FOR SORTING.
*-----------------------------------------------------------------
   INPUT-PROC SECTION.
      OPEN INPUT INDS
      READ INDS AT END SET SFLAG TO TRUE
      END-READ
      PERFORM UNTIL SFLAG
        IF INDS-OMIT NOT = "ZZZZZ"
          RELEASE SORT-RECORD FROM INDS-RECORD
        END-IF
        READ INDS AT END SET SFLAG TO TRUE
        END-READ
      END-PERFORM.
      CLOSE INDS.
*-----------------------------------------------------------------
*     SORT OUTPUT PROCEDURE:
*       RECEIVE RECORDS FROM DFSORT/VSE INTO TEMP.
*       WRITE ONE RECORD WITH EACH KEY TO OUTDS.
*-----------------------------------------------------------------
   OUTPUT-PROC SECTION.
      OPEN OUTPUT OUTDS
      RETURN SORT-FILE INTO TEMP-RECORD AT END SET TFLAG TO TRUE
      END-RETURN
      PERFORM UNTIL TFLAG
        IF PSTART = 0
*-----------------------------------------------------------------
*     FIRST RECORD - SAVE KEY AND WRITE RECORD TO OUTDS.
*-----------------------------------------------------------------
          MOVE TEMP-KEY TO SAVE-KEY
          WRITE OUTDS-RECORD FROM TEMP-RECORD
          MOVE 1 TO PSTART
        ELSE
          IF TEMP-KEY NOT = SAVE-KEY
*-----------------------------------------------------------------
*     RECORD HAS NEW KEY - SAVE KEY AND WRITE RECORD TO OUTDS.
*-----------------------------------------------------------------
            MOVE TEMP-KEY TO SAVE-KEY
            WRITE OUTDS-RECORD FROM TEMP-RECORD
          END-IF
        END-IF
        RETURN SORT-FILE INTO TEMP-RECORD
```

```
            AT END SET TFLAG TO TRUE
          END-RETURN
        END-PERFORM.
        CLOSE OUTDS.
```

Figure 37. COBOL Calling Program for Method 2

2.5.4.2 Operation (NOFASTSRT in Effect)

The SORT statement results in a call to DFSORT/VSE with a parameter list that contains the following DFSORT/VSE control statements:

```
SORT    FIELDS=(0005,0020,CH,D),WORK=1
RECORD  TYPE=F,LENGTH=(000160,,)
INPFIL  EXIT
OUTFIL  EXIT
MODS    PH1=(,,E15),PH3=(,,E35)
```

DFSORT/VSE treats the INPUT PROCEDURE as an E15 user exit which must supply all the input records. DFSORT/VSE calls the INPUT PROCEDURE once for each input record. The INPUT PROCEDURE reads the input file and uses RELEASE to pass each record with a nonzero OMIT field to DFSORT/VSE.

DFSORT/VSE sorts the records passed to it by the INPUT PROCEDURE as requested by the SORT statement passed to it by the calling program.

DFSORT/VSE treats the OUTPUT PROCEDURE as an E35 user exit which must dispose of all the output records. DFSORT/VSE calls the OUTPUT PROCEDURE once for each sorted record. The OUTPUT PROCEDURE uses RETURN to obtain the records passed from DFSORT/VSE and writes one record with each key to the output file.

2.5.4.3 Performance

Recall that Method 1 made three read/write passes over the data. In contrast, Method 2 and all the Methods that follow only make one read/write pass over the data. The input file is read by the INPUT PROCEDURE and the output file is written by the OUTPUT PROCEDURE. DFSORT/VSE handles all other processing with just this single pass over the data.

Since the COBOL program's INPUT and OUTPUT PROCEDUREs must, by definition, read and write the files, NOFASTSRT is in effect for Method 2.

The overhead related to passing each record between DFSORT/VSE and the COBOL INPUT/OUTPUT PROCEDUREs results in an increase in CPU time compared to Method 1. However, the significant reduction in elapsed time and SIOs makes Method 2 preferable to Method 1. Figure 38 shows a performance comparison between Method 1 and Method 2.

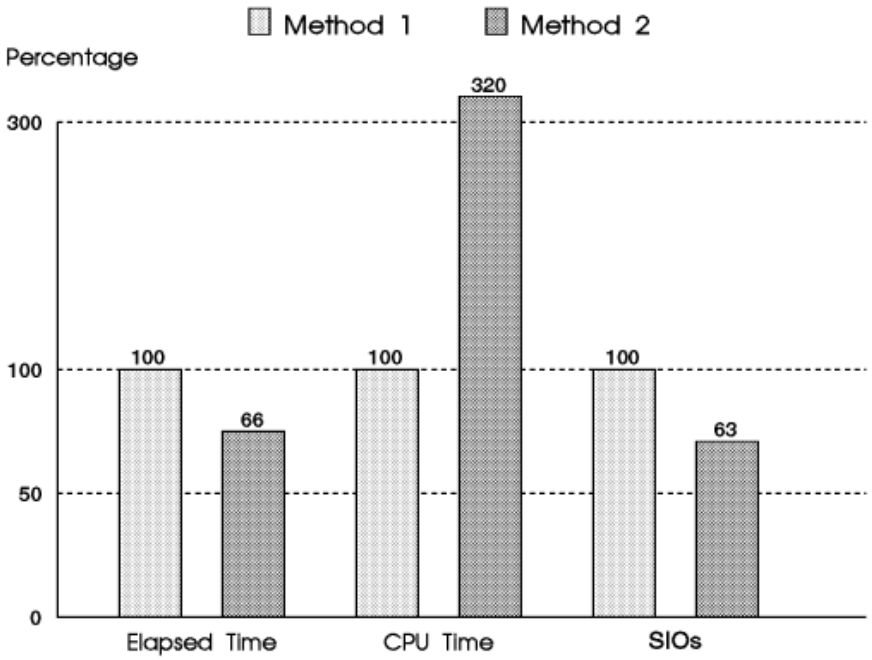


Figure 38. Method 1 vs Method 2 Performance Comparison

2.5.5 Method 3: COBOL Program and DFSORT/VSE Control Statements

Method 3 eliminates the COBOL preprocessing and postprocessing code by using DFSORT/VSE control statements to provide the equivalent functions. An OMIT statement is used instead of an E15, and a SUM statement is used instead of an E35.

FASTSRT is used for input and output processing.

DFSORT/VSE provides a powerful set of collating and editing functions available though the use of control statements and options. Collating and editing functions can be used to replace program code. They are designed to adapt to the run-time characteristics of an application in order to provide significant performance benefits.

DFSORT/VSE's most significant collating and editing functions are:

- | SORT or MERGE -- enable you to override the SORT or MERGE control
  | statement generated by the compiler. This override statement can
  | contain the DFSORT/VSE Year 2000 field format (Y2x).
- INCLUDE or OMIT -- enable you to include or delete records whose fields meet certain criteria.
- INREC -- enables you to delete and rearrange fields in your records, and to insert blanks, zeros and constants in records.
- OUTREC -- enables you to delete, rearrange, edit, and lookup and change fields in your records, and to insert blanks, zeros and constants in records.
- SUM -- enables you to sum fields in records with equal keys and to keep only one record with each key.
- SKIPREC and STOPAFT -- enable you to delete records at the beginning or end of your file.

| When a COBOL calling program is used, SORT or MERGE, INCLUDE or OMIT, INREC, OUTREC, SUM, and OPTION statements can be specified in the SYSIPT file by means of the SORT-CONTROL special register. SKIPREC and STOPAFT options can be specified on the OPTION statement.

Figure 39 in topic 2.5.5.1 and Figure 40 in topic 2.5.5.2 show the COBOL calling program and DFSORT/VSE control statements for Method 3.

Subtopics:

2.5.5.1 COBOL Calling Program

```
*-----------------------------------------------------------------
*  METHOD 3: MAIN COBOL PROGRAM.
*-----------------------------------------------------------------
*  1. PRE-PROCESS:
*     A DFSORT/VSE OMIT CONTROL STATEMENT DELETES RECORDS WITH
*     A 'ZZZZZ' OMIT FIELD BEFORE SORTING. THE OMIT FIELD IS IN
*     IN COLUMNS 30-34.
*  2. SORT
*     THE PROGRAM CALLS DFSORT/VSE TO SORT THE RECORDS IN
*     DESCENDING ORDER. THE KEY IS IN COLUMNS 5-24.
*  3. POST-PROCESS:
*     A DFSORT/VSE SUM CONTROL STATEMENT WRITES ONE RECORD WITH
*     EACH KEY AFTER SORTING.
*  INPUT/OUTPUT: DFSORT/VSE READS INPUT AND WRITES OUTPUT.
*-----------------------------------------------------------------
 ID DIVISION.
  PROGRAM-ID. CASE4.
 ENVIRONMENT DIVISION.
  INPUT-OUTPUT SECTION.
    FILE-CONTROL.
       SELECT SORTIN   ASSIGN TO SORTIN.
       SELECT SORTOUT ASSIGN TO SORTOUT.
       SELECT SORT-FILE  ASSIGN TO SORTFIL.
 DATA DIVISION.
   FILE SECTION.
    FD SORTIN  RECORD CONTAINS 160 CHARACTERS
       LABEL RECORD STANDARD BLOCK 27840
       DATA RECORDS ARE SORTIN-RECORD.
    01 SORTIN-RECORD.
      05 FILLER          PIC X(160).
    FD SORTOUT  RECORD CONTAINS 160 CHARACTERS
       LABEL RECORD STANDARD BLOCK 27840
       DATA RECORDS ARE SORTOUT-RECORD.
    01 SORTOUT-RECORD.
      05 FILLER          PIC X(160).
    SD SORT-FILE RECORD CONTAINS 160 CHARACTERS
       DATA RECORD SORT-RECORD.
    01 SORT-RECORD.
      05      FILLER        PIC X(4).
      05      SORT-KEY     PIC X(20).
      05      FILLER        PIC X(136).
   WORKING-STORAGE SECTION.
 PROCEDURE DIVISION.
   MASTER SECTION.
*-----------------------------------------------------------------
*     CALL DFSORT/VSE TO SORT THE RECORDS IN DESCENDING ORDER.
*-----------------------------------------------------------------
      MOVE "SYSIPT" TO SORT-CONTROL.
      SORT SORT-FILE
        ON DESCENDING KEY SORT-KEY
        USING SORTIN
        GIVING SORTOUT.
        IF SORT-RETURN > 0
          DISPLAY "SORT FAILED".
      STOP RUN.
```

Figure 39. COBOL Calling Program for Method 3

2.5.5.2 Control Statements on a SYSIPT file

```
OMIT COND=(30,5,CH,EQ,C'ZZZZZ')
SUM FIELDS=NONE
```

Figure 40. DFSORT/VSE Control Statements for Method 3

2.5.5.3 Operation (FASTSRT in Effect)

The SORT statement with SORT-CONTROL special register results in a call to DFSORT/VSE with a parameter list that contains the following DFSORT/VSE control statements:

```
SORT    FIELDS=(0005,0020,CH,D),WORK=1
RECORD  TYPE=F,LENGTH=(000160,,000160)
INPFIL  BLKSIZE=00027840
OUTFIL  BLKSIZE=00027840
OPTION  FILNM=(SORTOUT,SORTIN)
SUM     FIELDS=NONE
OMIT    COND=(30,5,CH,EQ,C'ZZZZZ')
```

DFSORT/VSE reads the input file and deletes each record with a zero OMIT field as requested by the OMIT statement.

DFSORT/VSE sorts the remaining records as requested by the SORT statement passed from the calling program.

DFSORT/VSE writes one record with each key to the output file as requested by the SUM statement.

2.5.5.4 Productivity

The use of DFSORT/VSE editing functions rather than COBOL code reduces considerably the effort required to perform the application. Source code for preprocessing and postprocessing is eliminated along with the time to compile and debug the code.

In addition, future changes to the editing functions performed by the application can be made by simply changing the control statements. Access to source code, and recompiles, are not necessary.

2.5.5.5 Performance

Since Method 3 uses DFSORT/VSE editing functions in place of Method 2's INPUT and OUTPUT PROCEDUREs, FASTSRT is in effect. As previously described, FASTSRT results in significant performance gains. In addition, by eliminating the overhead related to passing each record between DFSORT/VSE and the COBOL exits, and enabling DFSORT/VSE to use its highly optimized editing code, Method 3 achieves significant performance gains in CPU time, elapsed time, and SIOs compared to Method 2. Figure 41 shows a performance comparison between Method 2 and Method 3.

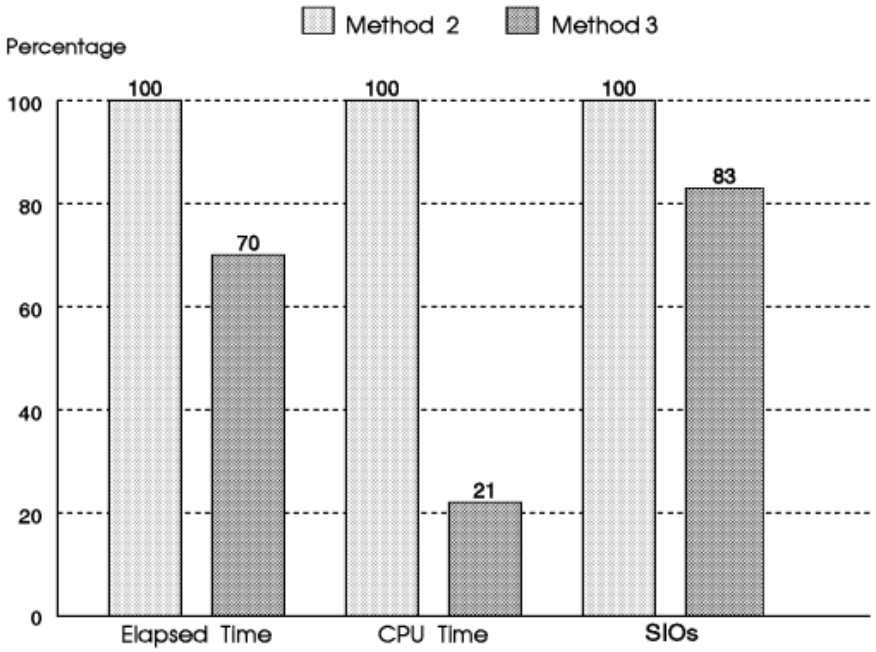


Figure 41. Method 2 vs Method 3 Performance Comparison

2.5.6 Method 4: DFSORT/VSE with Control Statements

Method 4 takes the final step of eliminating the COBOL calling program by invoking DFSORT/VSE directly (using SORT on the EXEC statement). Actually, for this application, the COBOL calling program could have been eliminated with Method 3 as well. Calling DFSORT/VSE directly is only feasible if all of the functions of the COBOL program can be duplicated using DFSORT/VSE control statements or user exits.

The SORT statement of the COBOL program is replaced by an equivalent DFSORT/VSE SORT control statement.

When DFSORT/VSE is invoked directly, control statements can be specified using the SYSIPT file (see Figure 42 in topic 2.5.6.1).

Subtopics:

2.5.6.1 Control Statements on a SYSIPT file

```
SORT FIELDS=(5,20,CH,D),WORK=1
RECORD  TYPE=F,LENGTH=(160,,160)
INPFIL  BLKSIZE=00027840
OUTFIL  BLKSIZE=00027840
OMIT COND=(30,5,CH,EQ,C'ZZZZZ')
SUM FIELDS=NONE
```

Figure 42. DFSORT/VSE Control Statements for Method 4

2.5.6.2 Operation

The system calls DFSORT/VSE directly.

DFSORT/VSE reads the input file and deletes each record with a zero OMIT field as requested by the OMIT statement. The RECORD control statement describes the format and length of the records being processed. The INPFIL control statement specifies the block size of the input file, it must be supplied because DFSORT/VSE does not support the BLKSIZE parameter of the DLBL job control statement.

DFSORT/VSE sorts the remaining records as requested by the SORT control statement.

DFSORT/VSE writes one record with each key to the output file as requested by the SUM statement. The OUTFIL control statement specifies the block size of the output file.

2.5.6.3 Productivity

Elimination of all COBOL code reduces significantly the effort required to perform an application. You can use control statements to duplicate complex code logic quickly and effectively. Source code for preprocessing, sorting, and postprocessing is eliminated along with the time to compile and debug the code.

In addition, future changes to the editing and sorting functions performed by the application can be made by simply changing the control statements. Access to source code, and recompiles, are not necessary.

2.5.6.4 Performance

Performance for Method 4 is comparable to that for Method 3. Since the COBOL program of Method 3 has no user exits or INPUT/OUTPUT PROCEDUREs and uses FASTSRT, the call to DFSORT/VSE is the only extra overhead it incurs. This overhead does not affect performance noticeably, except for applications that involve small data sets or minimal storage.

To put all of the techniques we have described in perspective, see Figure 43, which shows a performance comparison between Method 1 and Method 4. By replacing a COBOL application (Method 1) with a DFSORT/VSE application (Method 4), we have achieved truly significant performance improvements. Even the partial replacement techniques described (Methods 2 and 3) resulted in considerable performance gains. Be on the alert for opportunities to improve the performance of your own applications using the techniques described and their variations.

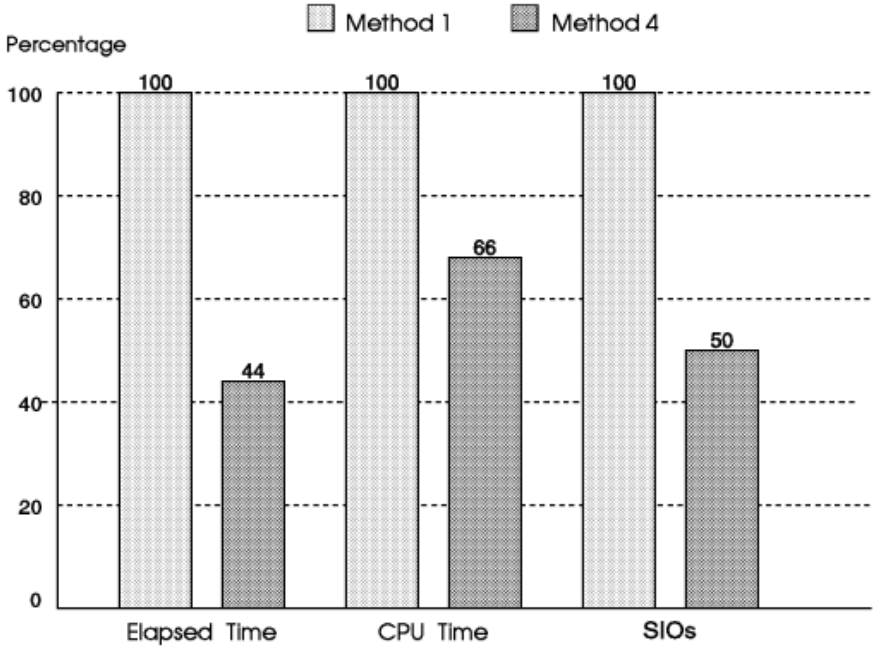


Figure 43. Method 1 vs Method 4 Performance Comparison

3.0 Part 3. Appendixes

A.0 Appendix A. Installation Messages

Messages are produced by the assembler during expansion of the ILUINST macro and during expansion of your message modules. These messages are printed as part of the assembler listing. Two types of messages are produced: *error messages* and *informational messages*.

The general format of DFSORT/VSE installation messages is:

```
 ___ For installation messages _____________________________________________
| s, text                                                                   |
|                                                                           |
|___________________________________________________________________________|
```

Where Represents

s
Severity code (4, 8 or 12)

text
Message text

Severity code defines the type of message:

4
Informational message

8
Error message

12
Error message

Subtopics:

- A.1 ILUINST Messages
- A.2 ILUMSGS Messages

A.1 ILUINST Messages

The following MNOTE messages may be produced during expansion of the ILUINST macro. A complete list of these messages is given below, along with their explanations. In each case, the entire ILUINST macro is analyzed even if errors are encountered. After you have corrected any errors indicated, you can rerun the job (see Figure 20 in topic 1.4.2).

4, DIAG CONFLICTS WITH PRINT OPTION, DIAG ACCEPTED

Explanation: PRINT=NONE or PRINT=CRITICAL was specified with DIAG; DIAG was accepted and PRINT=NONE or PRINT=CRITICAL was ignored.

4, DUMP CONFLICTS WITH PRINT OPTION, DUMP NOT ACCEPTED

Explanation: PRINT=NONE was specified with DUMP; PRINT=NONE was accepted and DUMP was ignored.

4, GVSRANY OPTION APPLICABLE ONLY WITH GVSIZE=MAX

Explanation: Option GVSRANY reserves 31-bit GETVIS area for system and invoking program use when GVSIZE=MAX is in effect.

4, GVSRLOW OPTION APPLICABLE ONLY WITH GVSIZE=MAX

Explanation: Option GVSRLOW reserves 24-bit GETVIS area for system and invoking program use when GVSIZE=MAX is in effect.

8, MACRO CALL HAS TOO MANY POSITIONAL PARAMETERS

Explanation: The maximum number (6) of positional parameters that ILUINST supports has been exceeded.

8, *x* IS AN INVALID POSITIONAL PARAMETER

Explanation: *x* is not a valid positional parameter. ILUINST only supports CHALT, DIAG, DUMP, EQUAL, ERASE and VERIFY as positional parameters.

8, ALTSEQ VALUE *x* INVALID

Explanation: The value *x* specified for ALTSEQ is not four digits or is not a valid pair of hexadecimal digits (*ffft*). The message appears once for each invalid value.

8, PRINT = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for PRINT is not a valid keyword.

8, ROUTE = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for ROUTE is not a valid keyword.

8, STORAGE = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for STORAGE is not a valid decimal number or is not a valid keyword.

8, SORTOUT = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for SORTOUT is not a valid logical unit number.

8, TOO MANY SORTIN PARAMETERS

Explanation: The maximum number (9) of logical unit numbers that can be specified for SORTIN has been exceeded.

8, SORTIN = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for SORTIN is not a valid logical unit number.

8, STXIT = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for STXIT is not a valid keyword.

8, GVSIZE = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for GVSIZE is not a valid decimal number or MAX, or is not in the valid range for GVSIZE.

8, GVSRANY = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for GVSRANY is not a valid decimal number or is not in the valid range for GVSRANY.

8, GVSRLOW = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for GVSRLOW is not a valid decimal number or is not in the valid range for GVSRLOW.

8, DSPSIZE = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for DSPSIZE is not a valid decimal number or MAX, or is not in the valid range for DSPSIZE.

8, WRKSEC = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for WRKSEC is not a valid keyword.

8, FMS = *x* **INVALID SPECIFICATION**

Explanation: The value *x* specified for FMS is not a valid keyword.

8, LOCALE VALUE *x* **GREATER THAN 32 CHARACTERS**

Explanation: The value *x* specified for the LOCALE name exceeds the maximum length (32) that can be specified.

