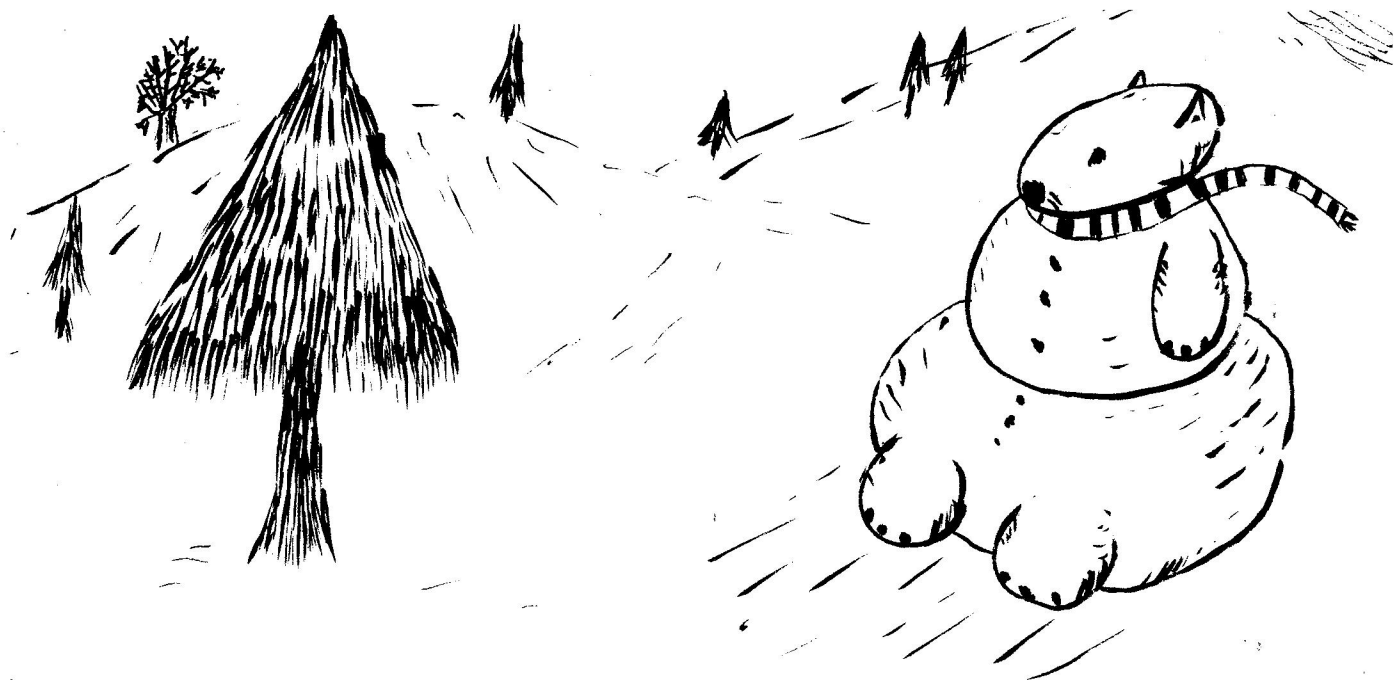
The Wombat

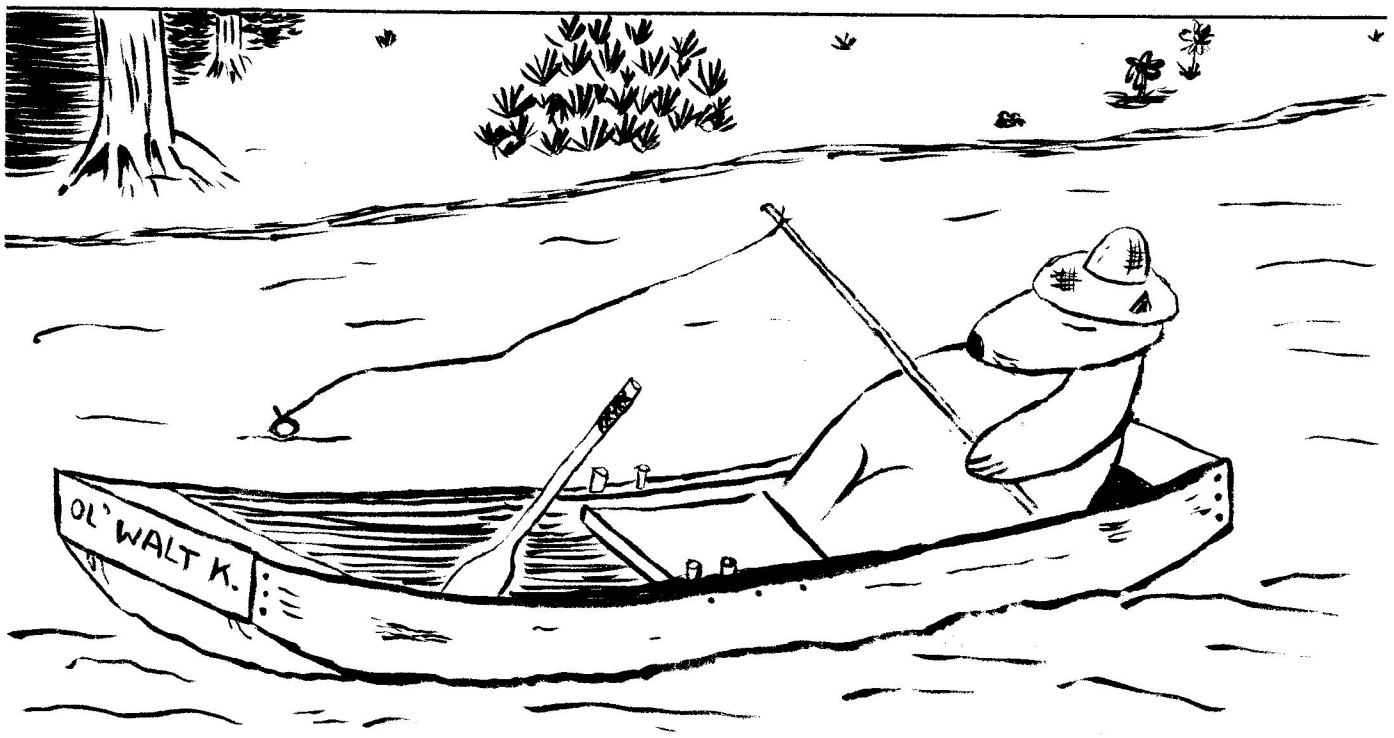
EXAMINER

APRIL

"Increases the Circulation of Anyone in America" Volume 6 Number 1 & 2

Winter/Spring 1985





Contributions

Contributions for the newsletter can be sent to either of the following addresses:

