DFSORT/VSE: Ask Professor Sort

DFSORT Web Site

For papers, online books, examples and more, visit the DFSORT/VSE website at URL:

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

Contents

DFSORT/VSE: Ask Professor Sort

What is ICETOOL and how can it help me?

Professor Sort says ...

ICETOOL is a versatile file processing and reporting utility that provides an easy-to-use batch front-end for DFSORT/VSE. ICETOOL combines new features with previously available DFSORT/VSE features to perform complex sorting, copying, reporting and analytical tasks using multiple files in a single job step.

The thirteen ICETOOL operators briefly described below provide new tools for programmers:

- **COPY** - copies one or more input files to one or more output files.
- **COUNT** - prints a message containing the count of records.
- **DEFAULTS** - prints the DFSORT/VSE installation defaults.
- **DEFINE** - specifies input or output file characteristics for COUNT, COPY, DISPLAY, OCCUR, RANGE, SELECT, SORT, STATS, UNIQUE and VERIFY operations, and can also be used to indicate where the DFSORT/VSE messages are to be routed for a group of operations.
- **DISPLAY** - prints the values or characters of specified numeric or character fields. Simple, tailored or sectioned reports can be produced.
- **MODE** - sets/resets scanning and error actions.
- **OCCUR** - prints each unique value for specified character or numeric fields and how many times it occurs. Simple or tailored reports can be produced. The values printed can be limited to those for which the value count meets specified criteria (for example, only duplicate values or only non-duplicate values).
- **RANGE** - prints a message containing the count of values in a specified range for a specified numeric field.
- **SELECT** - selects records from files for inclusion in an output file based on meeting criteria for the number of times specified numeric or character field values occur (for example, only duplicate values or only non-duplicate values).
- **SORT** - sorts one or more input files to one or more output files.
- **STATS** - prints messages containing the minimum, maximum, average, and total for specified numeric fields.
- **UNIQUE** - prints a message containing the count of unique values for a specified numeric or character field.
- **VERIFY** - examines specified decimal fields in one or more input files and prints a message identifying each invalid value found for each field.

