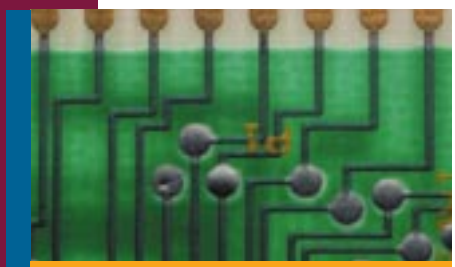

BY FRANK L. YAEGER
AND HOLLY YAMAMOTO-SMITH

Using DFSORT (MVS and VSE) Features From COBOL Applications

DFSORT's productivity features can be used from COBOL applications for both MVS and VSE to simplify and even eliminate COBOL logic, improve performance and handle two-digit year dates.

WHEN customers wanted to know how to use the Year 2000 features of DFSORT/MVS and DFSORT/VSE with their COBOL applications, IBM's DFSORT and COBOL developers got together, figured out some solutions and shipped enabling PTFs. As it turns out, these solutions can be applied to other DFSORT features besides Year 2000.

This article discusses some techniques for using DFSORT's productivity features from COBOL applications for both MVS and VSE. Examples demonstrate how DFSORT's control statements can be used in many ways, including for multiple sorts within a COBOL application. These control statements can simplify and even eliminate COBOL logic, improve performance, and further handle two-digit year dates.

If you're not familiar with the DFSORT features discussed in this article, refer to your DFSORT books or check the "References" section for other sources of information.

PRODUCT LEVELS

The examples used in this article require the following product levels:

- DFSORT/MVS R13 with PTFs UN90139 and UQ05520
- DFSORT/VSE V3R3 or DFSORT/VSE V3R2 with PTF UN99635
- LE/VSE COBOL at the LE/VSE 1.4 level with PTF UQ05847

COBOL SORT VERB AND YEAR 2000 OVERRIDES

You may not be aware of it, but when you code a SORT verb in a COBOL application, you are telling COBOL to call DFSORT (or another sort product). A typical SORT verb looks like the following from Figure 1:

```
SORT SORT-FILE1
  ON ASCENDING KEY
  SORT1-YY, SORT1-MMDD
  USING INDS1
  GIVING OUTDS1.
```

This causes COBOL to generate control statements containing information for DFSORT. COBOL passes these control statements to DFSORT in a parameter list. You can find the control statements COBOL generates for an application by looking at DFSORT's control statements listing. For the SORT verb above, COBOL generates the following SORT statement:

```
SORT FIELDS=(0007,0002,CH,A,0003,0004,CH,A)
```

COBOL has translated the SORT1-YY and SORT1-MMDD fields into the position, length and format information required by DFSORT. To COBOL, the SORT1-YY and SORT1-MMDD fields are character format. However, let's say you want to treat SORT1-YY as a two-digit year and apply your installation's default century window of 1970-2069 to it. How can you do that? The answer is to override the SORT statement that COBOL generates with a different SORT statement — one with a two-digit year format. In this case, you'd want to use

```
SORT FIELDS=(7,2,Y2C,A,3,4,CH,A)
```

where Y2C is DFSORT's two-digit year character format. Now all you have to do is get COBOL to use your SORT control statement instead of its SORT control statement.