Editor, DATATRIEVE Newsletter
c/o DECUS U. S. Chapter
249 Northboro Road, BP02
Marlboro, MA 01752

Joe H. Gallagher, Ph. D.
DATATRIEVE Newsletter Editor
Director, Medical and
Research Computing
Cleveland Clinic Foundation
9500 Euclid Avenue
Cleveland, Ohio 44106

Letters and articles for publication are requested from members of the SIG. They may include helpful hints, inquiries to other users, reports on SIG business, summaries of SPRs submitted to Digital or other information for members of the DATATRIEVE SIG. Machine readable input is highly desirable and machine-to-machine transfer of material is preferred, but most anything legible will be considered. However, this newsletter is not a forum for job and/or head hunting, nor is commercialism appropriate.

Table of Contents

About the Cover

The cover for this double issue was drawn by Bart Lederman of Bankers Trust Company. Our snow Wombat is trying to get through the winter with a little fun. And he is fishin' in the bayou outside of New Orleans before the Spring Symposium.

DATATRIEVE Wish List - 1984 Fall Symposium

Bart Z. Lederman, Bankers Trust Company, New York, NY

A wish list is a method by which the users can get their requests for changes or enhancements in a product back directly to the persons in DEC who work on the product. The DATATRIEVE wish list is somewhat informal, being collected during the symposium up until Thursday night, when a copy is given to a DEC developer. DEC then responds during the SIG closing session. The answers which follow are comments only, and must not be taken as a commitment by DEC to supply any new feature, but in the past most of the suggestions from the user community which appear useful or which address a need not presently satisfied have ended up in the product. In some cases (though there were not too many of them this time), DEC will suggest an alternative solution to the same problem. DEC will also often point out that the solution to a problem depends upon some other software product, and it sometimes is necessary to go to that group and let them know there is a problem, though DEC sometimes passes the information on internally. Please note that anything labeled "Editor's note" is my opinion only, not DEC's.

1. Provide a method to reset RUNNING-TOTAL or RUNNING-COUNT (at bottom and breaks).

   Comment: Good suggestion.

   Editor's note: There were two different workarounds described in the Wombat Magic session.

2. Change the READY command to allow READY *."Domain to ready"

   Comment: Not possible. Will consider for future.

3. When using the linear regression line in the DATATRIEVE graphics (PLOT LR), would like to see the equation on the plot.

   Comment: Not in the realm of DTR graphics (another package may be more appropriate for this type of graphics).

4. Put the release notes into the installation guide.

   Comment: Done in V3.1 [of VAX-DATATRIEVE].

5. Would like to put "other" stuff into ReGIS memory so that our stuff can be put in with DTR plots.

   Comment: Will consider for future. Can work around by doing a PLOT to a file, edit, then plot to ReGIS device.

6. Allow logical name for node and access control such as:

```
$ASSIGN NODE "user password":: DTR_NODE
DTR>READY YACHTS AT DTR_NODE
```

Comment: Good suggestion, will consider.

Newsletter editor's note: This is the same as wish 21 on the 1984 Spring Wishlist.

7. Allow the "STARTING WITH" Boolean expression for DTR-11 and PRO/DTR.

Comment: Will pass [this wish] to the DTR-11 developers, but generally any new feature in DTR-11 will take away pool space.

8. Fix parenthetical arithmetical expression in Boolean statements. You cannot put parenthesis around only the left hand side of a statement. The statement

IF (1 + 2) equal 3 then . . .

does not give the expected result.

Comment: Will investigate correcting this, might have to document it as a restriction. Putting parenthesis around the entire expression may act as a workaround.

9. Be able to specify not to split a record on two pages of a report (printing a list sometimes runs over the bottom of the page).

Comment: Can't do in current architecture (it treats the inner list as a single field). Will consider a fix, a workaround could be to flatten the record with a CROSS or by writing to a temporary file.

Newsletter Editor's note: The Report Writer in VAX-DATATRIEVE Version 3.0 handles inner print lists differently than Version 2.X. Some of this problem has been fixed in Version 3.0 where it skips to the next page when the inner print reaches the bottom of the page.

10. Add Group Fields to ADT

Comment: Good suggestion, will consider.

Editor's note: Some re-evaluation of how ADT is being used and what functionality is required may be done.

Newsletter Editor's note: Group fields are supported in ADT in VAX-DATATRIEVE Version 3.0.

11. [Support for] FMS scrolling regions.

    Comment: Probably not in DTR, it has been looked at before and is very difficult. A workaround is possible by using a simple application program and callable DTR.

    Newsletter Editor's note: This is same as wish 7 on the 1984 Spring Wishlist.

12. On-line HELP for all supplied functions.

    Comment: Good suggestion.

13. Be able to divide by zero without getting "-1" as an answer.

    Comment: Good suggestion. (DTR-11 does this already.)

14. Support segmented strings and varying text.

    Comment: This is near the top on the priority list; it is being given serious attention.

15. Concatenate field with report name.

    Comment: Good suggestion, will consider. A sort of workaround is to explicitly print all of the text at the top of the report and use your own field for the title.

    Newsletter Editor's note: This is the same as wish 8 on the 1984 Spring Wishlist.

16. EXTRACT ALL (and EXTRACT ALL DOMAINS, EXTRACT ALL RECORDS, etc.)

    Comment: Good suggestion. DTR-11 has the QXTR utility for EXTRACT ALL

    Editor's note: Also see the magic in this issue.

    Newsletter Editor's note: This is the same as wish 3 on the 1984 Spring Wishlist.

17. A long suggestion which boils down to: add a $SPAWN word to DTR so that the user doesn't have to work out the way to spawn things on the VAX.

    Comment: Good suggestion, will look at.

18. Now that VMS V4 RMS supports quad-word keys, give that capability to DTR. (Especially for Dates.)

    Comment: High on the priority list, getting serious attention.