```
8, Y2PAST = x INVALID SPECIFICATION
```

Explanation: The value *x* specified for Y2PAST is not a valid decimal number or is not in the valid range for Y2PAST.

```
| 8, DIAGINF VALUE x GREATER THAN 8 CHARACTERS
```

| **Explanation:** The value *x* specified for DIAGINF is too long.

```
| 8, NRECOUT = x INVALID SPECIFICATION
```

| **Explanation:** The value *x* specified for NRECOUT is not a valid keyword.

```
| 8, VSAMBSP = x INVALID SPECIFICATION
```

| **Explanation:** The value *x* specified for VSAMBSP is not a valid keyword.

```
| 8, ZDPRINT = x INVALID SPECIFICATION
```

| **Explanation:** The value *x* specified for ZDPRINT is not a valid keyword.

A.2 ILUMSGS Messages

The following MNOTE messages may be produced during expansion of the ILUMSGS macro. A complete list of these messages is given below, along with their explanations. After you have corrected any errors indicated, you can rerun the job (see Figure 24 in topic 1.4.3.2).

12, * TEXT FOR MESSAGE *n* IS NOT PRESENT IN ILUMSGS**

Explanation: The message text requested does not appear in the ILUMSGS macro; that is, a statement of the form:

```
ILUMSET n,'textn...'
```

was not found in your copy of the ILUMSGS macro.

12, * MESSAGE *n* TEXT LENGTH IS *m* BYTES TOO LONG**

Explanation: The message text requested is *m* bytes too long as it appears in your copy of the ILUMSGS macro.

12, * TEXT FOR MESSAGE *n* HAS INCORRECT INSERT FIELD(S)**

Explanation: The message text requested has incorrect insert field(s) as it appears in your copy of the ILUMSGS macro. That is, an insert field character (?, %, or !) was deleted, added, or changed, or an insert field or insert field identifier was added, deleted or duplicated.

12, * TEXT FOR MESSAGE *n* HAS UNMATCHED SHIFTOUT/SHIFTIN PAIR**

Explanation: The message text requested has unmatched shift-out and shift-in control characters as it appears in your copy of the ILUMSGS macro. Shift-out and shift-in control characters must be paired with zero or an even number of intervening bytes. Nested shift-out or shift-in control characters are not allowed. All characters between shift-out and shift-in must be valid double-byte characters.

B.0 Appendix B. Locales Supplied with C Run-Time Library

| The following table lists the locales supplied with the LE/VSE 1.4 C
| Run-time library or VSE/ESA 2.3 C Run-time library. Consult your system programmer to determine whether they have been installed at your site.

For details regarding locales and their customization, see *LE/VSE Library*. The table of supported locales has been reproduced here for your convenience.

Throughout this book, references to LE/VSE also apply to the VSE C
| Language Run-Time Support feature of VSE/ESA Version 2 Release 3.

Figure 44. Compiled Locales Supplied with LE/VSE C

Locale name	Language	Country	Codeset	Phase name
Bg_BG.IBM-1025	Bulgarian	Bulgaria	IBM-1025	EDC$BGFE
Cs_CZ.IBM-870	Czech	Czechoslovaki	IBM-870	EDC$CZEQ
Da_DK.IBM-277	Danish	Denmark	IBM-277	EDC$DAEE
Da_DK.IBM-1047	Danish	Denmark	IBM-1047	EDC$DAEY
De_CH.IBM-500	German	Switzerland	IBM-500	EDC$DCEO
De_CH.IBM-1047	German	Switzerland	IBM-1047	EDC$DCEY
De_DE.IBM-273	German	Germany	IBM-273	EDC$DDEB
De_DE.IBM-1047	German	Germany	IBM-1047	EDC$DDEY
El_GR.IBM-875	Ellinika	Greece	IBM-875	EDC$ELES
En_GB.IBM-285	English	United Kingdom	IBM-285	EDC$EKEK
En_GB.IBM-1047	English	United Kingdom	IBM-1047	EDC$EKEY
En_JP.IBM-1027	English	Japan	IBM-1027	EDC$EJEX
En_US.IBM-037	English	United States	IBM-037	EDC$EUEA
En_US.IBM-1047	English	United States	IBM-1047	EDC$EUEY
Es_ES.IBM-284	Spanish	Spain	IBM-284	EDC$ESEJ
Es_ES.IBM-1047	Spanish	Spain	IBM-1047	EDC$ESEY
Et_EE.IBM-1122	Estonian	Estonia	IBM-1122	EDC$EEFD
Fi_FI.IBM-278	Finnish	Finland	IBM-278	EDC$FIEF
Fi_FI.IBM-1047	Finnish	Finland	IBM-1047	EDC$FIEY
Fr_BE.IBM-500	French	Belgium	IBM-500	EDC$FBEO
Fr_BE.IBM-1047	French	Belgium	IBM-1047	EDC$FBEY
Fr_CA.IBM-037	French	Canada	IBM-037	EDC$FCEA

Fr_CA.IBM-1047	French	Canada	IBM-1047	EDC$FCEY
Fr_FR.IBM-297	French	France	IBM-297	EDC$FFEM
Fr_FR.IBM-1047	French	France	IBM-1047	EDC$FFEY
Fr_CH.IBM-500	French	Switzerland	IBM-500	EDC$FSEO
Fr_CH.IBM-1047	French	Switzerland	IBM-1047	EDC$FSEY
Hr_HR.IBM-870	Croatian	Croatia	IBM-870	EDC$HREQ
Hu_HU.IBM-870	Hungarian	Hungary	IBM-870	EDC$HUEQ
Is_IS.IBM-871	Iceland	Iceland	IBM-871	EDC$ISER
Is_IS.IBM-1047	Iceland	Iceland	IBM-1047	EDC$ISEY
It_IT.IBM-280	Italian	Italy	IBM-280	EDC$ITEG
It_IT.IBM-1047	Italian	Italy	IBM-1047	EDC$ITEY
Iw_IL.IBM-424	Israeli	Israel	IBM-424	EDC$ILFB
Ja_JP.IBM-290	Japanese	Japan	IBM-290	EDC$JAEL
Ja_JP.IBM-930	Japanese	Japan	IBM-930	EDC$JAEU
Ja_JP.IBM-939	Japanese	Japan	IBM-939	EDC$JAEV
Ja_JP.IBM-1027	Japanese	Japan	IBM-1027	EDC$JAEX
Ko_KR.IBM-933	Korean	Korea	IBM-933	EDC$KRGZ
Lt_LT.IBM-1112	Lithuanian	Lithuania	IBM-1112	EDC$LTGD
Mk_MK.IBM-1025	Macedonian	Macedonia	IBM-1025	EDC$MMFE
Nl_BE.IBM-500	Dutch	Belgium	IBM-500	EDC$NBEO
Nl_BE.IBM-1047	Dutch	Belgium	IBM-1047	EDC$NBEY
Nl_NL.IBM-037	Dutch	Netherlands	IBM-037	EDC$NNEA
Nl_NL.IBM-1047	Dutch	Netherlands	IBM-1047	EDC$NNEY
No_NO.IBM-277	Norwegian	Norway	IBM-277	EDC$NOEE
No_NO.IBM-1047	Norwegian	Norway	IBM-1047	EDC$NOEY
Pl_PL.IBM-870	Polish	Poland	IBM-870	EDC$PLEQ
Pt_BR.IBM-037	Portugese	Brazil	IBM-037	EDC$BREA
Pt_BR.IBM-1047	Portugese	Brazil	IBM-1047	EDC$BREY
Pt_PT.IBM-037	Portugese	Portugal	IBM-037	EDC$PTEA

Pt_PT.IBM-1047	Portugese	Portugal	IBM-1047	EDC$PTEY
Ro_RO.IBM-870	Romanian	Romania	IBM-870	EDC$ROEQ
Ru_RU.IBM-1025	Russian	Russia	IBM-1025	EDC$RUFE
Sh_SP.IBM-870	Serbian (Latin alphabet)	Serbia	IBM-870	EDC$SLEQ
Si_SI.IBM-870	Slovenian	Slovenia	IBM-870	EDC$SIEQ
Sk_SK.IBM-870	Slovakian	Slovakia	IBM-870	EDC$SKEQ
Sq_AL.IBM-1047	Albanian	Albania	IBM-1047	EDC$SAEY
Sr_SP.IBM-1025	Serbian (Cyrillic alphabet)	Serbia	IBM-1025	EDC$SCFE
Sv_SE.IBM-278	Swedish	Sweden	IBM-278	EDC$SVEF
Sv_SE.IBM-1047	Swedish	Sweden	IBM-1047	EDC$SVEY
Th_TH.IBM-838	Thai	Thailand	IBM-838	EDC$THEP
Tr_TR.IBM-1026	Turkish	Turkey	IBM-1026	EDC$TREW
Zh_CN.IBM-935	Chinese (simplified)	China	IBM-935	EDC$ZCGY
Zh_TW.IBM-937	Chinese (traditional)	Taiwan	IBM-937	EDC$ZTGW

C.0 Appendix C. The ILUMSGS Macro

The following figure is a copy of the ILUMSGS macro. You can modify the message text in this macro if you want DFSORT/VSE to issue messages in a language other than English, or if you want to change the wording of some of the messages. See "Modifying ILUMSGS Macro Message Text" in topic 1.4.3 for more information.

| **Note:** If you modify the message texts in ILUMSGS, you need to modify the
| corresponding message texts in the online message file. See "Modifying
| DFSORT/VSE Online Message Explanations" in topic 1.3.3 for more
| information.