Figure 1: COBOL Source for MVS

```
CBL FASTSRT
ID DIVISION.
  PROGRAM-ID.  CALLSORT.
ENVIRONMENT DIVISION.
  INPUT-OUTPUT SECTION.
    FILE-CONTROL.
       SELECT  INDS1 ASSIGN TO INDS1.
       SELECT  OUTDS1 ASSIGN TO OUTDS1.
       SELECT  SORT-FILE1 ASSIGN TO SORTFL1.
       SELECT  INDS2 ASSIGN TO INDS2.
       SELECT  OUTDS2 ASSIGN TO OUTDS2.
       SELECT  SORT-FILE2 ASSIGN TO SORTFL2.
DATA DIVISION.
  FILE SECTION.
    FD  INDS1 RECORD CONTAINS 160 CHARACTERS
        LABEL RECORD STANDARD BLOCK 27840
        DATA RECORDS ARE INDS1-RECORD.
    01  INDS1-RECORD.
      05       FILLER              PIC X(2).
      05       INDS1-MMDDYY        PIC X(6).
      05       FILLER              PIC X(1).
      05       INDS1-DEPT          PIC X(3).
      05       FILLER              PIC X(148).
    FD  OUTDS1 RECORD CONTAINS 160 CHARACTERS
        LABEL RECORD STANDARD BLOCK 27840
        DATA RECORDS ARE OUTDS1-RECORD.
    01  OUTDS1-RECORD.
      05       FILLER              PIC X(160).
    SD  SORT-FILE1 RECORD CONTAINS 160 CHARACTERS
        DATA RECORD SORT1-RECORD.
    01  SORT1-RECORD.
      05       FILLER              PIC X(2).
      05       SORT1-MMDD          PIC X(4).
      05       SORT1-YY            PIC X(2).
      05       FILLER              PIC X(152).
    FD  INDS2 RECORD CONTAINS 80 CHARACTERS
        LABEL RECORD STANDARD BLOCK 27840
        DATA RECORDS ARE INDS2-RECORD.
    01  INDS2-RECORD.
      05       FILLER              PIC X(12).
      05       INDS2-YYDDD         PIC X(5).
      05       INDS2-YYMMDD        PIC X(6).
      05       INDS2-PRODUCT       PIC X(20).
      05       FILLER              PIC X(37).
    FD  OUTDS2 RECORD CONTAINS 80 CHARACTERS
        LABEL RECORD STANDARD BLOCK 27840
        DATA RECORDS ARE OUTDS1-RECORD.
    01  OUTDS2-RECORD.
      05       FILLER              PIC X(80).
    SD  SORT-FILE2 RECORD CONTAINS 80 CHARACTERS
        DATA RECORD SORT2-RECORD.
    01  SORT2-RECORD.
      05       FILLER              PIC X(12).
      05       SORT2-YYDDD         PIC X(5).
      05       SORT2-YYMMDD        PIC X(6).
      05       SORT2-PRODUCT       PIC X(20).
      05       FILLER              PIC X(37).
  WORKING-STORAGE SECTION.
PROCEDURE DIVISION.
  MASTER SECTION.
       SORT SORT-FILE1
         ON ASCENDING KEY
         SORT1-YY, SORT1-MMDD
         USING INDS1
         GIVING OUTDS1.
         IF SORT-RETURN > 0
         DISPLAY "FIRST SORT FAILED".
       SORT SORT-FILE2
         ON ASCENDING KEY
         SORT2-PRODUCT, SORT2-YYDDD, SORT2-YYMMDD
         USING INDS2
         GIVING OUTDS2.
         IF SORT-RETURN > 0
         DISPLAY "SECOND SORT FAILED".
       STOP RUN.
```