19. Document the way DTR-32 stores midnight in the date. Other languages calling DTR use VAX "NOW" for the date which includes the time, and they don't get a match on date.

    Comment: Will attempt to clarify the documentation, some of which is already there.

    Editor's note: Some comments were also made that the whole matter of dealing with date and time matches is receiving attention.

20. In DTR/FMS interface, do validation on a per field rather than a per form basis.

    Comment: Can't do this with the current architecture. Will look at it for something in the future.

From the Editor's Pen

Joe H. Gallagher, Cleveland Clinic Foundation, Cleveland, OH

While the winter in Cleveland has been your typically normal yucky winter, this has been a particularly bad winter for me. Being around the hospital all the time and exposed to all the good (and bad) bugs, I usually never get sick. But this year made up for it! I've been down three times with the flu since the first of the year and a chronic cough that I had had for years has finally been diagnosed as asthma. All my ills coupled with a new boss who thinks everything with IBM on it is great has made me consider giving up and heading southwest to a warmer, more hospitable climate. In addition, the Wombat Magic session at the Anaheim meeting did not get recorded. This meant a whole lot of extra work trying to get that material in shape for publication. I just kept putting off starting work on this issue of the Newsletter.

Well, this is my back-handed way of saying I'll try to do better about getting the Wombat Examiner out on a more timely basis. I have put my requisition in for some 36 hour days, but nobody seem to be able to supply me any.

Wombat Magic Session - 1984 Fall Symposium

Session Chairman: Dick Azzi
Session Editor: Joe H. Gallagher

Editor's note: Due to an oversight, the Wombat Magic Session at the 1984 Fall DECUS Symposium in Anaheim was not recorded. The following is a highly edited version of only a part of the material which was presented. I apologize to Anthony E. Scandora, Jr., Steven Cordiviola, Phil Dickerson, Ali Diba, Joe Meagher, and Bob Axelrod who presented very interesting magic. However, without an audio record of what transpired at the magic session, I found it impossible to reconstruct a meaningful representation of the presented material from only the transparencies. Material which appears in the text within square brackets [] is the Editor's interpretation of what was presented and may not necessarily represent what was presented by the speaker.

[At the beginning of the Magic Session, a special challenge was issued to the group. The problem was to create a set of report writer commands which would have the apparent effect of resetting the RUNNING COUNT at the beginning of each group so that each group would be numbered starting at one. Two Wombat Wizard offered solutions to the problem. However, after closer inspections and testing by the Newsletter Editor, only one was found to correctly solve the stated problem. Congratulations to Katherine Wrobel, Colorado Support Center, Digital Equipment Corporation, for providing the following elegant, simple solution to the problem. The example to illustrate the solution was provided by the Newsletter Editor.]

Katherine Wrobel, Digital Equipment Corporation, Colorado Springs, CO

Consider a domain EXAMPLE with a record which consists of employee's name and employee's department like:

```
01 EXAMPLE-REC.
   03 NAME PIC X(10).
   03 DEPARTMENT PIC X(10).
```

An effective reset of the RUNNING COUNT can be accomplished by:

```
REPORT A IN EXAMPLE SORTED BY DEPARTMENT
PRINT NAME, DEPARTMENT, ((RUNNING COUNT) - -
   (COUNT OF EXAMPLE WITH DEPARTMENT < A.DEPARTMENT), RUNNING COUNT
END-REPORT
```

This would give a report which looks like:

Joe	Accounting	1	1
Mary	Accounting	2	2
Laura	Accounting	3	3
Kathleen	Accounting	4	4
Shirley	Engineering	1	5
Mario	Engineering	2	6
Frank	Engineering	3	7
Henry	Maintenance	1	8
Philip	Maintenance	2	9

Chris Wool, E. I. duPont, Wilmington, DE

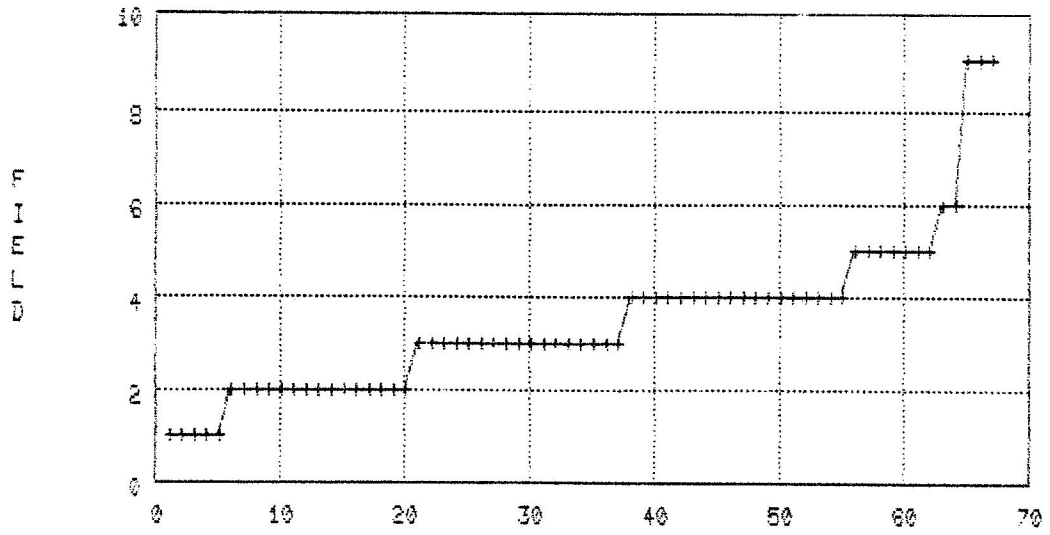
The problem is to print a number using only two significant digits. The solution: define a PIC clause for the number like NUM PIC 99V99 and then

```
PRINT CHOICE OF
    NUM LT  1 THEN FORMAT NUM USING .99
    NUM LT 10 THEN FORMAT NUM USING 9.9
    ELSE           FORMAT NUM USING 99|"."
    END_CHOICE
```