```
|            MACRO                                                        00050000
|            ILUMSGS                                                      00100000
|     .*************************************************************** / 00150000
|     .**PROPRIETARY V3 STATEMENT                                      */ 00200000
|     .**LICENSED MATERIALS - PROPERTY OF IBM                           */ 00250000
|     .**"RESTRICTED MATERIALS OF IBM"                                 */ 00300000
|     .**5746-SM3                                                      */ 00350000
|     .**(C) COPYRIGHT 1995,1998  IBM CORP.                            */ 00400000
|     .**END PROPRIETARY V3 STATEMENT                                  */ 00450000
|     .**                                                              */ 00625000
|     .*************************************************************** / 00650000
|     .*  $MAC(ILUMSGS) COMP(5746-SM3):                                 * 00700000
|     .*  $4HKF3=RAS,SRTV34A,98/01/22,MTAANSM: FINAL TUNING OF R34  @4HKF3A*/ 00704100
|     .*  $4CDPI=LNF,SRTV34A,97/07/15,MTAANTG: PROVIDE CDPI         @4CDPIA * 00708200
|     .*  $4HKF3=LNF,SRTV34A,98/01/18,MTAANSM: MISC                 @4HKF3A*/ 00712300
|     .*  $4OUTR=LNF,SRTV34A,97/08/08,MTAALSD: OUTREC ENHANCEMENTS  @4OUTRA*/ 00716400
|     .*  $4NOUT=LNF,SRTV34A,97/11/15,MTAAKVV: NRECOUT SUPPORT      @4NOUTA*/ 00720500
|     .*  $4IOWK=LNF,SRTV34A,97/10/01,MTAAKVV: WORK INPUT/OUTPUT    @4IOWKA*/ 00724600
|     .*  $4IOXS=LNF,SRTV34A,97/10/01,MTAAKVV: EXTENDED SORTING     @4IOXSA*/ 00728700
|     .*  $4IORT=LNF,SRTV34A,97/10/01,MTAAKVV: NON-VSAM INPUT/OUTPUT@4IORTA*/ 00732800
|     .*  $4IOSO=LNF,SRTV34A,97/10/01,MTAAKVV: VLR SAM OUTPUT       @4IOSOA*/ 00736900
|     .*  $4IOVS=LNF,SRTV34A,97/10/01,MTAAKVV: VSAM INPUT/OUTPUT    @4IOVSA*/ 00741000
|     .*  $4STXT=LNF,SRTV34A,97/10/01,MTAAKVV: STXIT IMPROVEMENTS   @4STXTA*/ 00745100
|     .*                                                               * 00750000
|     .*************************************************************** 00800000
|     .*                                                               * 00850000
|     .* FUNCTION:  CONTAINS ALL THE MESSAGE TEXT FOR MESSAGES ISSUED  * 00900000
|     .*             BY THE STANDARD TECHNIQUES. EACH MESSAGE TEXT     * 00950000
|     .*             CONSISTS OF FIXED CHARACTERS AND INSERT FIELDS.  THE * 01000000
|     .*             INSERT FIELDS SUPPLY VARIABLE INFORMATION IN THE  * 01050000
|     .*             FOLLOWING FORMS:                                  * 01100000
|     .*                   ----?----                                   * 01115300
|     .*                   |---%---|---N---                            * 01130600
|     .*                   ----!---                                    * 01145900
|     .*                   WHERE                                       * 01161200
|     .*                   1.  ?   REPRESENTS CHARACTER DATA (ONE ? FOR EACH * 01176500
|     .*                           CHARACTER).                         * 01191800
|     .*                   2.  %   REPRESENTS FIXED LENGTH NUMERIC DATA * 01207100
|     .*                           (ONE % FOR NUMBER).                 * 01222400
|     .*                   3.  !   REPRESENTS HEXADECIMAL INFORMATION (ONE ! * 01237700
|     .*                           FOR EACH CHARACTER).                * 01253000
|     .*                   4.  N   IS AN INSERT CODE (1, 2, ... 16) WHICH * 01268300
|     .*                           UNIQUELY IDENTIFIES AN INSERT FIELD. * 01283600
|     .*                                                               * 01300000
|     .* NOTES:                                                        * 01350000
|     .*                                                               * 01400000
|     .*   DEPENDENCIES: THE FOLLOWING MODULES MUST BE RE-ASSEMBLED AND * 01450000
|     .*                 RE-LINKEDITED TO APPLY CHANGES MADE TO THIS   * 01500000
|     .*                 MACRO:                                        * 01550000
|     .*                 - ILUMSGA  FOR MODIFYING DFSORT/VSE MESSAGES  * 01570000
|     .*                            (000-599, 700-999).                * 01590000
|     .*                 - ILUMUTL  FOR MODIFYING ICETOOL MESSAGES (600-699)* 01610000
|     .*                                                               * 01630000
|     .*                                                               * 01650000
|     .*   RESTRICTIONS: THE FOLLOWING RULES APPLY WHEN MODIFYING MESSAGES: * 01700000
|     .*                                                               * 02750000
|     .*       1.  AN AMPERSAND (&) MUST BE CODED AS TWO AMPERSANDS,   * 02766600
|     .*           FOLLOWING THE ASSEMBLER CODING CONVENTION.          * 02783200
|     .*                                                               * 02800000
|     .*       2.  AN APOSTROPHE (') MUST BE CODED AS TWO APOSTROPHES, * 02801000
|     .*           FOLLOWING THE ASSEMBLER CODING CONVENTION.          * 02802000
|     .*                                                               * 02803000
|     .*       3.  DOUBLE-BYTE DATA MAY BE USED IN A TEXT STRING.  THE * 02804000
|     .*           START OF DOUBLE-BYTE DATA IS DELIMITED BY THE       * 02805000
|     .*           SHIFT-OUT CONTROL CHARACTER (X'0E'), AND THE END BY * 02806000
|     .*           THE SHIFT-IN CONTROL CHARACTER (X'0F').  SHIFT-OUT  * 02807000
|     .*           AND SHIFT-IN CONTROL CHARACTERS MUST BE PAIRED.     * 02808000
|     .*           EITHER NO BYTES OR AN EVEN NUMBER OF BYTES MUST     * 02809000
|     .*           APPEAR BETWEEN PAIRED SHIFT-OUT AND SHIFT-IN CONTROL * 02810000
|     .*           CHARACTERS.                                         * 02811000
|     .*                                                               * 02812000
|     .*       4.  MESSAGE MACROS MUST NOT BE ADDED OR DELETED.        * 02813000
```

```
|    .*                                                                * 02814000
|    .*       5.  MACRO CALL NAMES (ILUMSET) AND MESSAGE NUMBERS MUST  * 02815000
|    .*           NOT BE CHANGED.                                      * 02816000
|    .*                                                                * 02817000
|    .*       6.  FIXED CHARACTERS CAN BE CHANGED, ADDED, DELETED, OR  * 02818000
|    .*           RELOCATED.                                           * 02819000
|    .*                                                                * 02820000
|    .*       7.  INSERT FIELD CHARACTERS (?N, %N, !N) MUST NOT BE     * 02821000
|    .*           CHANGED, ADDED, OR DELETED.                          * 02822000
|    .*                                                                * 02823000
|    .*       8.  INSERT FIELDS MUST NOT BE MADE CONTIGUOUS.           * 02824000
|    .*                                                                * 02825000
|    .*       9.  INSERT FIELDS CAN BE RELOCATED.                      * 02826000
|    .*                                                                * 02827000
|    .*      10.  EACH TEXT STRING MUST NOT EXCEED 236 BYTES.          * 02828000
|    .*                                                                * 02829000
|    .*      11.  YOU CAN REQUEST THAT THE MESSAGE TEXT BE PRINTED ON  * 02830000
|    .*           SEVERAL CONSECUTIVE LINES (MESSAGE LINES).  USE THE  * 02831000
|    .*           SEMICOLON (;) TO INDICATE THE END OF A MESSAGE LINE. * 02832000
|    .*           THE SEMICOLON WILL NOT BE PRINTED.                   * 02833000
|    .*                                                                * 02834000
|    .*      12.  EACH MESSAGE LINE MUST NOT EXCEED 109 BYTES.         * 02835000
|    .*                                                                * 02836000
|    .*      13.  BECAUSE THE CHARACTERS ?, %, AND !  HAVE SPECIAL     * 02837000
|    .*           MEANINGS, YOU MUST NOT USE THEM AS TEXT CHARACTERS.  * 02838000
|    .*                                                                * 02839000
|    .*      14.  THE SPECIAL SYMBOLS ?N, %N, !N, AND THE SEMICOLON (;)* 02840000
|    .*           MUST NOT APPEAR BETWEEN SHIFT-OUT AND SHIFT-IN       * 02841000
|    .*           CONTROL CHARACTERS. OTHERWISE THEY WILL NOT BE       * 02841600
|    .*           PROCESSED CORRECTLY AS INSERT FIELDS.                * 02842200
|    .*                                                                * 02843000
|    .*      15.  IF CONTINUATION IS NECESSARY, USE STANDARD MACRO     * 02844000
|    .*           STATEMENT CONTINUATION RULES.                        * 02845000
|    .*                                                                * 02850000
|    .**************************************************************** 02900000
|             ILUMSET 0,'--- CONTROL STATEMENTS/MESSAGES --- DFSORT/VSE 5746X02900200
|                   -SM3 RELEASE 4.0  DATE ??1/??2/????3'              02900490
|    .*       ??1   - MM  (MONTH)                                      02900600
|    .*       ??2   - DD  (DAY)                                        02900800
|    .*       ????3 - YYYY  (YEAR)                                     02901000
|             ILUMSET 1,'INSUFFICIENT STORAGE'                         02901200
|             ILUMSET 2,'????????1 HAS WRONG SVA STATUS'               02901400
|             ILUMSET 10,'INSUFFICIENT STORAGE, ADD 4K'                02901600
|             ILUMSET 11,'INSUFFICIENT STORAGE, %1K AVAILABLE, ADD %2K, MODUX02901800
|                   LES ARE NOT IN SVA'                               02902000
|             ILUMSET 12,'MODULE STATUS: PARTITION (POSSIBLE PERFORMANCE DEGX02902290
|                   RADATION)'                                        02902400
|             ILUMSET 13,'RECORD LENGTH INVALID FOR ????1'             02902600
|             ILUMSET 20,'STORAGE USED = %1 BYTES'                     02902800
|             ILUMSET 21,'FIXABLE STORAGE = %1 BYTES'                  02903000
|             ILUMSET 25,'TOO MANY FIELDS'                             02903200
|             ILUMSET 29,'ANALYZE END'                                 02903400
|             ILUMSET 30,'INSUFFICIENT STORAGE, %1K AVAILABLE, ADD %2K, MODUX02903600
|                   LES ARE IN SVA'                                   02903800
|             ILUMSET 31,'OVERLAPPING WORK EXTENTS ON SYS(%1) AND SYS(%2)' 02904000
|             ILUMSET 32,'FBA SORT WORK AREA IS LARGER THAN 2000 MB'    02904200
|             ILUMSET 33,'FBA SORT WORK EXTENT IGNORED, SMALLER THAN 32KB' 02904400
|             ILUMSET 34,'MODAL RECORD LENGTH ASSUMED = %1 BYTES'       02904600
|             ILUMSET 37,'INSUFFICIENT PARTITION PROGRAM AREA: %1K AVAILABLEX02904850
|                   , ADD %2K'                                        02904900
|             ILUMSET 38,'INSUFFICIENT ????????????1 %2K AVAILABLE, ADD %3K' 02905000
|             ILUMSET 39,'DFSORT/VSE MODULES ARE NOT IN SVA'           02905200
|             ILUMSET 40,'DFSORT/VSE MODULES ARE IN SVA'               02905400
|             ILUMSET 41,'INSUFFICIENT GETVIS AREA'                    02905500
|             ILUMSET 101,'SUM FIELD %1, ????????2 INVALID'            02905600
|             ILUMSET 102,'FIELD VALUE EXCEEDS MAXIMUM ALLOWABLE CHARACTERS X02905800
|                   - ????????1'                                      02906000
|             ILUMSET 103,'MULTIPLY DEFINED EXIT E??1'                 02906200
|             ILUMSET 104,'NO ??????1 STATEMENT'                       02906400
|             ILUMSET 105,'STATEMENT DEFINER ERROR'                    02906600
|             ILUMSET 106,'DUPLICATE ????????1 STATEMENT'              02906800
|             ILUMSET 107,'COLUMN 1 OR 1-15 NOT BLANK'                 02907000
|             ILUMSET 108,'COLUMN 2-16 BLANK IN CONTINUATION STATEMENT' 02907200
```