Here's an example of the JCL and control statements for an ICETOOL job. The details and output for this example can be found in *Getting Started with DFSORT/VSE.* Other ICETOOL examples can be found in *DFSORT/VSE Application Programming Guide*, and in *ICETOOL mini-user guide.*

```
* $$ JOB JNM=PGMA
* $$ LST CLASS=E
// JOB   PGMA JOBA,PROGRAMMER
// ASSGN SYS012,FEF
* $$ LST CLASS=E,JNM=DFSMSG,LST=SYS012
// OPTION NOSYSDMP
// ASSGN SYS013,SYSLST
// ASSGN SYS014,SYSLST
// ASSGN SYS015,SYSLST
// ASSGN SYS016,SYSLST
// DLBL VSESPUC,'VSESP.USER.CATALOG', VSAM
// DLBL ALL,'SORT.BRANCH', VSAM,CAT=VSESPUC,DISP=(OLD,KEEP)
// DLBL CADASD,'%%CA', VSAM,RECORDS=20,RECSIZE=33,CAT=VSESPUC
// DLBL CODASD,'%%CO', VSAM,RECORDS=20,RECSIZE=33,CAT=VSESPUC
// DLBL D1,'SORT.COPY1', VSAM,RECORDS=20,RECSIZE=33,CAT=VSESPUC
// DLBL D2,'SORT.COPY2', VSAM,RECORDS=20,RECSIZE=33,CAT=VSESPUC
// DLBL D3,'SORT.COPY3', VSAM,RECORDS=20,RECSIZE=33,CAT=VSESPUC
// DLBL D4,'SORT.COPY4', VSAM,RECORDS=20,RECSIZE=33,CAT=VSESPUC
// ASSGN SYS010,181
// TLBL CATAPE,'CA.BRANCH', 111111, 1
// ASSGN SYS011,181
// TLBL COTAPE,'CO.BRANCH', 111111, 2
// DLBL BKIN,'SORT.SAMPIN', VSAM,CAT=VSESPUC,DISP=(OLD,KEEP)
// DLBL BKADD,'SORT.SAMPADD', VSAM,CAT=VSESPUC,DISP=(OLD,KEEP)
// DLBL DAPUBS,'%%SORT', VSAM,RECORDS=100,RECSIZE=173,CAT=VSESPUC
// DLBL BKOUT,'SORT.BOOKS1', VSAM,RECORDS=100,RECSIZE=173,CAT=VSESPUC
// EXEC ICETOOL,SIZE=100K

* Input files characteristics for ICETOOL operations
DEFINE NAME(ALL) TYPE(F) LENGTH(33)
DEFINE NAME(CODASD) TYPE(F) LENGTH(33)
DEFINE NAME(CADASD) TYPE(F) LENGTH(33)
DEFINE NAME(BKADD) TYPE(F) LENGTH(173)
DEFINE NAME(BKIN) TYPE(F) LENGTH(173)
DEFINE NAME(BKOUT) TYPE(F) LENGTH(173)
DEFINE NAME(DAPUBS) TYPE(F) LENGTH(173)

* Tape output files characteristics
DEFINE NAME(COTAPE) UNIT(011)
DEFINE NAME(CATAPE) UNIT(010)

* DFSORT/VSE messages printer queue
DEFINE ROUTE(012)

* Statistics from all branches
STATS FROM(ALL) ON(18,4,ZD) ON(28,6,PD) ON(22,6,PD)
* Books from VALD and WETH
SORT FROM(BKIN,BKADD) TO(DAPUBS) USE
USTART
   SORT FIELDS=(106,4,A,1,75,A),FORMAT=CH
   INCLUDE COND=(106,4,EQ,C'VALD',OR,106,4,EQ,C'WETH'),
           FORMAT=CH
UEND
```

```
* Separate output for California and Colorado branches
SORT FROM(ALL) TO(CADASD,CATAPE) USE
USTART
  SORT FIELDS=(1,15,CH,A)
  INCLUDE COND=(16,2,CH,EQ,C'CA')
UEND
SORT FROM(ALL) TO(CODASD,COTAPE) USE
USTART
   SORT FIELDS=(1,15,CH,A)
   INCLUDE COND=(16,2,CH,EQ,C'CO')
UEND
* Copy multiple files
COPY FROM(ALL) TO(D1,D2,D3)
COPY FROM(ALL) TO(D4) USE
USTART
   INCLUDE COND=(16,2,CH,EQ,C'CA')
UEND
* California branches profit analysis
RANGE FROM(CADASD) ON(28,6,PD) HIGHER(-1500) LOWER(+8000)
* Branches with less than 32 employees
RANGE FROM(ALL) ON(18,4,ZD) LOWER(32)
* Print profit, employees, and city for each Colorado branch
DISPLAY FROM(CODASD) LIST(013) ON(28,6,PD) ON(18,4,ZD) ON(1,15,CH)
* Print a report for the Colorado branches
DISPLAY FROM(CODASD) LIST(014) -
    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')
DISPLAY FROM(DAPUBS) LIST(015) -
    TITLE('Books for individual publishers') PAGE -
    HEADER('Title of book') ON(1,35,CH) -
    HEADER('Price of book') ON(170,4,BI,C1,F'$') -
    BTITLE('Publisher:') BREAK(106,4,CH) -
    BAVERAGE('Average for this publisher') -
    BTOTAL('Total for this publisher') -
    AVERAGE('Average for all publishers') -
    TOTAL('Total for all publishers')
* Print the count of books in use from each publisher
OCCUR FROM(BKIN) LIST(016) BLANK -
    TITLE('Books from Publishers') DATE(DMY.) -
    HEADER('Publisher') HEADER('Books in use') -
    ON(106,4,CH) ON(VALCNT)
* Separate output containing records for publishers
* with more than 4 books in use
  SELECT FROM(BKIN) TO(BKOUT) ON(106,4,CH) HIGHER(4)
/*
/&
* $$ EOJ
```

ICETOOL can be called directly, as in the example above, or from a program using a parameter list to which ICETOOL can additionally return information for further processing.