Philip Dickerson, Norther Telecom Inc., Concord, NH

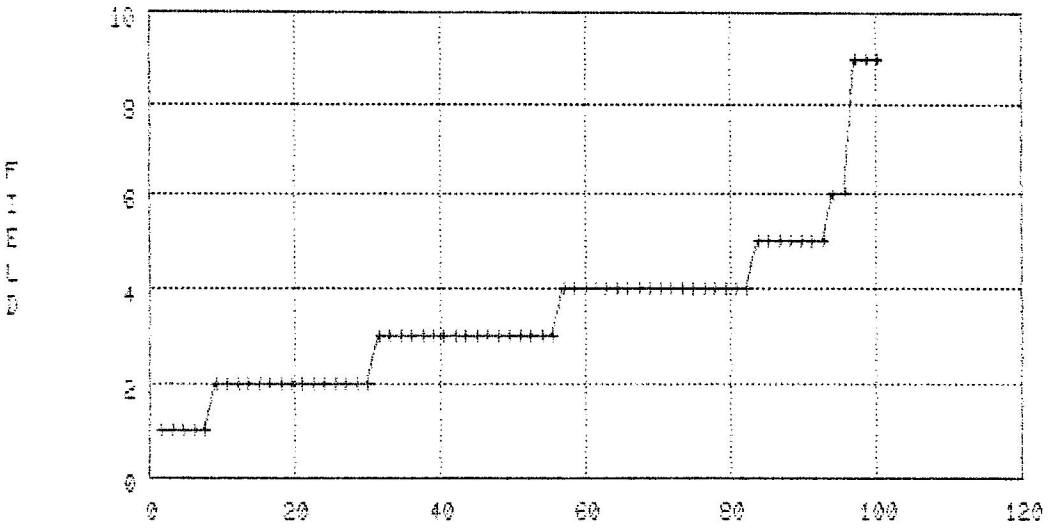
[We want to plot the running count of records against the value of a field to give a plot like]

```
PLOT X_Y ALL RUNNING COUNT, FIELD
```



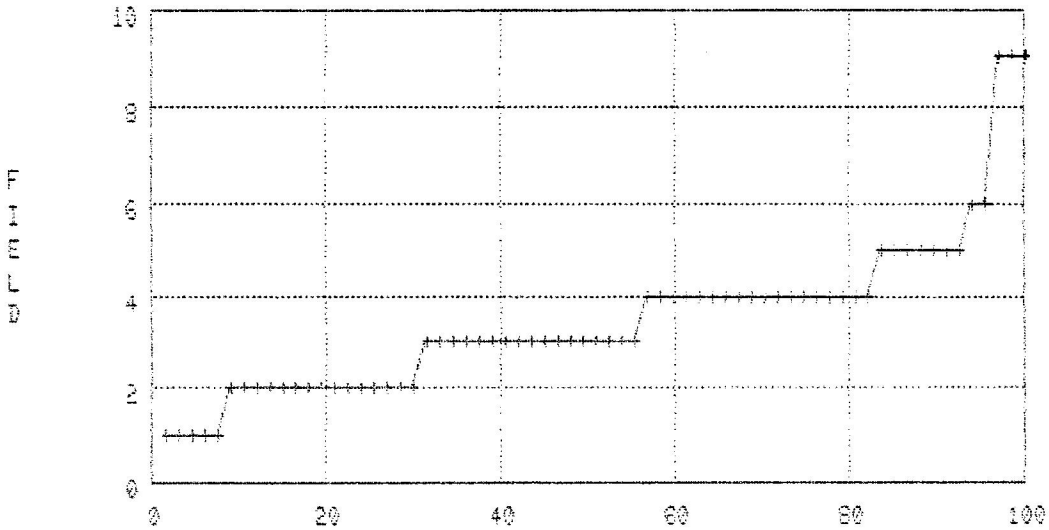
[However, this gives a graph which is in terms of the number of records. In this case 67. What we really want in a plot in terms of the percentage of running count. So we try]

```
PLOT X_Y ALL ((RUNNING COUNT * 100)/COUNT), FIELD
```



[But this is not quite right either, since the graph goes to 120 (percent) on the x-axis rather than stop at 100 (percent). The magic is to plot]

```
PLOT X_Y ALL ((RUNNING COUNT * 99.99)/COUNT), FIELD
```



Anthony E. Scandora, Jr., Wheaton, IL

One Friday afternoon, I got a phone call from a poor soul who had initialized the wrong disk in a cluster, was sorry, and wanted it back. The volume was mounted on another CPU, but VMS let it be initialized, anyway. The system's TU77 had been busted for a few weeks, so the only tape backup was too old to be useful. I look at Mike Higgins' and Phil Cannon's disk recovery tools from old RSX SIG tapes, but they assumed that the index file was still available. unlike RSX, which creates a reasonably sized index file by default, VMS just creates a little index file by default and extends it a lot. Fortunately, the initialize command had no arguments in it (except, of course, the wrong device name), so all we had to do was find all of the blocks that were in the old index file and use them to recover their files.

What does an index file block look like? I looked in STARLET.MLB for a macro describing it, but I couldn't find one (now I feel stupid for forgetting to look in LIB.MLB). The telephone support center had a huge manual on the subject, but they didn't want to read it to me over the phone. A long time ago, an ancient ODS-2 document for the PDP-11 fell into my hands. It was worth a shot. Sure enough, it still describes ODS-2. It even names the macro FHD02$, which is in RSXMAC.SML. I translated that macro to VAX macro, and we were in business.

I wrote SCAN2 to read every block on a disk, decide if it could have been in the index file, and if so, write some of the fields (see INDEXF.DTR) into a file called INDEXF.SEQ on a good disk. Then I converted it to INDEXF.IDX, and let DATATRIEVE analyze it.

Good old ODS-2 has back file pointers, which point to files' directory files. We had DATATRIEVE write a report sorted by back file pointer and file name, which produced a great sigh of relief.

The other half of the program, RECOVER2, reads every record in INDEXF.IDX, and for each record that is marked to be recovered, reads a file header by logical block numb (LBN) from the dead disk and restores the file to the directory logically named RECOVERED:.

That program is driven by RECOVER2.COM, which takes two arguments: a directory name and its FID on the dead disk. It calls DATATRIEVE to mark all files whose back file pointer matches the directory's FID to be recovered, creates the destination directory and assigns the logical "RECIVERED" to it, runs RECOVER2, and calls DATATRIEVE again to change the status of all of the files to be recovered to done.

Finally, we marked all unprocessed files to be recovered, and let RECOVER2 put them in SYSLOST.