```
|             ILUMSET 110,'TOO MANY ????????1 VALUES'                          02907400
|             ILUMSET 111,'INVALID ????????1 KEYWORD'                          02907600
|             ILUMSET 112,'INVALID FORMAT'                                     02907800
|             ILUMSET 113,'CONTROL FIELD %1 DISPLACEMENT INVALID'              02908000
|             ILUMSET 114,'CONTROL FIELD %1 LENGTH INVALID'                    02908200
|             ILUMSET 115,'UNIT ASSIGN ERROR ????????1 SYS(???2)'             02908400
|             ILUMSET 116,'CONTROL FIELD %1 SEQUENCE INVALID'                  02908600
|             ILUMSET 117,'BOTH SORT AND MERGE DEFINED'                        02908800
|             ILUMSET 118,'??????1 ??????2 OPERAND MISSING OR INVALID'        02909000
|             ILUMSET 119,'BLANK STATEMENT OR NO OPERAND ENCOUNTERED'         02909200
|             ILUMSET 120,'GIVEN FILE SIZE INVALID'                           02909400
|             ILUMSET 121,'FILES VALUE INVALID'                               02909600
|             ILUMSET 122,'????????1 OPTION HAS INVALID PARAMETER'            02909800
|             ILUMSET 123,'SORT WORK VALUE INVALID'                           02910000
|             ILUMSET 124,'INVALID DATA TYPE'                                 02910200
|             ILUMSET 125,'?????????1 OPERAND IGNORED'                        02910400
|             ILUMSET 126,'INVALID ???1 NAME'                                 02910600
|             ILUMSET 127,'INVALID MODS ADDRESS OR LENGTH FIELD'              02910800
|             ILUMSET 128,'INVALID ???1 EXIT'                                 02911000
|             ILUMSET 129,'ERROR IN LENGTH VALUE'                             02911200
|             ILUMSET 130,'BOTH INCLUDE AND OMIT DEFINED'                     02911400
|             ILUMSET 131,'RECORD TYPE INVALID'                               02911600
|             ILUMSET 132,'ALTSEQ STATEMENT HAS INVALID DATA'                 02911800
|             ILUMSET 133,'SUM FORMAT INVALID'                                02912000
|             ILUMSET 134,'VIRT PARAMETER IGNORED'                            02912200
|             ILUMSET 135,'VOLUME VALUE INVALID'                              02912400
|             ILUMSET 136,'?????????1 ???????2 BEYOND 4092'                   02912600
|             ILUMSET 137,'SYNTAX ERROR - ????????1'                          02912800
|             ILUMSET 138,'????????1 FIELD %2 INVALID VALUE'                  02913000
|             ILUMSET 139,'PHASE 2 EXIT(S) IGNORED'                           02913200
|             ILUMSET 140,'LABEL ERROR'                                       02913400
|             ILUMSET 141,'SUBTASKED CHECKPOINT IGNORED'                      02913600
|             ILUMSET 142,'CHECKPOINT IGNORED - %1 TRACKS OR BLOCKS NEEDED OX02913800
|                   N SORTCKP'                                                02914000
|             ILUMSET 143,'CHECKPOINT IGNORED - A PARTITION CROSSES THE 16 MX02914200
|                   B VIRTUAL'                                                02914400
|             ILUMSET 144,'INVALID SPLIT CYLINDER EXTENT ON ????????1'        02914600
|             ILUMSET 145,'STORAGE PARAMETER IGNORED. %1 USED'                02914800
|             ILUMSET 146,'INVALID ADDRESS IN THE PARAMETER LIST (DISPLACEMEX02915000
|                   NT = %1)'                                                 02915200
|             ILUMSET 148,'ROUTE PARAMETER IGNORED'                           02915400
|             ILUMSET 149,'ERROR(%1) GETTING LABEL FOR ????????2'            02915600
|             ILUMSET 150,'ERROR(%1) READING VTOC FOR ????????2 LABEL'       02915800
|             ILUMSET 151,'CENTURY WINDOW FOR Y2 FORMATS IS FROM %1 TO %2'   02916000
|             ILUMSET 152,'WORK FILE NAME SPECIFIED TWICE'                    02916200
|             ILUMSET 154,'ERROR(%1) RETURN FROM GETVCE FOR ????????2'        02916400
|             ILUMSET 155,'GETVIS FOR ????????1 COMMON VTOC HANDLER WORK AREAX02916600
|                    FAILED. %2 NEEDED'                                       02916800
|             ILUMSET 158,'????????1 SYSNO IGNORED'                           02917000
|             ILUMSET 161,'SECONDARY ALLOCATION FOR SORTWORK WILL NOT BE SUPX02917200
|                   PORTED, RETURN CODE = %1'                                 02917400
|             ILUMSET 166,'MORE THAN 12 CONTROL FIELDS SPECIFIED - GETVIS STX02917600
|                   ORAGE NOT AVAILABLE'                                      02917800
|             ILUMSET 167,'SUM CANNOT BE USED WITH SORT FIELDS=COPY OR MERGEX02918000
|                    FIELDS=COPY'                                             02918200
|             ILUMSET 168,'ONLY FILES, SKIPREC, AND STOPAFT ARE APPLICABLE TX02918400
|                   O FIELDS=COPY'                                            02918600
|             ILUMSET 169,'ALL OPERANDS OTHER THAN FIELDS=NONE ARE IGNORED'   02918800
|             ILUMSET 170,'INREC CANNOT BE USED WITH MERGE OR SORT FIELDS=COX02919000
|                   PY'                                                       02919200
|             ILUMSET 171,'EXTENT EXCEEDS TRACK LIMIT OF 65535 ON ????????1'  02919400
|             ILUMSET 172,'LOADING INFORMATION SPECIFIED FOR PHASE%1 PRELOADX02919600
|                   ED USER EXIT ROUTINES'                                    02919800
|             ILUMSET 180,'REQUESTED DATA SPACE IS NOT AVAILABLE. RETURN CODX02920000
|                   E = %1, REASON CODE = !!!!!!!!2'                          02920200
|             ILUMSET 185,'REQUESTED GETVIS SORTING AREA IS NOT AVAILABLE'    02920400
|             ILUMSET 201,'TOO MANY RECORD LENGTH PARAMETERS'                 02920600
|             ILUMSET 202,'??????1 STATEMENT OPERANDS ARE IGNORED'            02920800
|             ILUMSET 203,'INVALID INCLUDE OR OMIT DELIMITER'                 02921000
|             ILUMSET 204,'DEFAULT ALTSEQ USED'                               02921200
|             ILUMSET 205,'?????????1 OPERAND IGNORED'                        02921400
|             ILUMSET 206,'BOTH INCLUDE AND DELBLANK DEFINED'                 02921600
|             ILUMSET 207,'RECORD DESCRIPTOR WORD NOT INCLUDED IN ????????1'  02921800
```

```
|             ILUMSET 208,'????????1 FIELD %2 BEYOND RECORD'                02922000
|             ILUMSET 209,'ADDROUT OPTION INVALID'                          02922200
|             ILUMSET 210,'INVALID LENGTH IN RELATIONAL CONDITION %1 FOR INCX02922400
|                   LUDE OR OMIT'                                          02922600
|             ILUMSET 211,'??????1 EXIT REQUIRES ???2'                      02922800
|             ILUMSET 212,'INVALID FORMAT FOR ASCII DATA'                   02923000
|             ILUMSET 213,'DELBLANK POSITION BEYOND 4092'                   02923200
|             ILUMSET 214,'EFFECTIVE L3 VALUE = %1'                         02923400
|             ILUMSET 215,'SUM FIELD %1 OVERLAPS CONTROL FIELD %2'          02923600
|             ILUMSET 216,'SUM FIELD %1 OVERLAPS RECORD DESCRIPTOR WORD'    02923800
|             ILUMSET 217,'SUM FIELD %1 OVERLAPS SUM FIELD %2'              02924000
|             ILUMSET 218,'TOO MANY VOLUME POSITIONAL PARAMETERS'           02924200
|             ILUMSET 219,'INVALID SELF-DEFINING TERM IN RELATIONAL CONDITIOX02924400
|                   N %1 FOR INCLUDE OR OMIT'                              02924600
|             ILUMSET 220,'????????1 FIELD %2 INVALID VALUE'                02924800
|             ILUMSET 221,'SPECIFIED E??1 VALID ONLY FOR VSAM FILE'         02925000
|             ILUMSET 222,'INVALID FORMAT IN RELATIONAL CONDITION %1 FOR INCX02925200
|                   LUDE OR OMIT'                                          02925400
|             ILUMSET 223,'INSUFFICIENT STORAGE FOR ????????1 FUNCTION'     02925600
|             ILUMSET 224,'PHASE 1 EXITS IGNORED BY MERGE OR SORT FIELDS=COPX02925800
|                   Y'                                                     02926000
|             ILUMSET 225,'EXIT E32 OR E38 IGNORED BY SORT'                 02926200
|             ILUMSET 226,'PRIORITY PARENTHESIS MISPLACED FOR INCLUDE OR OMIX02926400
|                   T'                                                     02926600
|             ILUMSET 227,'INCLUDE OR OMIT FORMAT INVALID'                  02926800
|             ILUMSET 228,'INVALID COMPARISON OPERATOR IN RELATIONAL CONDITIX02927000
|                   ON %1 FOR INCLUDE OR OMIT'                             02927200
|             ILUMSET 229,'ERROR IN LENGTH VALUE'                           02927400
|             ILUMSET 230,'L%1 VALUE INVALID'                               02927600
|             ILUMSET 231,'DATA = A INVALID'                                02927800
|             ILUMSET 232,'ALTERED RECORDS REQUIRE INREC, OUTREC, OR E15 OR X02928000
|                   E35 EXIT'                                              02928200
|             ILUMSET 233,'??????1 BLOCK SIZE = %2 BYTES'                   02928400
|             ILUMSET 234,'RECORD CONFLICTS WITH ??????1 BLOCK SIZE'        02928600
|             ILUMSET 235,'MISSING FORMAT IN RELATIONAL CONDITION %1 FOR INCX02928800
|                   LUDE OR OMIT'                                          02929000
|             ILUMSET 236,'DELBLANK POSITION BEYOND ??1'                    02929200
|             ILUMSET 237,'SYNTAX ERROR - INCLUDE OR OMIT'                  02929400
|             ILUMSET 238,'??1 INVALID FOR ????????2'                       02929600
|             ILUMSET 239,'SUM FIELD %1 LENGTH INVALID'                     02929800
|             ILUMSET 240,'FIELD OR VALUE GREATER THAN 8 CHARACTERS FOR INCLX02930000
|                   UDE OR OMIT'                                           02930200
|             ILUMSET 241,'??1 GREATER THAN ??2'                            02930400
|             ILUMSET 242,'INVALID FORMAT COMBINATION IN RELATIONAL CONDITIOX02930600
|                   N %1 FOR INCLUDE OR OMIT'                              02930800
|             ILUMSET 243,'INVALID LOGICAL OPERATOR IN RELATIONAL CONDITION X02931000
|                   %1 FOR INCLUDE OR OMIT'                                02931200
|             ILUMSET 244,'TOO MANY ????????1 OPERANDS'                     02931400
|             ILUMSET 245,'INCLUDE OR OMIT FIELD IN RELATIONAL CONDITION %1 X02931600
|                   BEYOND ???????2'                                       02931800
|             ILUMSET 246,'CONTROL FIELD %1 TOO LONG FOR TYPE'              02932000
|             ILUMSET 247,'EXIT E??1 NOT GIVEN FOR NONSTANDARD LABELS'      02932200
|             ILUMSET 248,'INVALID FIELD POSITION IN RELATIONAL CONDITION %1X02932400
|                    FOR INCLUDE OR OMIT'                                  02932600
|             ILUMSET 249,'CONTROL FIELD %1 BEYOND RECORD LENGTH'           02932800
|             ILUMSET 250,'ALTSEQ CANNOT BE USED WITH DATA = A'             02933000
|             ILUMSET 251,'INVALID ????????1 DEVICE'                        02933200
|             ILUMSET 252,'????????1 OPTION NOT APPLICABLE FOR ??????2 DEVICEX02933400
|                   '                                                      02933600
|             ILUMSET 253,'INVALID BLOCK SIZE FOR ??????1 DEVICE'           02933800
|             ILUMSET 254,'ALTSEQ STATEMENT IGNORED'                        02934000
|             ILUMSET 255,'INVALID PHASE %1 BRANCH TABLE ADDRESS'           02934200
|             ILUMSET 256,'TOTAL LENGTH OF CONTROL FIELDS EXCEEDS MAXIMUM (3X02934400
|                   072 BYTES)'                                            02934600
|             ILUMSET 257,'INVALID INCLUDE OR OMIT OPERAND'                 02934800
|             ILUMSET 258,'INCLUDE OR OMIT COND OPERAND MISSING'            02935000
|             ILUMSET 259,'TOO MANY ????????1 FIELDS'                       02935200
|             ILUMSET 260,'????????1 FIELD CONTAINS ONLY RDW'               02935400
|             ILUMSET 261,'INPUT RECORD LENGTH OR BLOCK SIZE LESS THAN 12 BYX02935600
|                   TES'                                                   02935800
|             ILUMSET 262,'SPECIFIED E??1 VALID ONLY FOR SAM FILE'          02936000
|             ILUMSET 263,'SPECIFIED E31 VALID ONLY WHEN CKPT SPECIFIED'    02936200
|             ILUMSET 264,'OUTPUT RECORD LENGTH LESS THAN ??1 BYTES'        02936400
```