What kind of reports can I produce with ICETOOL?

Professor Sort says ...

With DFSORT/VSE's ICETOOL utility, you can produce reports quickly and easily. You can produce detailed reports with section headings and statistics, page headings and statistics, and a summary page with headings and statistics. You can produce reports that identify and count unique values, duplicate values, or values that occur a specific number of times. You can also use ICETOOL to create a report showing the DFSORT/VSE installation defaults selected at your site.

Here's an example of a complete ICETOOL job. It uses the DISPLAY operator to produce a monthly revenue report:

```
// JOB EXAMP JOBA,PROGRAMMER
// LIBDEF PHASE,SEARCH=(PRD2,PRSMPROD)
// DLBL VSESPUC,'VSESP.USER.CATALOG', VSAM
// DLBL IN2,'SORT.BRANCH', VSAM,DISP=(OLD,KEEP),CAT=VSESPUC
// EXEC ICETOOL,SIZE=128K
 DEFINE NAME(IN2) TYPE(F) LENGTH(33)
* Print monthly revenue report
 DISPLAY FROM(IN2) LIST(LST) DATE -
   TITLE('Monthly Revenue Report') PAGE -
   HEADER('Location') ON(1,15,CH) -
   HEADER('Revenue') ON(22,6,PD,A1) -
   HEADER('Profit/Loss') ON(28,6,PD,A1) -
   TOTAL('Totals') AVERAGE('Averages') BLANK
/*
/&
```

Here's the report in SYSLST that might be produced by this DISPLAY operator:

```
 10/27/95        Monthly Revenue Report        -1-

 Location                          Revenue             Profit/Loss
 -------------------  --------------------   --------------------
 Los Angeles                        22,530                 -4,278
 San Francisco                      42,820                  6,832
 Fort Collins                       12,300                 -2,863
 Sacramento                         42,726                  8,276
 ...                                   ...                    ...

 Totals                            329,637                 50,665

 Averages                           27,469                  4,222
```

Notice that the numbers have commas and negative signs. The numbers look like this because the "A1" editing mask was specified for both the revenue and the profit/loss data. ICETOOL has 33 different editing masks encompassing many of the numeric notations used throughout the world.

Here's another example of ICETOOL control statements. In this example, ICETOOL's OCCUR operator is used to identify userids that appear more than 4 times in a data set as possible system intruders:

```
OCCURS FROM(FAILURS) LIST(LST) BLANK -
   PAGE TITLE('Possible System Intruders) DATE(DM4.) -
   HEADER('Userid') HEADER('Logon Failures') -
   ON(23,8,CH)      ON(VALCNT) -
   HIGHER(4)
```

Here's the report in SYSLST that might be produced by this OCCUR operator:

```
 - 1 -   Possible System Intruders        27.10.1995

 Userid      Logon Failures
 --------   ---------------
 B7234510                 5
 D9853267                11
 ...                    ...
```

What is DFSORT/VSE OME and how can it help me?

Professor Sort says ...

With Online Message Explanations (OME), you request that an explanation of a DFSORT/VSE message be displayed on the console of your display station. OME makes it easier for you to access information about messages you receive and can help you diagnose and correct errors more quickly. You can use the DFSORT/VSE OME just as you would use the VSE/ESA OME.

Should I upgrade from Sort/Merge V2R5 to DFSORT/VSE?

Professor Sort says ...

If you need a high performance data arranger for the VSE/ESA operating system, you should seriously consider upgrading from Sort/Merge V2R5 to DFSORT/VSE V3R4. DFSORT/VSE is based on and is a replacement for the previous IBM DOS/VS-VM/SP Sort/Merge Version 2 Release 5 program product.