They only had an ancient PDP-11 DTR, which appears to read commands from its terminal, so I had to "MCR DTR @FILE" to call it. The DATATRIEVE procedures could be a lot better for VAX DATATRIEVE. The old DTR uses the size of the PIC to decide if COMP FIELDS are WORD or LONGs.

There were no multi-header files on the disk, so RECOVER2 doesn't do them. it could be added. I started to have it create the file with XABs, but it got too complicated. I commented out that code and had it write the header. I think there is code in there for more than 64K files. The high byte of the relative volume is the high byte of a three byte file number. To do it right, the program should be changed to swap the bytes of the relative volume number and use that as the high work of a long file number.

This was done on a commercial system, so this SIG tape submission does not contain any of their directories. [The programs described have been submitted to the VAX? SIG tape. The INDEXF record definition is as follows:]

```
define record indexf_record
    01  indexf.
        05  lbn  usage is comp pic 9(6).
        05  fnum usage is comp pic 9(5).
        05  fseq usage is comp pic 9(4).
        05  fseg usage is comp pic 9(4).
        05  efnu usage is comp pic 9(5).
        05  efsq usage is comp pic 9(4).
        05  status pic x.
        05  direct pic x.
        05  bfnu usage is comp pic 9(5).
        05  bfsq usage is comp pic 9(4).
        05  fnam pic x(20).
        05  crdt usage is date.
        05  efbk usage is comp pic 9(5).
        05  hibk usage is comp pic 9(5).
;

$ ! recover2.com
$ set noon
$ if p1 .eqs. "" then inquire p1 "Directory to recover"
$ if p2 .eqs. "" then inquire p2 "Directory file number"
$ open/write dtrcom dtr.dtr
$ write dtrcom "ready indexf write"
$ write dtrcom "find indexf with bfnu = ", p2
$ write dtrcom "find current with status = "" """"
$ write dtrcom "modify all using status = ""!"""
$ write dtrcom "sort current by fnam"
$ write dtrcom "print all fnam, efbk, hibk"
$ write dtrcom "exit"
$ close dtrcom
$ mcr dtr @dtr.dtr
$ del dtr.dtr;0
$ cr/dir 'p1'
$ assign 'p1' recovered:
$ run recover2
$ open/write dtrcom dtr.dtr
$ write dtrcom "ready indexf write"
$ write dtrcom "find indexf with bfnu = ",p2
$ write dtrcom "find current with status = ""!"""
$ write dtrcom "modify all using status = ""*"""
$ write dtrcom "sort current by fnam"
$ write dtrcom "print all fnam, efbk, hibk"
$ write dtrcom "exit"
$ close dtrcom
$ mcr dtr @dtr.dtr
$ del dtr.dtr;0
```

Dick Azzi, Motorola SPS, Phoenix, AZ

[How do you determine the number of working days (week days) in a month excluding holidays? You do it as follows:]

```
declare report_month pic 99.
declare report_year pic 99.
declare first_of_month usage is date.
declare last_of_month usage is date.
declare days_to_check usage is date.
declare days pic 99.
! a table HOLIDAY-TABLE contains the dates of all holidays
report_month = *
report_year = *
first_of_month = report_month|"/01/"|report_year
last_of_month = first_of_month + 35
last_of_month = fn$month(last_of_month)|"/01/"|fn$year(last_of_month)
last_of_month = last_of_month + (-1)
days = 0
days_to_check = first_of_month
while days_to_check le last_of_month begin
    if "S" ne (format days_to_check using w) and
        days_to_check not in HOLIDAY-TABLE
        days = days + 1
    days_to_check = days_to_check + 1
    end
print days
```

Jim McMillan, Arizona Supreme Court, Tuscon, AZ

[How do you create reports in DATATRIEVE which have video or printing attributes? First, create a table with the desired attributes like

```
define table attributes
0:"<ESC>[0m"    ! all attributes off
B:"<ESC>[1m"    ! bold
U:"<ESC>[4m"    ! underscore
K:"<ESC>[5m"    ! blink
R:"<ESC>[7m"    ! reverse (inverse video)
end-table
```

then include within the record definition of the desired domain some PIC X fields between the other fields of the record to contain the video attributes or use a PRINT statement like

```
print   "0" via attributes | "Field One " | "0" via attributes | -
        "B" via attributes | "Field Two " | "0" via attributes | -
        "U" via attributes | "Field Three " | "0" via attributes | -
                           "Field Four"
```

which would give

Field One **Field Two** <u>Field Three</u> Field Four

If the attributes where stored within a record and there was a computed by variable which contained the "VIA ATTRIBUTES" it would be shorter and more convenient to manage.]

Pat Scopelliti, Corning Glass Works

This presentation is called "Domainless DATATRIEVE or Using DATATRIEVE as an FMS forms interface without using domains or hardly even knowing what you're doing!" Step 1 is to crate an FMS form named BADGE like the following:

```
+------------------------------------------------------------------+
|                    Domainless Datatrieve DEMO                    |
|                         Create a Badge                           |
|                                                                  |
|      Name:     XXXXXXXXXXXXXXX X XXXXXXXXXXXXXXXXXXX            |
|      Title:    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX                   |
|      Company:  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX             |
|      City:     XXXXXXXXXXXXXXXXXX    State: XX                   |
|                                                                  |
|                                                 Ready? X         |
+------------------------------------------------------------------+
```

Step 2 is to create the field declarations by using FMS with the following command.

```
$ FMS/DESCRIPTION/DECLARATIONS BADGE
```

This will create a file BADGE.TXT which contains the following:

```
01  BADGE.
    05  FNAME              PIC X(15).
    05  MI                 PIC X(1).
    05  LASTNAME           PIC X(20).
    05  TITLE              PIC X(30).
    05  COMPANY            PIC X(35).
    05  CITY               PIC X(19).
    05  STATE              PIC X(2).
    05  JUNK               PIC X(1).
```

This file will now be used three times with minor editing to perform steps four, five, and six.

Step 3. Create an empty procedure in DATATRIEVE by the following:

```
DTR> DEFINE PROCEDURE MAKE_BADGE
DFN> END-PROCEDURE
DTR> EDIT MAKE_BADGE
```

Now perform the next three steps using the editor [inside of DATATRIEVE] and making liberal use of the [editor] command 'INCLUDE BADGE.TXT' to retrieve the field names and definitions.