Figure 2: COBOL Source for VSE

```
CBL FASTSRT
ID DIVISION.
  PROGRAM-ID.  CALLSORT.
ENVIRONMENT DIVISION.
  INPUT-OUTPUT SECTION.
    FILE-CONTROL.
       SELECT INDS1 ASSIGN TO INDS1.
       SELECT OUTDS1 ASSIGN TO OUTDS1.
       SELECT SORT-FILE1 ASSIGN TO SORTFL1.
       SELECT INDS2 ASSIGN TO INDS2.
       SELECT OUTDS2 ASSIGN TO OUTDS2.
       SELECT SORT-FILE2 ASSIGN TO SORTFL2.
DATA DIVISION.
  FILE SECTION.
    FD  INDS1 RECORD CONTAINS 160 CHARACTERS
        LABEL RECORD STANDARD BLOCK 27840
        DATA RECORDS ARE INDS1-RECORD.
    01  INDS1-RECORD.
      05       FILLER              PIC X(2).
      05       INDS1-MMDDYY        PIC X(6).
      05       FILLER              PIC X(1).
      05       INDS1-DEPT          PIC X(3).
      05       FILLER              PIC X(148).
    FD  OUTDS1 RECORD CONTAINS 160 CHARACTERS
        LABEL RECORD STANDARD BLOCK 27840
        DATA RECORDS ARE OUTDS1-RECORD.
    01  OUTDS1-RECORD.
      05       FILLER              PIC X(160).
    SD  SORT-FILE1 RECORD CONTAINS 160 CHARACTERS
        DATA RECORD SORT1-RECORD.
    01  SORT1-RECORD.
      05       FILLER              PIC X(2).
      05       SORT1-MMDD          PIC X(4).
      05       SORT1-YY            PIC X(2).
      05       FILLER              PIC X(152).
    FD  INDS2 RECORD CONTAINS 80 CHARACTERS
        LABEL RECORD STANDARD BLOCK 27840
        DATA RECORDS ARE INDS2-RECORD.
    01  INDS2-RECORD.
      05       FILLER              PIC X(12).
      05       INDS2-YYDDD         PIC X(5).
      05       INDS2-YYMMDD        PIC X(6).
      05       INDS2-PRODUCT       PIC X(20).
      05       FILLER              PIC X(37).
    FD  OUTDS2 RECORD CONTAINS 80 CHARACTERS
        LABEL RECORD STANDARD BLOCK 27840
        DATA RECORDS ARE OUTDS1-RECORD.
    01  OUTDS2-RECORD.
      05       FILLER              PIC X(80).
    SD  SORT-FILE2 RECORD CONTAINS 80 CHARACTERS
        DATA RECORD SORT2-RECORD.
    01  SORT2-RECORD.
      05       FILLER              PIC X(12).
      05       SORT2-YYDDD         PIC X(5).
      05       SORT2-YYMMDD        PIC X(6).
      05       SORT2-PRODUCT       PIC X(20).
      05       FILLER              PIC X(37).
  WORKING-STORAGE SECTION.
PROCEDURE DIVISION.
  MASTER SECTION.
     MOVE "SYSIPT" TO SORT-CONTROL.
     SORT SORT-FILE1
       ON ASCENDING KEY
        SORT1-YY, SORT1-MMDD
        USING INDS1
        GIVING OUTDS1.
        IF SORT-RETURN > 0
        DISPLAY "FIRST SORT FAILED".
     MOVE "SYSIPT" TO SORT-CONTROL.
     SORT SORT-FILE2
        ON ASCENDING KEY
        SORT2-PRODUCT, SORT2-YYDDD, SORT2-YYMMDD
        USING INDS2
        GIVING OUTDS2.
        IF SORT-RETURN > 0
          DISPLAY "SECOND SORT FAILED".
     STOP RUN.
```