```
|             ILUMSET 265,'??1 VALUE IS IGNORED FOR MERGE OR SORT FIELDS=COPX02936600
|                   Y'                                                    02936800
|             ILUMSET 266,'L%1 VALUE TOO LARGE FOR SORTIN%2'              02937000
|             ILUMSET 267,'INVALID CI SIZE - ???????1'                    02937200
|             ILUMSET 268,'CI SIZE NOT SPECIFIED FOR FBA OUTPUT. (%1) USED' 02937400
|             ILUMSET 269,'BLOCK SIZE DOES NOT MATCH ????????1 CI SIZE'    02937600
|             ILUMSET 270,'ADDROUT INVALID WITH CI FORMAT INPUT'           02937800
|             ILUMSET 271,'BYPASS IGNORED FOR FBA I/O ERRORS'             02938000
|             ILUMSET 272,'LABELS SET STANDARD FOR MANAGED ???????1'       02938200
|             ILUMSET 273,'LOADING INFORMATION NOT SPECIFIED FOR PHASE%1 EXIX02938400
|                   TS'                                                   02938600
|             ILUMSET 274,'INCONSISTENT REFORMATTING FIELDS FOUND'         02938700
|             ILUMSET 275,'WRONG RDW FIELD SPECIFIED WITH ????????1'      02938800
|             ILUMSET 276,'FIXED DATA NOT INCLUDED FOR ????????1'         02939000
|             ILUMSET 277,'SKIPREC OR STOPAFT NOT APPLICABLE TO MERGE'    02939200
|             ILUMSET 278,'SKIPREC NOT APPLICABLE TO INPFIL EXIT SPECIFICATIX02939400
|                   ON'                                                   02939600
|             ILUMSET 301,'INVALID SIGN'                                  02939800
|             ILUMSET 302,'MATCH NOT FOUND FOR CHANGE FIELD AT POSITION %1' 02939900
|             ILUMSET 303,'INSUFFICIENT VIRTUAL STORAGE FOR IN-CORE SORT'  02940090
|             ILUMSET 305,'I/O ERROR ON SORTWORK  CCB = ??????????????????????X02940200
|                   ????????????1'                                        02940400
|             ILUMSET 306,'INSUFFICIENT WORK SPACE'                       02940600
|             ILUMSET 308,'SECONDARY ALLOCATION OF ????????1 FAILED - RECOVERX02940800
|                   Y IN PROGRESS'                                        02941000
|             ILUMSET 310,'UNRECOVERABLE CHANNEL OR INTERFACE CONTROL CHECK' 02941200
|             ILUMSET 321,'?????1 COMPLETE, INSERT %2, DELETE %3, IN %4, OUTX02941400
|                    %5'                                                  02941600
|             ILUMSET 322,'?????1 ERROR, INSERT %2, DELETE %3, IN %4, OUT %5X02941800
|                   '                                                     02942000
|             ILUMSET 323,'?????1 COMPLETE, IN %2, OUT %3'                02942200
|             ILUMSET 324,'VSAM CONTROL BLOCK ERROR (%1) AT ???????2'      02942400
|             ILUMSET 325,'VSAM CLOSE ERROR ????????1 (%2)'               02942600
|             ILUMSET 326,'ERASE IN PROGRESS'                             02942800
|             ILUMSET 327,'I/O ERROR DURING ERASE'                        02943000
|             ILUMSET 330,'RECORD COUNT OFF, INSERT %1, DELETE %2, IN %3, OUX02943200
|                   T %4'                                                 02943400
|             ILUMSET 331,'WORK SPACE USED: %1 FBA BLOCKS'                02943600
|             ILUMSET 332,'WORK SPACE USED: %1 TRACKS ON ????2'           02943800
|             ILUMSET 333,'?????1 ERROR, IN %2, OUT %3'                   02944000
|             ILUMSET 334,'SORT CAPACITY APPROXIMATELY %1 RECORDS'        02944200
|             ILUMSET 335,'SORT CAPACITY APPROXIMATELY %1 RECORDS OF MODAL LX02944400
|                   ENGTH'                                                02944600
|             ILUMSET 340,'DATA SPACE STORAGE USED = %1K BYTES'           02944800
|             ILUMSET 345,'GETVIS SORTING AREA USED = %1 BYTES'           02945000
|             ILUMSET 352,'OUT OF SEQUENCE ON SORTIN%1'                   02945200
|             ILUMSET 361,'I/O ERROR. CCB/IORB=??????????????????????????????X02945400
|                   ???1'                                                 02945600
|             ILUMSET 362,'I/O ERROR - BYPASS'                            02945800
|             ILUMSET 363,'WRONG LENGTH BLOCK - ????????1, %2'            02946000
|             ILUMSET 364,'WRONG LENGTH BLOCK - ????????1, %2'            02946200
|             ILUMSET 366,'WRONG LENGTH RECORD EXIT E??1 - %2'            02946400
|             ILUMSET 367,'SOME LOGICAL RECORDS ARE LESS THAN THE MINIMUM LEX02946600
|                   NGTH'                                                 02946800
|             ILUMSET 369,'RETURN CODE ERROR E??1'                        02947000
|             ILUMSET 370,'TOO SHORT RECORD FOUND - %1'                   02947200
|             ILUMSET 371,'INVALID SIGN IN INCLUDE OR OMIT INPUT'         02947400
|             ILUMSET 374,'INPUT SEGMENTS IN WRONG ORDER ????????1'       02947600
|             ILUMSET 375,'SORTOUT FILE ON SYS(%1) OVERLAPS WORK EXTENT ON SX02947800
|                   YS(%2)'                                               02948000
|             ILUMSET 377,'INPUT SEGMENT TOO LONG, SEGMENT LENGTH = %1, TOTAX02948200
|                   L LENGTH = %2, ????????3'                             02948400
|             ILUMSET 381,'VSAM OPEN ERROR ????????1   (%2)'              02948600
|             ILUMSET 382,'VSAM CONTROL BLOCK ERROR (%1) AT ???????2'      02948800
|             ILUMSET 383,'VSAM INPUT ERROR ????????1   ?2 (%3)'          02949000
|             ILUMSET 384,'VSAM OUTPUT ERROR   ?1 (%2)'                    02949200
|             ILUMSET 385,'VSAM CLOSE ERROR ????????1   (%2)'             02949400
|             ILUMSET 387,'VSAM LOAD ERROR'                               02949600
|             ILUMSET 400,'NO RECORDS WERE WRITTEN TO THE OUTPUT FILE - RC =X02949680
|                    %1'                                                  02949760
|             ILUMSET 401,'NO RECORDS WERE WRITTEN TO THE OUTPUT FILE - RC =X02949840
|                    %1'                                                  02949920
|             ILUMSET 600,'ICETOOL UTILITY RUN STARTED'                   02950000
```

```
|             ILUMSET 601,'ICETOOL UTILITY RUN ENDED - RETURN CODE:  ??1'    03000000
|             ILUMSET 602,'OPERATION RETURN CODE:  ??1'                      03050000
|             ILUMSET 603,'INFORMATION PRINTED ON ??????1'                   03100000
|             ILUMSET 604,'ERROR IN KEYWORD, PARAMETER, OR DELIMITER'        03150000
|             ILUMSET 607,'STATISTICS FOR ??????????????1:'                  03200000
|             ILUMSET 608,'  MINIMUM:  ??????????????????1, MAXIMUM:  ????????*03250000
|                   ????????2'                                               03300000
|             ILUMSET 609,'  AVERAGE:  ??????????????????1, TOTAL  :  ????????*03350000
|                   ????????2'                                               03400000
|             ILUMSET 610,'NUMBER OF UNIQUE VALUES FOR ??????????????1:  ???*03450000
|                   ????????????2'                                           03500000
|             ILUMSET 611,'TOTAL FOR ??????????????1 OVERFLOWED 15 DECIMAL D*03550000
|                   IGITS'                                                   03600000
|             ILUMSET 612,'NO ERRORS FOUND IN STATEMENT'                     03900000
|             ILUMSET 613,'REQUIRED KEYWORD MISSING:  ??????????????????????*03950000
|                   ????????????????????????????????????????1'             04000000
|             ILUMSET 614,'INVALID OPERATOR'                                 04050000
|             ILUMSET 616,'STATEMENT DOES NOT END AT OR BEFORE COLUMN 72'    04300000
|             ILUMSET 617,'RECORD COUNT OVERFLOWED 15 DECIMAL DIGITS'        04350000
|             ILUMSET 618,'INVALID ??????????????1 VALUE - RECORD:  ????????*04387500
|                   ????????2, HEX VALUE:  ????????????????????????????????????*04425000
|                   ??3'                                                     04462500
|             ILUMSET 619,'INVALID LENGTH, FORMAT, OR COMBINATION FOR ??????*04500000
|                   ??1 OPERATION'                                           04550000
|             ILUMSET 620,'SUM OF POSITION AND LENGTH GREATER THAN ?????1'   04600000
|             ILUMSET 621,'DFSORT/VSE DETECTED AN ERROR AFTER E35 EXIT PROCE*04650000
|                   SSING COMPLETED'                                         04700000
|             ILUMSET 622,'EXPECTED CONTINUATION LINE NOT FOUND'             04750000
|             ILUMSET 623,'MAXIMUM NUMBER OF ????????1 KEYWORDS EXCEEDED'    04800000
|             ILUMSET 624,'MAXIMUM NUMBER OF TO FILENAMES EXCEEDED'          04850000
|             ILUMSET 625,'OPEN FAILED FOR ??????1'                          04900000
|             ILUMSET 626,'LIMIT FOR INVALID VALUES REACHED'                 04950000
|             ILUMSET 627,'DFSORT/VSE CALL ????1 FOR ????2 FROM ????????3 TO *05000000
|                   ????????4 ??????????5'                                  05050000
|             ILUMSET 628,'RECORD COUNT:  ??????????????1'                   05100000
|             ILUMSET 629,'HIGHER AND LOWER VALUES EXCLUDE ALL RECORDS'      05150000
|             ILUMSET 630,'MODE IN EFFECT:  ????????1'                       05200000
|             ILUMSET 631,'NUMBER OF VALUES IN RANGE FOR ??????????????1:  ?*05250000
|                   ??????????????2'                                        05300000
|             ILUMSET 632,'SOURCE FOR ICETOOL STATEMENTS:  ????????1'        05350000
|             ILUMSET 633,'RETURN AREA IS ?????1 BYTES, BUT ?????2 BYTES ARE*05400000
|                    REQUIRED'                                               05450000
|             ILUMSET 635,'NUMBER OF HEADER KEYWORDS DOES NOT MATCH NUMBER O*05650000
|                   F ON KEYWORDS'                                           05700000
|             ILUMSET 637,'??????1 RECORD LENGTH OF ????2 BYTES EXCEEDS MAXI*05750000
|                  MUM OF ????3 BYTES'                                       05800000
|             ILUMSET 638,'NUMBER OF RECORDS RESULTING FROM CRITERIA:  ?????*06050000
|                   ??????????1'                                            06100000
|             ILUMSET 640,'INVALID FORMATTING ITEM'                          06350000
|             ILUMSET 650,'DEFINE STATEMENT MISSING OR DOES NOT CONTAIN REQU*06450000
|                   IRED KEYWORD(S) FOR FILE ????????1'                      06500000
|             ILUMSET 651,'MAXIMUM NUMBER OF DEFINE STATEMENTS EXCEEDED'     06550000
|             ILUMSET 652,'NO DFSORT/VSE STATEMENT(S) FOUND'                 06600000
|             ILUMSET 653,'INVALID DFSORT/VSE STATEMENT DEFINER FOUND IN LIN*06650000
|                   E ??1'                                                   06700000
|             ILUMSET 654,'TOO MANY STATEMENTS IN DFSORT/VSE SECTION'        06750000
|             ILUMSET 655,'LOAD FAILED FOR SORT PHASE. RETURN CODE = ??1'    06800000
|             ILUMSET 656,'??????1K OF EXTRA STORAGE REQUESTED, ??????2K AVA*06850000
|                   ILABLE'                                                  06900000
|             ILUMSET 657,'GETVIS FAILED FOR EXTRA STORAGE'                  06950000
|             ILUMSET 658,'GETVIS FAILED FOR PRINT BUFFERS'                  07000000
|             ILUMSET 659,'DEFINE STATEMENT IGNORED'                         07050000
|             ILUMSET 660,'CDLOAD FAILED. RETURN CODE = ??1'                 07100000
|             ILUMSET 661,'??1 ????????????????????????????????????????????*07150000
|                   ????????????????????????????????2'                      07200000
|             ILUMSET 662,'USTART, OR UEND, OR BOTH DELIMITERS MISSING'      07250000
|             ILUMSET 663,'MAXIMUM NUMBER OF FROM FILENAMES EXCEEDED'        07300000
|             ILUMSET 664,'DUPLICATE DFSORT/VSE STATEMENT DEFINER FOUND IN L*07350000
|                   INE ??1'                                                 07400000
|             ILUMSET 665,'GETVIS FAILED FOR DFSORT/VSE SECTION'             07450000
|             ILUMSET 701,'LOCALE PROCESSING WAS USED WITH ACTIVE LOCALE ???X07451400
|                   ????????????????????????????1'                          07452800
|             ILUMSET 702,'LOCALE PROCESSING CONFLICTS WITH ?????1'          07454200
```