Step 4. Add field declarations inside of MAKE_BADGE of the form

```
DEFINE PROCEDURE MAKE_BADGE

DECLARE   FNAME              PIC X(15).
DECLARE   MI                 PIC X(1).
DECLARE   LASTNAME           PIC X(20).
DECLARE   TITLE              PIC X(30).
DECLARE   COMPANY            PIC X(35).
DECLARE   CITY               PIC X(19).
DECLARE   STATE              PIC X(2).
DECLARE   JUNK               PIC X(1).

END_PROCEDURE
```

Step 5. Add retrieval from the FORM. [Add to the procedure the following:]

```
REPEAT 100 BEGIN
    DISPLAY_FORM BADGE IN DEMO
    RETRIEVE USING BEGIN
         FNAME      = GET_FORM      FNAME
         MI         = GET_FORM      MI
         LASTNAME   = GET_FORM      LASTNAME
         TITLE      = GET_FORM      TITLE
         COMPANY    = GET_FORM      COMPANY
         CITY       = GET_FORM      CITY
         STATE      = GET_FORM      STATE
         JUNK       = GET_FORM      JUNK
    END
END
```

Step 6. Add an output section

```
on txa6: print skip, "<ESC>[1m","DECUS Symposium",skip,"<ESC>[0m",skip,
    fname(-),mi(-),lastname(-),skip,title(-),skip,company(-),skip,
    city(-),state(-), skip, skip
```

A typical badge would look like [Pat's example really contained double width characters which can't be reproduced in the newsletter]

```
      D E C U S  S y m p o s i u m

         Theodore       E Barr
         Systems Analyst
         Forestry Systems, Inc.
         Frostbite Fills      Mn
```

To make things even better, here is a BASIC program to do most of the work for you:

```
1       ! MAKEPROC
        ! SIMPLE program to turn an FMS forms declarations file (FMS/DESC/DECL)
        ! into a DTR procedure.  This program is meant as a demo and as such
        ! does not handle data entry errors in any sort of a graceful manner.
900     DIM FIELD.NAME$(200)
1000    ON ERROR GOTO 19000
110     PRINT
        LINPUT "Form name";FORM_NAME$
        PRINT
        LINPUT "Library name"; LIB_NAME$
1110    OPEN FORM_NAME$+".TXT" FOR INPUT AS FILE #1%, ACCESS READ
1120    OPEN FORM_NAME$+".PRO" FOR OUTPUT AS FILE #2%
1130    PRINT
        LINPUT "What will the procedure name be";P$
1140    print #2%, "DEFINE PROCEDURE ";P$
1200    LINPUT #1%, L$                          ! get a line
        GOTO 1200 IF MID(L$,5%,2%)<>"05"        ! skip unless its an '05 field PIC
1210    L$ = EDIT$(RIGHT(L$,9%),16%)            ! remove the 05, reduce blank to 1
        T% = POS(L$," ",1%)                     ! find first blank
        FIELD.COUNT% = FIELD.COUNT% + 1%        ! incr count of fields in stack
        FIELD.NAME$(FIELD.COUNT%) = LEFT(L$,T%-1%) ! put field in stack
        PRINT #2%, "DECLARE ";L$                ! print DECLARE line
        GOTO 1200
1300    PRINT #2%,"REPEAT 100 BEGIN"            ! produce display & retrieval code
        PRINT #2%,"    DISPLAY_FORM ";FORM_NAME$;" IN ";LIB_NAME$
        PRINT #2%,"    RETRIEVE USING BEGIN"
        PRINT #2%,"        ";FIELD.NAME$(I%);" = GET_FORM ";FIELD.NAME$(I%) &
                FOR I% = 1% TO FIELD.COUNT%
        PRINT #2%,"    END"
        ! produce the print code...simple form, let users edit as needed
        PRINT #2%,"    ON USER.FIL PRINT SKIP,"
        PRINT #2%,"        ";FIELD.NAME$(I%);"(-),SKIP," &
                FOR I% = 1% TO FIELD.COUNT%
        PRINT #2%,"        SKIP"
        PRINT #2%,"    END"
        PRINT #2%,"END-PROCEDURE"
        CLOSE #1%, #2%
        GOTO 32767

19000   RESUME 1300 IF ERR=11% AND ERL=1200% ! go on after end-of-file
19900   ON ERROR GOTO 0
32767   END
```

The procedure which is created by this BASIC program is like:

```
DEFINE PROCEDURE TEST
DECLARE FNAME PIC X(15).
DECLARE MI PIC X(1).
DECLARE LASTNAME PIC X(20).
DECLARE TITLE PIC X(30).
DECLARE COMPANY PIC X(35).
DECLARE CITY PIC X(19).
DECLARE STATE PIC X(2).
DECLARE JUNK PIC X(1).
```

```
REPEAT 100 BEGIN
    DISPLAY_FORM BADGE IN DEMO
    RETRIEVE USING BEGIN
        FNAME = GET_FORM FNAME
        MI = GET_FORM MI
        LASTNAME = GET_FORM LASTNAME
        TITLE = GET_FORM TITLE
        COMPANY = GET_FORM COMPANY
        CITY = GET_FORM CITY
        STATE = GET_FORM STATE
        JUNK = GET_FORM JUNK
    END
    ON USER.FIL PRINT SKIP,
        FNAME(-),SKIP,
        MI(-),SKIP,
        LASTNAME(-),SKIP,
        TITLE(-),SKIP,
        COMPANY(-),SKIP,
        CITY(-),SKIP,
        STATE(-), SKIP,
        JUNK(-),SKIP,
        SKIP
    END
END-PROCEDURE
```