Figure 3: JCL and Control Statements for MVS

```
//SUDFSP EXEC  PGM=ICETOOL
//*** ICETOOL SET UP STEP
//TOOLMSG      DD    SYSOUT=*
//DFSMSG       DD    SYSOUT=*
//TOOLIN DD *
*  SET UP DFSPARM DATASET FOR FIRST COBOL SORT
   COPY FROM(IN1) TO(OUT1)
*  SET UP DFSPARM DATASET FOR SECOND COBOL SORT
   COPY FROM(IN2) TO(OUT2)
//IN1 DD *
*  DFSORT CONTROL STATEMENTS FOR FIRST COBOL SORT
   SORT FIELDS=(7,2,Y2C,A,3,4,CH,A)
   OPTION Y2PAST=1961
*  (1) INCLUDE - SELECT RECORDS FOR CERTAIN DEPARTMENTS
   INCLUDE COND=(10,3,SS,EQ,C'M72,L92,J82')
//IN2 DD *
*  DFSORT CONTROL STATEMENTS FOR SECOND COBOL SORT
   SORT FIELDS=(24,20,CH,A,
                13,2,Y2C,A,15,3,CH,A,
                18,2,Y2C,A,20,4,CH,A)
   OPTION Y2PAST=1961
*  (2) OUTFIL - EXTRACT AND PRESENT INFORMATION
*  (3) OUTFIL - TRANSFORM TWO-DIGIT YEAR DATES
   OUTFIL FNAMES=OUTDS2,
     OUTREC=(5X,24,20,
             5X,13,2,Y2C,C'/',15,3,
             5X,18,2,Y2C,C'/',20,2,C'/',22,2,
             80:X)
//OUT1 DD DSN=&D1,UNIT=SYSDA,SPACE=(TRK,(5,5)),DISP=(,PASS)
//OUT2 DD DSN=&D2,UNIT=SYSDA,SPACE=(TRK,(5,5)),DISP=(,PASS)
//*
//COBSORT EXEC PGM=CALLSORT,REGION=4400K
//*** COBOL APPLICATION STEP
//SYSOUT     DD    SYSOUT=*
//INDS1      DD    DSN=...
//INDS2      DD    DSN=...
//OUTDS1 DD DSN=...
//OUTDS2 DD DSN=...
//DFSPARM DD DSN=&D1,DISP=(OLD,PASS),FREE=CLOSE
//DFSPARM DD DSN=&D2,DISP=(OLD,PASS),FREE=CLOSE
```

Figure 4: JCL and Control Statements for VSE

```
// DLBL INDS1,...
// DLBL OUTDS1,...
// DLBL INDS2,...
// DLBL OUTDS2,...
// LIBDEF ...
*  COBOL APPLICATION STEP
*    (1) INCLUDE - SELECT RECORDS FOR CERTAIN DEPARTMENTS
*    (2) OUTREC - EXTRACT AND PRESENT INFORMATION
*    (3) OUTREC - TRANSFORM TWO-DIGIT YEAR DATES
// EXEC  ,SIZE=180K
   SORT FIELDS=(7,2,Y2C,A,3,4,CH,A)
   OPTION Y2PAST=1961
   INCLUDE COND=(10,3,SS,EQ,C'M72,L92,J82')
/*
   SORT FIELDS=(24,20,CH,A,
                13,2,Y2C,A,15,3,CH,A,
                18,2,Y2C,A,20,4,CH,A)
   OPTION Y2PAST=1961
   OUTREC FIELDS=(5X,24,20,
                  5X,13,2,Y2C,C'/',15,3,
                  5X,18,2,Y2C,C'/',20,2,C'/',22,2,
                  27X)
/*
```

For MVS, you can supply your own control statements in a DFSPARM dataset. For VSE, you can supply your own control statements in SYSIPT. We'll show you examples of how to do both below.

ADDING CONTROL STATEMENTS

Overriding the SORT statement is necessary if you want to use two-digit year formats. But you might also want to use a fixed century window of 1961-2060 instead of your installation's default. To do that, you need to add this OPTION control statement:

```
OPTION Y2PAST=1961
```

You can supply your own OPTION statement in the same way you supply your own SORT statement. Extending this technique further, you can add control statements that take advantage of other capabilities of DFSORT. Control statements such as INCLUDE, OMIT, OUTREC, OUTFIL (DFSORT/MVS only), SUM and ALTSEQ can simplify and even eliminate COBOL logic, improve performance and further handle two-digit year dates.

COBOL APPLICATION EXAMPLES

Figures 1 and 2 show COBOL programs for MVS and VSE, respectively. Without any enhancements, these programs do two different sorts using various two-digit year date fields. Since COBOL treats these dates as character format, the sorts would give incorrect results for years spanning say 1990-2010.

> You may not be aware of it, but when you code a SORT verb in a COBOL application, you are telling COBOL to call DFSORT (or another sort product).