```
|             ILUMSET 703,'LANGUAGE ENVIRONMENT INITIALIZATION FAILED'       07455600
|             ILUMSET 704,'LOAD FAILED FOR ACTIVE LOCALE ??????????????????X07457000
|                   ?????????????1'                                         07458400
|             ILUMSET 705,'LANGUAGE ENVIRONMENT SERVICE ERROR (%1)'          07459800
|             ILUMSET 706,'CONTROL FIELDS FOR LOCALE PROCESSING ARE TOO COMPX07461200
|                   LEX'                                                    07462600
|             ILUMSET 707,'LOCALE PROCESSING REQUESTED - GETVIS STORAGE NOT X07464000
|                   AVAILABLE'                                              07465400
|             ILUMSET 708,'CULTURAL ENVIRONMENT SUPPORT REQUESTED BUT NOT ENX07466800
|                   ABLED'                                                  07468200
|             ILUMSET 806,'???1: TIME A: %2 SECONDS, TIME B: %3 SECONDS'     07469600
|             ILUMSET 808,'MODULE STATUS: SVA'                               07471000
|             ILUMSET 810,'ECPS: VSE MODE OF OPERATION'                      07472400
|             ILUMSET 816,'GETVIS SORTING AREA OBTAINED = %1 BYTES BELOW, %2X07473800
|                    BYTES ABOVE'                                           07475200
|             ILUMSET 817,'IN-CORE SORT'                                     07476600
|             ILUMSET 820,'SECONDARY EXTENT FOR DATA WAS ALLOCATED ON ??????X07478000
|                   ?1'                                                     07479400
|             ILUMSET 821,'SECONDARY EXTENT FOR INDEXES WAS ALLOCATED ON ???X07480800
|                   ????1'                                                  07482200
|             ILUMSET 832,'??????1: REAL I/O NOT USED'                       07484490
|             ILUMSET 839,'%1'                                               07485380
|             ILUMSET 840,'DATA SPACE WAS NOT USED. RETURN CODE = %1'        07486400
|             ILUMSET 841,'%1 %2'                                            07488790
|             ILUMSET 842,'%1 %2'                                            07489780
|             ILUMSET 843,'???1 %2 %3'                                       07490770
|             ILUMSET 844,'???????????1'                                     07491760
|             ILUMSET 845,'%1'                                               07492750
|             ILUMSET 847,'%1'                                               07493740
|             ILUMSET 849,'%1 %2 %3'                                         07494730
|             ILUMSET 851,'%1'                                               07495720
|             ILUMSET 852,'%1 %2'                                            07496710
|             ILUMSET 858,'???1'                                             07497700
|             ILUMSET 859,'???1'                                             07498690
|             ILUMSET 863,'%1'                                               07499680
|             ILUMSET 864,'???1 %2 %3 %4'                                    07500670
|             ILUMSET 865,'???1 %2 %3'                                       07501660
|             ILUMSET 867,'???1 %2'                                          07502650
|             ILUMSET 869,'%1 %2'                                            07503640
|             ILUMSET 870,'%1'                                               07504630
|             ILUMSET 871,'%1'                                               07505620
|             ILUMSET 872,'%1 ????2'                                         07506610
|             ILUMSET 873,'%1'                                               07507600
|             ILUMSET 874,'TRACE TABLE NOT USED'                             07515800
|             ILUMSET 875,'??????1'                                          07518190
|             ILUMSET 876,'??????1'                                          07519180
|             ILUMSET 882,'???????1 ???????2 ????3'                         07520170
|             ILUMSET 883,'%1 %2 %3 %4 %5 %6 %7 %8 %9 %10 %11'              07522800
|             ILUMSET 884,'%1 %2 %3 %4 %5 %6 %7 %8 %9 %10'                   07524200
|             ILUMSET 885,'%1 %2 %3 %4 %5 %6'                                07525600
|             ILUMSET 886,'TRACE TABLE (LAST EVENT AT TOP)'                  07527000
|             ILUMSET 887,'     ??????????????????????1'                     07528400
|             ILUMSET 888,'%1 %2 %3 %4 %5 %6 %7 %8 %9 %10'                   07529800
|             ILUMSET 889,'%1 %2 %3 %4 %5 %6 %7 %8 %9 %10'                   07531200
|             ILUMSET 890,'%1 %2 %3'                                         07532600
|             ILUMSET 891,'%1 %2 %3'                                         07534000
|             ILUMSET 892,'%1 %2 %3 %4 %5 %6 %7'                            07534400
|             ILUMSET 893,'%1 %2 %3 %4 %5 %6'                               07534800
|             ILUMSET 894,'%1 %2 %3 %4 %5 %6 %7 %8 %9 %10'                   07534900
|             ILUMSET 895,'%1 %2 %3 %4 %5 %6 %7 %8 %9 %10'                   07535000
|             ILUMSET 896,'%1 %2 !!3 !!!!!!!!4 !!!!!!!!5'                    07535020
|             ILUMSET 897,'%1 !!2 !!!!!!!!3 !!!!!!!!4'                       07535050
|             ILUMSET 899,'%1 %2 %3'                                         07535100
|             ILUMSET 900,'CODE OVERLAID BY BUFFERS'                         07535400
|             ILUMSET 901,'ABNORMAL RETURN FROM REALAD'                      07536800
|             ILUMSET 902,'PFREE ERROR'                                      07538200
|             ILUMSET 903,'ABNORMAL CODE OVERLAY RSA'                        07539600
|             ILUMSET 904,'UNUSUAL OVERLAY CONDITION'                        07541000
|             ILUMSET 905,'ABNORMAL BUFFER AREA SIZE'                        07542400
|             ILUMSET 906,'WRITE BACK LIST FULL'                             07543800
|             ILUMSET 907,'MODULE OVERLAID BY TABLES'                        07545200
|             ILUMSET 925,'%1 %2'                                            07547200
|             MEND                                                          07550000
```