William Porteous, Cabot Corporation, Andover, MA

[One of the most common needs of DATATRIEVE users is to be able to access DCL functionality from within DATATRIEVE. Several magic presentations in the past and one of the examples in the DATATRIEVE manuals show how to get to specific DCL commands.] This magic shows how to perform to any one line DCL command with a FORTRAN front end to DATATRIEVE through the DATATRIEVE call interface.

```
C
C       This program allows DTR to execute single word DCL
C       commands when they are preceded by a "$"
C       This is a portion of the code form DECUS program VAX-78 (FILTRA)
C
        INTEGER*2 DTR_OPTIONS, INIT_OPTIONS
        INTEGER*4 DTR$DTR, RET_STATUS, DTR$GET_STRING
        INCLUDE 'DTR$LIBRARY:DAB.FOR'
        INCLUDE 'DTR$LIBRARY:INFO.FOR'
C       DECLARE NORMAL AND EXIT STATUS
        EXTERNAL SS$_NORMAL, DTR$_EXIT
        CHARACTER*80 LINE
C
C       INITIALIZE THE INTERFACE
C
C       SET OPTIONS
        INIT_OPTIONS = DTR$K_SEMI_COLON_OPT
        CALL DTR$INIT (DAB, 20, MSG_BUFF, AUX_BUFF, INIT_OPTIONS)
```

```
C
C       CREATE USER DEFINED KEYWORDS SEND=1, RECEIVE=2, FILTRA=3,
C        FTR=4, AND $ FOLLOWED BY VMS COMMAND=5, HST=6, MENU=7
C
        CALL DTR$CREATE_UDK (DAB, '$', 5, DTR$K_UDK_COMMAND)
C
C       DECLARE THE OPTIONS FOR THE DTR$DTR CALL
C
        DTR_OPTIONS = DTR$M_OPT_CONTROL_C
     1              + DTR$M_OPT_STARTUP
     2              + DTR$M_OPT_UDK
     3              + DTR$M_OPT_BANNER
50      RET_STATUS = DTR$DTR (DAB, DTR_OPTIONS)
        IF (RET_STATUS .EQ. %LOC(DTR$_EXIT)) GO TO 5500
        IF (DAB$W_UDK_INDEX .EQ. 5) GO TO 500
500     CONTINUE
        CALL DTR$GET_STRING (DAB, DTR$K_TOK_COMMAND, LINE)
        CALL LIB$SPAWN (LINE)
        CALL DTR$END_UDK (DAB)
        GO TO 50
C
C       CLEAN UP, AND END THE INTERFACE
C
5500    CONTINUE
        CALL DTR$FINISH (DAB)
        STOP
        END
```

[The instructions for linking a FORTRAN program, such as the one above, with the sharable DATATRIEVE image are contained in Chapter 3 "Sample FORTRAN Programs" of the **VAX DATATRIEVE Guide to Programming and Customizing**, AA-P863B-TE.]

Basil Harris, Digital Equipment Corporation, Nashua, NH

Problem: How do I get a plot output file to a system ReGIS device?

Solution:

```
DTR> PLOT WOMBAT ON WOMBAT.DAT   ! or your favorite plot
DTR> EXIT
$ ! Use DECslide to
$ SLIDE/NOINTERACT/PIX=WOMBAT.DAT
$ ! This create a file WOMBAT.SLS that can be sent to a ReGIS printer
```

Bert Rosberry, U. S. Coast Guard, New Orleans, LA

This is an oldie but goodie for VAX systems. [to remind users how to temporarily stop a process, answer the PHONE (or perform some other DCL function), and then return to the previous process]

```
DTR> READY YACHTS
DTR> FIND YACHTS
Jasmann is phoning you      (HH:MM:SS)
DTR> ^Y
$ SPAWN
%DCL-S-SPAWNED, process ROSEBERRY_1 spawn
%DCL-S-ATTACHED, terminal now attached to process ROSEBERRY_1
$ PHONE ANSWER
! carry on a phone conversation with Jasmann
$ LOGOUT
  Process ROSEBERRY_1, logged out at DD-MMM-YYYY HH:MM:SS.SS
%DCL-S-RETURNED, control returned to process ROSEBERRY
$ CONTINUE
DTR> ! domains, collections, etc are all in the same state as before
```

[Even if one does not use the SPAWN, there are certain DCL commands such as SET which do not change the environment so that it is still possible to use a CONTINUE. For example]

```
$ SHOW DEFAULT
 DRA0:[SYS0.SYSEXE]
$DTR
DTR> READY YACHTS
 File not found ---- ! [YACHTS.DAT is in [SYS0.DTR] not [SYS0.SYSEXE]
DTR> ^Y
$ SET DEFAULT DRA0:[SYS0.DTR]  ! [SET does not zap DTR]
$ CONTINUE                     ! [re-start DTR
DTR> READY YACHTS
```

Dick Azzi, Motorola SPS, Phoenix, AZ

[We want to send some notices to our users on a certain date and then keep track of whether we have sent the notice. But we didn't want to change the record layout. Thus, we needed to flag the records, but without adding another field to the record. This can be done by burying a flag in the least significant bits of a date, as follows:]

```
! Use a record definition like
03 OA_DATE USAGE IS DATE.
03 REDA REDEFINES OA_DATE.
   05 FLAG USAGE IS QUAD.  ! DATE and QUAD both take 8 bytes
```

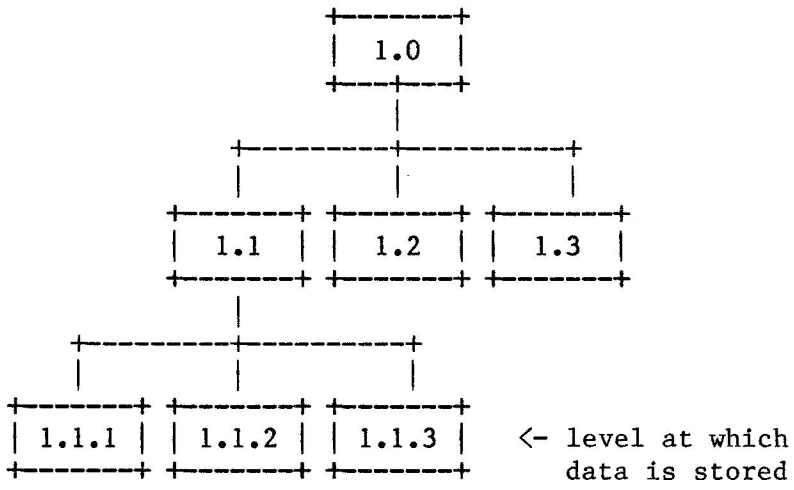
[Then the statements to select and process records]

```
FOR DOMAIN WITH (RSE on the OA_DATE and records) BEGIN
    IF FN$HUNDREDTH(OA_DATE) EQUAL 0 THEN BEGIN
        :SEND_NOTICES_PROCEDURE
        FLAG = FLAG + 1000000    ! -> 0.01 SECONDS
        END
    END
```

[This will set the date (and time) slightly past midnight (0.01 seconds) for each of the dates which will indicate that a notices has been sent.]