DFSORT/VSE provides very significant performance and productivity improvements over Sort/Merge V2R5 such as Year 2000 Features, National Language Support, interaction with File Management Systems, 31-bit addressing, secondary allocation of VSAM-managed SAM work files, dataspace sorting, getvis sorting, abend recovery, reduction in external work space used, ICETOOL reporting and analysis, OUTREC and INCLUDE/OMIT enhancements, and so on.

DFSORT/VSE is upward compatible with Sort/Merge V2R5 with the following exceptions:

- DFSORT/VSE can run only in the VSE/ESA environment.
- DFSORT/VSE will not support the CMS/DOS environment of VM/ESA. DFSORT/CMS is available for the CMS environment to help satisfy user's needs for a Sort product.
- DFSORT/VSE V3R2-V3R4 handle the END control statement differently than DFSORT/VSE V3R1 and Sort/Merge V2R5 did. See "Product updates" on the DFSORT/VSE website for details.

Upgrading from Sort/Merge V2R5 to DFSORT/VSE does not require changes to your jobs or recompilations of your programs.

How does DFSORT/VSE V3R4 performance compare to Sort/Merge V2R5?

Professor Sort says ...

If you use Sort/Merge V2R5, you should seriously consider upgrading to DFSORT/VSE. DFSORT/VSE is based on and is a replacement for the previous IBM DOS/VS-VM/SP Sort/Merge Version 2 Release 5 program product. Upgrading from Sort/Merge V2R5 to DFSORT/VSE does not require changes to your jobs or recompilations of your programs.

DFSORT/VSE V3R4 provides very significant performance improvements over Sort/Merge V2R5 using techniques such as dataspace sorting, getvis sorting and 31-bit addressing, which are not available with Sort/Merge V2R5.

Dataspace sorting

V3R4 provides significant improvements over Sort/Merge V2R5 for fixed-length record (FLR) and variable-length record (VLR) jobs that use dataspace sorting with and without external DASD.

For dataspace sorting **without** external work space, reductions up to 35% for elapsed time, 10% for CPU time and 71% for SIO counts were observed for **FLR**, and reductions up to 26% for elapsed time, 14% for CPU time and 72% for SIO counts were observed for **VLR**.

For dataspace sorting **with** external work space, reductions up to 56% for elapsed time, 38% for CPU time and 64% for SIO counts were observed for **FLR**, and reductions up to 58% for elapsed time, 31% for CPU time and 75% for SIO counts were observed for **VLR**.

Getvis sorting

V3R4 provides significant improvements over Sort/Merge V2R5 for FLR and VLR jobs that use getvis sorting with and without external DASD.

For getvis sorting **without** external work space, reductions up to 36% for elapsed time, 10% for CPU time and 71% for SIO counts were observed for **FLR**, and reductions up to 32% for elapsed time, 11% for CPU time and 72% for SIO counts were observed for **VLR**.

For getvis sorting **with** external work space, reductions up to 56% for elapsed time, 40% for CPU time and 64% for SIO counts were observed for **FLR**, and reductions up to 58% for elapsed time, 31% for CPU time and 75% for SIO counts were observed for **VLR**.

Copy

V3R4 provides significant improvements over Sort/Merge V2R5 for FLR and VLR copy applications with DASD and tape input and output.

For copy applications using **DASD** for input and output, reductions up to 39% for elapsed time, 71% for CPU time and 97% for SIO counts were observed for **FLR**, and reductions up to 35% for elapsed time, 75% for CPU time and 97% for SIO counts were observed for **VLR**.

Merge

V3R4 provides significant improvements over Sort/Merge V2R5 for FLR and VLR merge applications with DASD and tape input and output.

For merge applications using **DASD** for input and output, reductions up to 36% for elapsed time, 66% for CPU time and 97% for SIO counts were observed for **FLR**, and reductions up to 31% for elapsed time, 74% for CPU time and 97% for SIO counts were observed for **VLR**.