```
Figure 45. ILUMSGS Macro
```

BACK_1 Summary of Changes for Previous Releases of DFSORT/VSE

Subtopics:

BACK_1.1 Third Edition, February 1997

Subtopics:

BACK_1.1.1 Programming Support for Release 3

Subtopics:

BACK_1.1.1.1 National Language Support

Cultural Sort and Merge: DFSORT/VSE will allow the selection of an active locale at installation or run time and will produce sorted or merged records for output according to the collating rules defined in the active locale. This provides sorting and merging for single- or multi-byte character data based on defined collating rules which retain the cultural and local

characteristics of a language.

Cultural Include and Omit: DFSORT/VSE will allow the selection of an active locale at installation or run time and will include or omit records for output according to the collating rules defined in the active locale. This provides inclusion or omission for single- or multi-byte character data based on defined collating rules which retain the cultural and local characteristics of a language.

DFSORT/VSE Messages Translation: DFSORT/VSE provides the ability for easy translation of messages into different languages. Flexible positioning of variables within a message is supported.

ICETOOL Reports: ICETOOL's DISPLAY operator allows date, time, and numeric values in reports to be formatted in many of the notations used throughout the world. (Note that this support was first made available in DFSORT/VSE Version 3 Release 2.)

BACK_1.1.1.2 Productivity

Year 2000 Features: DFSORT/VSE's Year 2000 support will help you prepare for the turn of the century by correctly processing your two-digit year data. New Y2C, Y2Z, Y2P, and Y2D formats, in conjunction with a new Y2PAST installation and run-time option, allow you to handle two-digit year data in the following ways:

- Set the appropriate century window for your applications (for example, 1915-2014 or 1950-2049).
- Order two-digit character, zoned decimal, packed decimal or decimal year data according to the century window using SORT or MERGE (for example, order 96 representing 1996 before 00 representing 2000 in ascending sequence, or order 00 before 96 in descending sequence).
- Transform two-digit character, zoned decimal, packed decimal or decimal year data to four-digit character (or zoned decimal) year data according to the century window using OUTREC (for example, transform 96 to 1996 and 00 to 2000).

A new PD0 format allows you to order parts of packed decimal fields, such as month and day in date fields, using SORT or MERGE.

New PZ, PSI and ZSI formats allow you to transform packed decimal and zoned decimal fields, such as month and day in date fields, to character fields using OUTREC.

New character and hexadecimal string separators allow you to insert literals in reformatted fields using OUTREC (for example,

'/' in mm/dd/yyyy fields).

File Management Systems: New installation option FMS allows you to specify that DFSORT/VSE should attempt to take advantage of benefits provided by the File Management System installed at your site, such as:

- Dynamic logical and physical device assignment
- Dynamic primary and secondary extent allocation

INCLUDE and OMIT Enhancements: Do sophisticated filtering with new INCLUDE and OMIT features:

- Use the substring search capability to allow inclusion of records when:
    - A specified character or hexadecimal constant is found anywhere within a specified input field (that is, a constant is a substring within a field) or
    - A specified input value is found anywhere within a specified character or hexadecimal constant (that is, a field is a substring within a constant).
- Use bit level logic to allow inclusion of records based on:
    - Bit comparison tests of a binary field against a bit string constant. The bit string constant allows you to specify which bits of the input field must be on, which must be off and which can be ignored.
    - Bit operator tests of a binary field against a bit or hexadecimal mask. The mask allows you to test many different possible bit combinations with a single operation, similar to what you can do using the Test Under Mask (TM) machine instruction.
- Have unlimited levels of parentheses within logical expressions which allows you to create more complex conditions for inclusion of records.

SORT or MERGE Bit Control Fields: Use DFSORT/VSE's new bit control fields in conjunction with byte control fields to give you additional ways to sort or merge your data.

Input/Output Improvements: You can take advantage of the following input/output handling improvements:

- Copy up to nine input files to one output file.
- Process records and output them to a printer or punch device for sort, merge, or copy applications.
- Process input and output files on the IBM 3590 Tape Subsystems for sort, merge, and copy applications.

ICETOOL: ICETOOL is now more versatile as a result of enhancements to the existing operators. The improvements include:

- Allowing up to nine input files to be processed with the COPY, COUNT, DISPLAY, RANGE, STATS, and VERIFY operators.
- Allowing the active locale to be specified for the COPY, COUNT, and SORT operators, overriding the installation default for the active locale. Locale processing provides a way to SORT or COPY and INCLUDE or OMIT records according to the language and country rules defined in the active locale.

BACK_1.1.1.3 Performance

Performance enhancements for DFSORT/VSE Version 3 Release 3 include the following:

- Dataspace sorting, introduced in DFSORT/VSE Version 3 Release 1 for fixed-length record sort applications, is now available for variable-length record sort applications.
- Improved Extended Count-Key-Data (ECKD) disk device support for work files.
- Improved data processing methods for:
  - Incore and non-incore dataspace sorting
  - Incore getvis sorting
  - Copy and merge applications
  - Input and output tape file processing for sort, merge, and copy applications.

BACK_1.1.1.4 Additional Enhancements

EQUALS/NOEQUALS: Preserve the original sequence of records that collate identically from input to output for merge applications using the EQUALS option.

Virtual Storage Constraint Relief (VSCR): Reduction in the number of DFSORT/VSE modules that are loaded into the 24-bit Shared Virtual Area (SVA) and partition program area.

HDR2: HDR2 support for tape output files.

BACK_1.2 Second Edition, October 1995

Subtopics:

BACK_1.2.1 Programming Support for Release 2

DFSORT/VSE Release 2 continues the strategy of providing performance improvements and productivity features. These improvements and features are described in more detail in the subsections that follow.

Subtopics:

BACK_1.2.1.1 Performance

With DFSORT/VSE Release 2, performance enhancements for dataspace sorting of fixed-length records and for getvis sorting

of fixed-length and variable-length records include:

- Dynamic control of storage for dataspace sorting and getvis sorting
- Improved data processing methods for incore getvis sorting
- Improved data processing methods for non-incore getvis and dataspace sorting
- Reduced amount of work space required for SAM ESDS work files

---

BACK_1.2.1.2 Productivity

With DFSORT/VSE Release 2, your productivity is improved because you can:

- Create reports and analyze data using the new ICETOOL utility
- Select subsets of data using the STOPAFT and SKIPREC options
- Tailor DFSORT/VSE to suit your needs by using the new and modified ILUINST installation defaults and run-time options
- Access both introductory and reference information more quickly using a new DFSORT/VSE publication and a new DFSORT/VSE CD-ROM

BACK_1.2.1.3 ICETOOL

With ICETOOL, a versatile new DFSORT/VSE utility, you can do reporting and analysis of data at your site. ICETOOL allows you to perform multiple operations on one or more files in a single job step. This batch front-end utility uses the capabilities of DFSORT/VSE to perform the operations you request.

ICETOOL has thirteen operators: COPY, COUNT, DEFAULTS, DEFINE, DISPLAY, MODE, OCCUR, RANGE, SELECT, SORT, STATS, UNIQUE, and VERIFY. By using one operator or a combination of these operators, you can easily create applications that perform a variety of tasks including:

- Sorting input files to one or more output files
- Creating multiple copies of input files

- Creating output files containing subsets of input files based on various criteria
- Creating detailed reports allowing control of title, date, time, page numbers, headings, lines per page, field formats, and total, maximum, minimum, and average values
- Creating reports showing unique values for selected character and numeric fields and the number of times each occurs
- Creating reports or output files for records with: duplicate values, nonduplicate values, or values that occur n times, less than n times, or more than n times
- Creating a wide variety of reports using the preferred date, time, and numeric notations of individual countries
- Creating a report showing the DFSORT/VSE installation defaults in use
- Printing messages that give statistical information for selected numeric fields such as minimum, maximum, average, total, count of values, and count of unique values
- Printing messages that identify invalid decimal values
- Using three different modes, (stop, continue, and scan) to control error checking and actions after error detection for groups of operators

ICETOOL can be called directly or from a program. It also produces messages and return codes describing the results of each operation and any errors detected. Although you generally do not need to look at the DFSORT/VSE messages produced as a result of an ICETOOL run, they are available if you need them.

BACK_1.2.1.4 SKIPREC and STOPAFT Options

With DFSORT/VSE Release 2, there are two new options that allow you to select subsets of data for sorting or copying. These options are:

- The SKIPREC option which enables you to specify the number of records you want to skip before starting to sort or copy the input files.
- The STOPAFT option which enables you to specify the maximum number of records you want accepted for sorting or copying.

BACK_1.2.1.5 Installation Defaults and Run-time Options

Several ILUINST installation defaults have been added or changed providing you with more flexibility in using installation defaults and run-time options. These defaults include:

- GVSRLOW and GVSRANY which enable you to specify the reserved partition GETVIS area for user application requirements.
- WRKSEC which enables you to allow or suppress the dynamic secondary allocation for SAM ESDS work files.
- STORAGE which may be specified in megabytes.

To give you even more flexibility on a daily basis, several run-time options have been added or changed. These run-time options include:

- WRKSEC and NOWRKSEC which enable you to override the installation defaults.
- GVSRLOW and GVSRANY which enable you to override the installation defaults.
- STORAGE which may be specified in megabytes.
- CKPT which enables you to restart DFSORT/VSE when the DFSORT/VSE modules are placed in SVA.

BACK_1.2.1.6 Additional Enhancements

Additional changes that you will find helpful are:

- An extended DFSORT/VSE parameter list feature which enables you to supply the *end of parameter* list indicator.
- Tagging of the error position in an invalid parameter list control statement image.
- The END control statement which enables you to discontinue accepting control statements.

- Improved work file processing through internal enhancements to the SAM ESDS work files secondary allocation algorithm which allows you to use multivolume SAM ESDS work files.

- Improved system and application availability with additional program modules placed above 16 MB virtual.

With DFSORT/VSE, Release 2, a new book is included in the DFSORT/VSE library, *Getting Started with DFSORT/VSE*, SC26-7101. This book gives you all of the information and instructions you need to start using the features of DFSORT/VSE.

Also new with this release is the *DFSORT/VSE Online Product Library*, SK2T-8730. This CD-ROM contains all of the books in the DFSORT/VSE library except for the *DFSORT/VSE Reference Summary* and the *DFSORT/VSE Licensed Program Specifications*.

BACK_1.3 First Edition, September 1994

Subtopics:

- BACK_1.3.1 Programming Support for Release 1

BACK_1.3.1 Programming Support for Release 1

Subtopics:

- BACK_1.3.1.1 Performance
- BACK_1.3.1.2 31-bit Addressing Support
- BACK_1.3.1.3 Other Enhancements
- BACK_1.3.1.4 Operating System Environment

BACK_1.3.1.1 Performance

Enhancements include:

- Dataspace sorting, a DFSORT/VSE capability that uses data space available with VSE/ESA 1.3 in supervisor MODE=ESA in place of intermediate work space for fixed-length record sort applications.

- Getvis sorting, a DFSORT/VSE capability that uses main storage above and below 16 MB virtual for fixed-length and variable-length record sort applications.

BACK_1.3.1.2 31-bit Addressing Support

ESA/370 and ESA/390 technology in VSE/ESA 1.3 provides the following enhancements for DFSORT/VSE:

- 31-bit addressing support for a calling program and user exit routines.
- DFSORT/VSE phases can be loaded into 31-bit SVA.
- DFSORT/VSE and VSAM working storage can be located above 16 MB virtual.
- Reduction in the use of work files by using partition storage above 16 MB virtual with getvis sorting.

BACK_1.3.1.3 Other Enhancements

Secondary allocation for SAM ESDS work files is supported with the VSE/VSAM Space Management for SAM Feature installed.

Several ILUINST installation and run-time options have been added:

- DSPSIZE, a new operand on the OPTION control statement, controls the use of dataspace sorting for fixed-length record sort applications.
- GVSIZE, a new operand on the OPTION control statement, controls the use of getvis sorting for fixed-length and variable-length record sort applications.
- STXIT or NOSTXIT, new operands on the OPTION control statement, control the use of DFSORT/VSE's STXIT routine for sort, merge, and copy applications.
- FIELDS=NONE, a new operand on the SUM control statement, allows the elimination of duplicate records with equal control fields, without summing, for sort and merge applications.

BACK_1.3.1.4 Operating System Environment

DFSORT/VSE Version 3 Release 1 can run only in the VSE/ESA environment.

DFSORT/VSE Version 3 Release 1 does not support the CMS/DOS environment of VM/ESA. DFSORT/CMS 5684-134 Version 2 Release 1 is available to satisfy users' needs for a sort product in the CMS/DOS environment of VM/ESA.

INDEX Index

Special Characters

A

B

C

D

E

F

G

I

J

L

M

N

O

P

R

S

T

V

W

Y

Z

ZDPRINT, 1.4.1.4

BACK_2 Communicating Your Comments to IBM

DFSORT/VSE
Installation and Tuning Guide
Version 3 Release 4

Publication No. SC26-7041-03

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you are mailing a readers' comment form (RCF) from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments by FAX, use this number:
  - United States: 1-800-426-6209
  - Other countries: (+1)+408+256-7896
- If you prefer to send comments electronically, use this network ID:
  - IBMLink from U.S. and IBM Network: STARPUBS at SJEVM5
  - IBMLink from Canada: STARPUBS at TORIBM
  - IBM Mail Exchange: USIB3VVD at IBMMAIL
  - Internet: starpubs@vnet.ibm.com

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies.

COMMENTS Readers' Comments -- We'd Like to Hear from You

```
DFSORT/VSE
Installation and Tuning Guide
Version 3 Release 4

Publication No. SC26-7041-03
```

Overall, how satisfied are you with the information in this book?

Legend:

1
Very satisfied
2
Satisfied
3
Neutral
4
Dissatisfied
5
Very dissatisfied

	1	2	3	4	5
Overall satisfaction					

How satisfied are you that the information in this book is:

	1	2	3	4	5
Accurate					
Complete					
Easy to find					
Easy to understand					
Well organized					
Applicable to your tasks					

Please tell us how we can improve this book:

International Business Machines Corporation
RCF Processing Department
M86/050
5600 Cottle Road
SAN JOSE, CA 95193-0001

Name . . . . . _____
Company or Organization _____
Address . . . . . _____
_____
_____
Phone No. . . . . _____

IBM Library Server Print Preview

DOCNUM = SC26-7041-03
DATETIME = 04/27/98 08:39:45
BLDVERS = 1.3.0
TITLE = DFSORT/VSE Installation and Tuning Guide
AUTHOR =

PATH = /home/webapps/epubs/htdocs/book