We're going to use DFSORT control statements to sort the two-digit year dates correctly. However, we're also going beyond that to help the COBOL programs do these new tasks without any COBOL logic:

1. Select records for certain departments using the substring search feature of the INCLUDE statement. This feature makes it easy to select records containing only specified constants, such as department numbers.

2. Extract and present information from the selected records using the OUTFIL statement for DFSORT/MVS and the OUTREC statement for DFSORT/VSE.

3. Transform two-digit year dates to four-digit year dates with "/" separators using the OUTFIL statement for DFSORT/MVS and the OUTREC statement for DFSORT/VSE.

Additional details on how COBOL logic can be replaced with DFSORT control statements can be found in the *DFSORT Tuning Guide*, SC26-3111 (DFSORT/MVS) and the *DFSORT/VSE Installation and Tuning Guide*, SC26-7041 (DFSORT/VSE).

USING DFSORT/MVS CONTROL STATEMENTS WITH COBOL

The key to using DFSORT/MVS control statements with COBOL is the DFSPARM dataset. DFSPARM can be used to supply control statements and parameters for any DFSORT application

regardless of how DFSORT is invoked (e.g., from JCL, COBOL, PL/I, DB2, and so on). The PARMDDN=ddname installation parameter can be used to change DFSPARM to a different ddname. If the COBOL program has a single SORT verb, you can simply supply control statements inline like the following:

```
//DFSPARM DD *
  SORT FIELDS=(...)
  INCLUDE COND=(...)
/*
```

However, if the COBOL program has multiple SORT verbs, the inline method will only allow you to supply one set of overriding control statements because DFSORT/MVS will use the inline dataset for all of the COBOL sort verbs. We need a way to use different control statements for different COBOL sort verbs. DFSORT R13 PTF UQ05520 solves the problem by allowing FREE=CLOSE to be used for DFSPARM datasets. With FREE=CLOSE, the first DFSPARM dataset in the JCL is used for the first SORT verb, the second DFSPARM dataset is used for the second SORT verb, and so on.

FREE=CLOSE doesn't work for inline datasets, so you'll need to use temporary or permanent datasets. Sequential datasets or partitioned members can be used. DFSORT's versatile ICETOOL utility can be used prior to the COBOL application to set up the desired control statements in the DFSPARM datasets. **Note:** JES3 does not allow the use of duplicate DFSPARM DD statements if the DD statements request JES3 or jointly-managed devices.

In Figure 3, we have JCL to copy inline DFSORT control statements to temporary DFSPARM datasets, followed by JCL to execute our COBOL program. The DFSPARM DD statements for the COBOL application look like the following:

```
//DFSPARM DD DSN=&D1,DISP=(OLD,PASS),FREE=CLOSE
//DFSPARM DD DSN=&D2,DISP=(OLD,PASS),FREE=CLOSE
```

Because of FREE=CLOSE, DFSPARM dataset &D1 will be used for the first SORT verb and DFSPARM dataset &D2 will be used for the second SORT verb.

In DFSPARM dataset &D1, the INCLUDE statement uses a substring search to only include records for departments M72, L92 and J82 from INDS1. The OPTION statement sets up a century window of 1961-2060. The SORT statement overrides the COBOL-generated SORT statement and sorts the selected records correctly using two-digit year formats and the specified century window. Figures 5 and 6 show some sample input and output, respectively, for the first SORT verb.

In DFSPARM dataset &D2, the SORT and OPTION statements sort the records from INDS2 correctly using two-digit year formats and a century window of 1961-2060. The OUTFIL statement extracts certain fields from the sorted records, rearranges them, transforms two-digit years to four-digit years and adds "/" separators to the dates. Figures 7 and 8 show some sample input and output, respectively, for the second SORT verb.

You may not be able to use the DFSPARM technique described above in some situations such as with JES3 or when the COBOL sorts are not always executed in the same order. In these cases, you may be able to use a technique described on the DFSORT home page that takes advantage of COBOL's SORT-CONTROL special register.