Reminder

Test cases used were selected to show performance improvements possible when sorting large fixed-length record and large variable-length record files and were run in a laboratory environment. The actual performance characteristics that may be experienced by any specific user or for any specific file are dependent on many factors. Consequently, the results may differ from user to user. For a complete description of the test environment and results, please see the DFSORT/VSE V3R4 Announcement Letter (298-152).

If DFSORT/VSE abends when called by COBOL, how can I get a dump?

Professor Sort says ...

The new DIAGINF installation option allows you to request diagnostic information for all jobs or a specific job, regardless of the options in effect at run-time. Diagnostic information consists of diagnostic messages, and a dump if an abend occurs.

This option is useful when DFSORT/VSE is called by an application, such as COBOL, and the COBOL source code is missing or unavailable. In these cases, you are unable to change anything in the application, including the DFSORT/VSE control statements. With the new DIAGINF option, you don't need to change the application. You can force DFSORT/VSE diagnostic messages and a dump if an abend occurs.

Can DFSORT/VSE interact with my File Management System?

Professor Sort says ...

You can use DFSORT/VSE's FMS=YES installation parameter to specify that you want DFSORT/VSE to attempt to interact with the File Management System installed at your site. FMS=YES can allow DFSORT/VSE to take advantage of File Management System benefits such as:

- Dynamic logical and physical device assignment
- Dynamic primary and secondary extent allocation
- Output file truncation of unused space

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".

FMS=NO is the IBM-supplied default for the FMS installation parameter. It specifies that DFSORT/VSE is not to attempt to interact with a File Management System.

What is DFSORT/VSE's National Language Support?

Professor Sort says ...

DFSORT/VSE's National Language Support includes the following features:

- **Cultural sort and merge:** DFSORT/VSE allows you to select an active locale at installation or run-time and produces 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 allows you to select an active locale at installation or run-time and includes or omits records for output according to the collating rules defined in the active locale. This provides filtering for single- or multi-byte character data based on defined collating rules which retain the cultural and local characteristics of a language.
- **Translation of DFSORT/VSE messages:** DFSORT/VSE provides the ability for easy translation of messages into different languages. Flexible positioning of variables within a message is supported. And, DFSORTVSE is shipped with both English and Japanese DFSORT/VSE messages.
- **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.

What is dataspace sorting and how can it help me?

Professor Sort says ...

DFSORT/VSE provides performance improvements for fixed-length record (FLR) and variable-length record (VLR) sorts through the use of dataspace sorting. Dataspace sorting is a DFSORT/VSE capability that uses data space available with VSE/ESA systems on ESA/370 and ESA/390 processors in place of intermediate work space for sort applications. A data space is an area of contiguous storage backed by processor or auxiliary storage.

Dataspace sorting enables a greater percentage of sort applications to be processed completely within virtual storage (incore sort), without the need to write intermediate data to external DASD work files. Incore sorts are the most efficient of all sort applications, reducing CPU and elapsed time as well as I/O and channel usage.

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

DFSORT/VSE V3R1 provided the initial dataspace sorting support for fixed-length records (FLR). V3R2 provided significant improvements for jobs that use dataspace sorting with external DASD. V3R3 provided support for variable-length records (VLR) and improvements for jobs that use dataspace sorting with and without external DASD.

Now V3R4 provides further significant improvements for FLR and VLR jobs that use dataspace sorting with and without external DASD.

For dataspace sorting **without** external work space, reductions up to 23% for elapsed time, 7% for CPU time and 78% for SIO counts were observed for **FLR**, and reductions up to 21% for elapsed time, 5% for CPU time and 69% for SIO counts were observed for **VLR**.

For dataspace sorting **with** external work space, reductions up to 13% for elapsed time, 23% for CPU time and 50% for SIO counts were observed for **FLR**, and reductions up to 29% for elapsed time, 5% for CPU time and 59% for SIO counts were observed for **VLR**.