Note that unlike other DFSORT/MVS statements, OUTFIL can only be used successfully for an output dataset for which COBOL selects FASTSRT processing. When FASTSRT is selected, DFSORT processes the output dataset directly with OUTFIL. However, if NOFASTSRT is selected, COBOL uses an E35 exit to process the output dataset and this E35 does not pass the records to OUTFIL so OUTFIL cannot be used.

Figure 5: INDS1 Records

```
031500 J82
120100 C38
031698 J82
041403 M72
051898 B32
120600 L92
050560 J82
021598 J82
091802 L92
110665 M72
080799 L92
041303 M72
021001 B32
```

Figure 6: OUTDS1 Records

```
110665 M72
021598 J82
031698 J82
080799 L92
031500 J82
120600 L92
091802 L92
041303 M72
041403 M72
050560 J82
```

Figure 7: INDS2 Records

```
00012011025Wrench-53167
61051000813Hammer-00321
01213981105Hammer-00321
98071750921Wrench-53167
61051990612Hammer-00321
01155920317Wrench-53167
```

Figure 8: OUTDS2 Records

```
Hammer-00321    1961/051    1999/06/12
Hammer-00321    1961/051    2000/08/13
Hammer-00321    2001/213    1998/11/05
Wrench-53167    1998/071    1975/09/21
Wrench-53167    2000/012    2001/10/25
Wrench-53167    2001/155    1992/03/17
```

Performance Tips

Always specify COBOL's FASTSRT compile-time option to allow COBOL to use FASTSRT when possible, since having DFSORT process the datasets directly can improve performance significantly.

DFSORT and COBOL support system-determined optimum block size (SDB) for new output datasets. The use of SDB can improve performance significantly. To use SDB for a new output dataset, specify the BLOCK CONTAINS 0 clause in your COBOL program and do not specify BLKSIZE in your DD statement.

USING DFSORT/VSE CONTROL STATEMENTS WITH COBOL

If you refer back to Figure 2, you'll see the following COBOL statement before each SORT verb:

```
MOVE "SYSIPT" TO SORT-CONTROL.
```

This MOVE statement is the key to using DFSORT/VSE control statements with COBOL. You'll need to recompile (once) any COBOL program you add it to. This MOVE statement tells COBOL to read the information in SYSIPT. Specify this MOVE statement before each SORT verb. This allows you to supply DFSORT/VSE control statements directly in SYSIPT or indirectly using VSE Librarian members. We'll discuss both methods, starting with direct SYSIPT input.

Input Using SYSIPT

If the COBOL program has a single SORT verb, you can simply supply control statements directly in SYSIPT like the following:

```
// EXEC ,SIZE=180K
  SORT FIELDS=(...)
  INCLUDE COND=(...)
/*
```

If the COBOL program has multiple SORT verbs, you can add control statements to SYSIPT for each SORT verb. In Figure 4, we show how to execute our COBOL program using a set of DFSORT/VSE control statements in SYSIPT for each of its two SORT verbs.

In the SYSIPT input for the first SORT verb, the INCLUDE statement uses substring search to only include records for departments M72, L92 and J82 from INDS1. The OPTION statement sets up a century window of 1961-2060. The SORT statement overrides the COBOL generated SORT statement and sorts the selected records correctly using two-digit year formats and the specified century window. Again, Figures 5 and 6 show sample input and output, respectively, for the first SORT verb.

In the SYSIPT input for the second SORT verb, the SORT and OPTION statements sort the records from INDS2 correctly using two-digit year formats and a century window of 1961-2060. The OUTREC statement extracts certain fields from the sorted records, rearranges them, transforms two-digit years to four-digit years and adds "/" separators to the dates. Figures 7 and 8 show sample input and output, respectively.

When you override COBOL generated control statements at run-time, they must be in the following order if specified:

1. SORT or MERGE
2. SMS=
3. OPTION
4. Other DFSORT/VSE control statements (ALTSEQ, ANALYZE, INCLUDE, OMIT, INREC, OUTREC, SUM) in any order

Input Using VSE Librarian Members

The COBOL statement:

```
MOVE "SYSIPT" TO SORT-CONTROL.
```

also allows you to pass control statements in VSE Librarian members by specifying the following in SYSIPT:

```
// EXEC ,SIZE=180K
* $$ SLI MEM=memname,S=sort.sublib
/*
```

Replace memname with the name of the member that contains your DFSORT/VSE control statements. Replace sort.sublib with the name of the sublibrary containing memname.

Another way to pass control statements in VSE Librarian members is to use a MOVE statement that sets SORT-CONTROL to a member name (instead of SYSIPT). This tells COBOL to read the control statements in the specified member, so you don't need the SYSIPT statements shown above.

Performance Tips

If possible, use the FASTSRT compiler option for your COBOL applications. Check the *IBM COBOL for VSE/ESA Programming Guide* for FASTSRT requirements. With FASTSRT, DFSORT/VSE processes the input and output files instead of COBOL, which can result in significant performance improvements. COBOL application performance can also be improved by blocking your input and output records (see the BLOCK CONTAINS clause in the *IBM COBOL for VSE/ESA Language Reference*).

Blocking SAM files on disk or tape can enhance processing speed and minimize storage requirements. Be careful not to specify BLOCK CONTAINS 0. Be sure that the SAM input file blocksize matches the BLOCK value in the COBOL program. If a blocked SAM ESDS input file is used, ensure that it has the RECFM=(FB lrecl) attribute specified. For this case, you can specify an ESDS file in your COBOL program and BUFSP=nnnnnn in the DLBL statement to achieve better performance.

SUMMARY

Whether you're on MVS or VSE, you can use the techniques shown in this article to help your COBOL programs tap into the many features provided by DFSORT control statements. Year 2000 sorting, record selection, reports and more are all available to you.

If you have a suggestion for a future article related to DFSORT, please email it to one of the authors.

What is DFSORT's ICETOOL?

ICETOOL is a versatile, easy-to-use file processing and reporting utility included with DFSORT/MVS and DFSORT/VSE. It helps you to perform complex sorting, copying, reporting and analytical tasks using multiple files in a single job step. You can use ICETOOL on your own data or on data produced by VSE/POWER, DCOLLECT, DFSMSrmm, RACF and other products. The ICETOOL operators are: COPY, COUNT, DEFAULTS, DEFINE, DISPLAY, MODE, OCCUR, RANGE, SELECT, SORT, STATS, UNIQUE and VERIFY. For more information on ICETOOL, see the appropriate DFSORT application programming guide or home page.

REFERENCES

DFSORT/MVS home page: *www.ibm.com/storage/dfsortmvs/*
DFSORT/VSE home page: *www.ibm.com/storage/dfsortvse/*
DFSORT/MVS FTP site: *ftp://index.storsys.ibm.com/dfsort/mvs/*
DFSORT/VSE FTP site: *ftp://index.storsys.ibm.com/dfsort/vse/*
"Year 2000 Compliance: New DFSORT Features" by Michael H. Carroll, ***Technical Support***, October 1997
"Sorting Out Two-Digit Year Dates" by Frank Yaeger, ***Technical Support***, December 1996
"RACF and DFSORT Security Analysis Tools" by Mark Nelson and Frank Yaeger, ***Technical Support***, October 1997. ts

NaSPA member Frank Yaeger is an IBM senior programmer. He has spent the last 16 of his 28 years with IBM designing and implementing functional enhancements to DFSORT such as ICETOOL, OUTFIL and Year 2000 features. He is also responsible for the DFSORT home pages on the World Wide Web. Frank can be reached via email at fyaeger@vnet.ibm.com.

Holly Yamamoto-Smith is the DFSORT/VSE Team Leader for IBM. She has worked in the DFSORT organization for the past 12 years and has participated in development and marketing support for the DFSORT products. She can be reached via email at hys@vnet.ibm.com.