DFSORT/VSE's management of data spaces for dataspace sorting also includes the following:

- When the DSPSIZE=MAX parameter is used, DFSORT/VSE dynamically determines the maximum amount of data space to be used for dataspace sorting. DFSORT/VSE bases its data space usage on the size of the available data space, the amount of real storage, and the paging activity of the system. DSPSIZE=MAX is especially useful for large, critical jobs.
- When the DSPSIZE=n parameter is used, n can't be larger than 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. If n is larger than the total amount of available space at run-time, dataspace sorting is not used. DFSORT/VSE decreases the value of n, if required, to avoid real storage overcommitment.

Note that dataspace sorting and getvis sorting provide similar performance. Getvis sorting uses only the partition virtual storage and is preferable when an application should not depend on the availability of virtual storage for data space. Dataspace sorting uses global system resources and is preferable when DFSORT/VSE is called from an application that uses the partition virtual storage extensively.

What is getvis sorting and how can it help me?

Professor Sort says ...

DFSORT/VSE provides performance improvements for fixed-length record (FLR) and variable-length record (VLR) sorts through the use of getvis sorting. Getvis sorting is similar to dataspace sorting except that it uses the partition GETVIS area for virtual storage instead of dataspace. A GETVIS area is a large contiguous area of partitioned main storage backed by processor or auxiliary storage and can exploit the available main storage above 16-megabyte virtual.

Just like dataspace sorting, getvis sorting enables a greater percentage of sort applications to be processed completely within virtual storage (incore sort), without the need to write intermediate data to external DASD work files. Incore sorts are the most efficient of all sort applications, reducing CPU and elapsed time as well as I/O and channel usage.

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

DFSORT/VSE V3R1 provided the initial getvis sorting support for FLR and VLR. V3R2 provided significant improvements for jobs that use getvis sorting with and without external DASD. V3R3 provided additional improvements for jobs that use getvis sorting without external DASD.

Now V3R4 provides further significant improvements for FLR and VLR jobs that use getvis sorting with and without external DASD.

For getvis sorting **without** external work space, reductions up to 23% for elapsed time, 6% for CPU time and 78% for SIO counts were observed for **FLR**, and reductions up to 22% for elapsed time, 4% for CPU time and 69% for SIO counts were observed for **VLR**.

For getvis sorting **with** external work space, reductions up to 11% for elapsed time, 8% for CPU time and 50% for SIO counts were observed for **FLR**, and reductions up to 30% for elapsed time, 7% for CPU time and 59% for SIO counts were observed for **VLR**.

DFSORT/VSE's management of GETVIS areas for getvis sorting also includes the following:

- When the GVSIZE=MAX parameter is used, DFSORT/VSE dynamically determines the maximum amount of GETVIS area to be used for getvis sorting. DFSORT/VSE bases its GETVIS area usage on the size of the available GETVIS area, the amount of reserved 24-bit GETVIS area (GVSRLOW parameter) and 31-bit GETVIS area (GVSRANY parameter) for the operating system and user application, the amount of real storage, and the paging activity of the system. GVSIZE=MAX is especially useful for large, critical jobs.
- When the GVSIZE=n parameter is used, n bytes of GETVIS area will be used for getvis sorting. DFSORT/VSE decreases the value of n, if necessary, to avoid real storage overcommitment.

Note that dataspace sorting and getvis sorting provide similar performance. Getvis sorting uses only the partition virtual storage and is preferable when an application should not depend on the availability of virtual storage for data space. Dataspace sorting uses global system resources and is preferable when DFSORT/VSE is called from an application that uses the partition virtual storage extensively.

How can I list my site's DFSORT/VSE installation defaults?

Professor Sort says ...

With DFSORT/VSE's ICETOOL utility, you can produce a report listing your site's current DFSORT/VSE installation defaults any time you want to.

Here's an example of a complete ICETOOL job to do it. It uses the DEFAULTS operator to produce the report:

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

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

If the installation default value for a parameter is different from the IBM-supplied default value, the IBM-supplied default value is shown to the right of the installation default value.

Here's an example of what part of the output from an ICETOOL DEFAULTS job might look like:

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

 * ONLY SHOWN IF DIFFERENT FROM THE SPECIFIED DEFAULT

PARAMETER      INSTALLATION DEFAULT     IBM-SUPPLIED DEFAULT *
----------     --------------------     --------------------
CHALT          NOCHALT
DIAG           DIAG                     NODIAG
DUMP           NODUMP
EQUALS         NOEQUALS
ERASE          NOERASE
VERIFY         VERIFY
ALTSEQ         *** SEE BELOW ***
DIAGINF        NONE
DSPSIZE        0
FMS            NO
GVSIZE         256K                     0
GVSRANY        32K
...
```

How does DFSORT/VSE take advantage of 31-bit addressing?

Professor Sort says ...

DFSORT/VSE takes advantage of 31-bit addressing by using the additional virtual storage above 16-megabytes virtual for work space and by allowing many of its modules to be loaded above 16-megabyte virtual.

- **Work space above 16-megabyte virtual**

  Sort/Merge V2R5 can only use storage **below** 16-megabyte virtual for work space. When that limited amount of space is used up, Sort/Merge V2R5 has to use external work space (DASD) to complete the sort.

  DFSORT/VSE can use storage **above and below** 16-megabyte virtual for work space. The additional storage available above 16-megabyte virtual makes it possible for DFSORT/VSE to complete the sort with less or even no external work space (DASD), which translates into reduced elapsed time, CPU time and SIO counts.

- **Modules above 16-megabyte virtual**

  With Sort/Merge V2R5, all modules must be loaded **below** 16-megabyte virtual in the Program Area or 24-bit SVA.

  With DFSORT/VSE, eligible reenterable modules can be loaded **above** 16-megabyte virtual. This gives the system more capacity for calling programs and user exits, thus easing virtual storage contraints below 16-megabyte virtual.

You can help improve virtual storage contraint relief by loading your user exits above 16-megabyte virtual.

Where can I find information to help me use DFSORT/VSE more effectively?

Professor Sort says ...

The following are sources of information and examples available from IBM that can help you use DFSORT/VSE's many features more effectively.

The DFSORT/VSE website

For papers, online books, examples and more, visit the DFSORT/VSE website at URL:

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

The DFSORT/VSE library

To get the most out of DFSORT/VSE, every programmer who uses it should have the following publications available:

- *Getting Started with DFSORT/VSE (SC26-7101)* is an excellent tutorial about DFSORT/VSE control statements and ICETOOL operators. The material is taught using numerous examples which can be run using sample data sets shipped with DFSORT/VSE.
- *DFSORT/VSE General Information (GC26-7039)* and *DFSORT/VSE Application Programming Guide (SC26-7040)* are important reference sources.
- *DFSORT/VSE Installation and Tuning Guide (GC26-7041)* provides valuable information about tuning DFSORT/VSE.

Programmers who install DFSORT/VSE should have *DFSORT/VSE Installation and Tuning Guide (GC26-7041)* available to them as a reference source.

A complete set of DFSORT/VSE publications can be ordered using order number *SBOF-6130*.

You can access all of the DFSORT/VSE books online from the DFSORT/VSE website.

Online DFSORT/VSE books

Online books provide capabilities (for example, search) not provided by traditional books. All of the DFSORT/VSE books can be accessed online in one of two ways:

- From the "Publications" link on the DFSORT/VSE website. Bookmark these links for faster access.
- Install the displayable softcopy publications available in the *DFSORT/VSE Online Product Library (SK2T-8730)* or *IBM Online Library VSE Collection (SK2T-0060)*, shipped on CD-ROM.

How can DFSORT/VSE help with the Year 2000 challenge?

Professor Sort says ...

The widespread use of two-digits to represent years can result in significant data processing problems. For example, the normal ordering of 00 before 99 is incorrect when 00 represents 2000 and 99 represents 1999. DFSORT/VSE's Year 2000 features provide tools that can help you correctly process a wide variety of dates for this century and the coming ones.

DFSORT/VSE provides powerful Year 2000 features that allow you to sort, merge, compare and transform character, zoned decimal and packed decimal dates with two-digit years according to a specified sliding or fixed century window.

You can handle two-digit year date fields, and their special indicators like zeros and nines, in the following ways:

- Set the appropriate **century window** for your applications and use it to interpret the years (yy) correctly when you sort, merge, compare or transform two-digit year dates.

  For example, set a century window of 1915-2014 or 1950-2049.

- **Order** character, zoned decimal or packed decimal two-digit year dates with the SORT and MERGE statements. For example, order 980622 (representing June 6th, 1998) before 000622 (representing June 6th, 2000) in ascending sequence, or order 000622 before 980622 in descending sequence.

  DFSORT/VSE's **full date formats** (Y2T, Y2U, Y2V, Y2W, Y2X and Y2Y) make it easy to sort or merge any type of yyx...x or x...xyy date (for example, yyq, yymm, yyddd, yymmdd, qyy, mmyy, dddyy or mmddyy). DFSORT/VSE's **year formats** (Y2C, Y2Z, Y2S, Y2P, Y2D and Y2B) are available if you need to sort or merge special dates (for example, ddmmyy) or year fields (yy).

- **Select** records by **comparing** character, zoned decimal or packed decimal two-digit year date fields to date constants or other date fields with the INCLUDE and OMIT statements. For example, select records with a date field between January 1st, 1998 and December 31st, 2003.

  DFSORT/VSE's full date formats make it easy to compare any type of yyx...x or x...xyy date field to a date constant or another date field. DFSORT/VSE's year formats are avalable if you need to compare years.

- **Transform** character, zoned decimal or packed decimal two-digit year dates to character four-digit year dates with or without separators, or transform packed decimal two-digit year dates to packed decimal four-digit year dates, with DFSORT/VSE's OUTREC statement. For example, transform P'99015' to C'1999015', C'1999/015' or P'1999015'.

  DFSORT/VSE's full date formats make it easy to transform any type of yyx...x or x...xyy date. DFSORT/VSE's year formats are available if you need to transform special dates or year fields.

In addition, you can use DFSORT/VSE's Year 2000 features with COBOL, either automatically with **COBOL MLE** or explicitly without MLE.

These DFSORT/VSE enhancements allow you to continue to use two-digit year dates for sorting, merging and comparing, and help you to change from using two-digit year dates to using four-digit year dates as appropriate.

How can I use INCLUDE/OMIT with two-digit year dates?

Professor Sort says ...

INCLUDE and OMIT are frequently used to select records with a field above a certain year or within a certain range of years. For example:

```
INCLUDE COND=(21,2,CH,GE,C'85')
```

might be used to include records where the **two-digits** in positions 21-22 correspond to a year greater than 1985. This will work fine for years between 1985 (C'85') and 1999 (C'99'). But it will not work for years from 2000 (C'00') on, because C'00' is less than C'85'. DFSORT/VSE's Year 2000 features can handle these types of situations. By using the appropriate Y2x format, you can do INCLUDE and OMIT logic that works correctly within the century window you choose. For example, we could change the INCLUDE statement above to the following using a century window of 1960-2059:

```
OPTION Y2PAST=1960
INCLUDE COND=(21,2,Y2C,GE,Y'85')
```

With DFSORT/VSE's **full date formats,** you can handle INCLUDE and OMIT conditions for CH, ZD and PD dates such as yymmdd, yyddd, yymm, yyq, mmddyy, dddyy, mmyy and qyy. For example, you could include dates

for a P'yymm' date field starting in position 18 for values from January, 1992 to March, 2005 using the following INCLUDE statement:

```
INCLUDE COND=(15,3,Y2V,GE,Y'9201',AND,
              15,3,Y2V,LE,Y'0503')
```