Diane Harris, Digital Equipment Corp, Colorado Springs, CO

Work breakdown structures often consist of a task broken into subtask and further into work packages. We want to be able to do totaling "on top of" a field. That is, if the work is broken down as



To create a report with this "upward" summarization we need a domain like

```
define domain foo using foo-record on foo.dat;
define record foo-record using
01 foo-rec.
03 work_package pic x(5).
   03 wp_bkd redefines work_package.
      05 subtask pic x(3).
      05 filler pic x(2).
   03 t_bkd redefines work_package.
      05 task pic x(2).
      05 filler pic x(3).
   03 amount pic 9(3).
;
```

Then, create a report on the records like:

```
REPORT FOO SORTED BY WORK_PACKAGE ON FOO.TMP
SET NO NUMBER
SET NO DATE
PRINT COL 1, WORK_PACKAGE(-), COL 7,  AMOUNT(-)
AT BOTTOM OF SUBTASK PRINT COL 1, WP_BKD(-), COL 7, TOTAL AMOUNT(-)
AT BOTTOM OF TASK PRINT COL 1, T_BKD(-), COL 7, TOTAL AMOUNT(-)
END_REPORT
```

which would give something like:

```
1.1.1 050
1.1.2 010
1.1.3 040
1.1   100
1.2.1 065
1.2.2 035
1.2   100
1.    200
```

Use an editor to remove the form feeds and the extra blank lines from FOO.TMP. Then create a second domain to work on this report as follows:

```
define domain tmpfoo using tmpfoo-record on foo.tmp;
define record tmpfoo-record using
01 tmpfoo-rec.
   03 work_package pic x(5).
   03 filler pic x(1).
   03 amount pic 999
      edit-string is zz9.
;
```

Then report on the report with:

```
report empfoo sorted by work_package on foo.rpt
print work_package, amount
end-report
```

which gives the information is the desired hierarchical structure.

```
1.    200
1.1   100
1.1.1 050
1.1.2 010
1.1.3 040
1.2   100
1.2.1 065
1.2.2 035
```

Bob Pearson

[Bob's magic involves a special DCL functions ASKIT which is available on one of the VAX SIG tapes. With this function, it is possible to do some very cleaver commands like:]

```
$ !* LIST.JCL
$ !* Simple list of the file
%%CLASS = ASKIT("Class code", "STR/UP", "A,B,F", "F", "")
%%DATE = ASKIT("Cutoff date", "DAT", "", "TODAY", "")
$ dtr
  ready CDD$TOP.SAMPLE_DOMAIN shared
  find SAMPLE_DOMAIN with class eq "%%CLASS" and
      date le "%%DATE"
  report current on list.lis;%%VERSION
  set report_name = "Simple listing"/"For class %%CLASS"
  print emp_no, name, cutoff_date
  end_report
  exit
$ %%PRINT list.lis;%%VERSION
$ exit
```

Which when executed would produce

```
Class code? X
please choose from A,B,F

Class code? B

Cutoff date? 120284

          Def       SYS$PRINT
          BL2       Building two
          AC        Accounting office
          NO        No print
Where shall we print this[Def]? AC

Job submitted
```

Would produce the job

```
$ !* LIST.JCL
$ !* Simple list of the file
$ dtr
  ready CDD$TOP.SAMPLE_DOMAIN shared
  find SAMPLE_DOMAIN with class eq "B" and
      date le "12/02/84"
  report current on list.lis;12345
  set report_name = "Sample listing"/"For class B"
  print emp_no, name, cutoff_date
  end_report
  exit
$ print/delete/queue=ttc0 list.lis;12345
$ exit
```

Walt Johnson,

This is CDD magic. BACKUP skips files which are open for write. Our CDD is always open for write. Therefore, our CDD on the backup tape was rather stale!! [Laughter] Our assistant system manager couldn't identify a CDD dictionary, so he deleted it. The user them called up with a complaint about her last dictionary. So... We call Colorado Springs. They suggested restoring out stale CDD and pointing to it long enough to extract her definitions. Since the other users wouldn't be thrilled, we used another solution: We defined the restored old dictionary as a subdictionary file, and used

```
DMU> COPY CDD$TOP.USER.CDD$TOP.USER...  CDD$TOP.USER...
```

which works just fine.

Andy Schneider, Digital Equipment Corporation

In DATATRIEVE-11, what does this do?

```
DTR> ready yachts
DTR> report yachts
DTR> at top of page print new-page
DTR> end-report
```

Answer: It empties a box of paper in 30 seconds or less!

Doug Cropper, Digital Equipment Corp
Presented by Andy Schneider

[VAX DATATRIEVE supports the EXTRACT ALL and the ability to extract the definition of a specific dictionary object. The last two wishlists have included requests to be able to perform commands like

```
DTR> EXTRACT ALL DOMAINS on dom.def  ! or
DTR> EXTRACT ALL RECORDS on rec.def  ! etc
```

which would work for domains, records, tables, procedures, plots, and databases -- essentially all dictionary objects.]

[Here is a way to do an EXTRACT on a class of dictionary objects.] First thing to do is to put all of these into a CDD dictionary and create the logical MAGIC$DIC to be the dictionary that contains them. Then to do an extract just issue the command

```
:MAGIC$DIC.EXTRACT_whatever
```

The procedure will prompt for a file name. It stores the file name in a variable FILE_NAME. Then it opens an output log file called DIC.DAT, does a SHOW whatever [whatever class of dictionary objects you are trying to extract] and then closes the log file. Then it readies the MAGIC$DIC.TEMP domain as OLD, defines a new file for TEMP with a key, then stores the old "whatever"

names in the new data file with the name as a key. This will allow me to erase the "DTR> SHOW whatever" and "DTR> CLOSE" lines and any blank lines. Then we finish all domains and open the new keyed data file. This file now has all the names of the particular [class of] dictionary objects requested. Then I loop through all the records ("Reduced to DIC_NAME" if only the highest version is requested), building one massive "EXTRACT ON FILE_NAME path_name, path_name, ... " command and put that in a file called EXTR.COM. I then execute the command file and everything is extracted on the one file. It takes longer to explain it than to execute. The domain, record, and procedures you need are as follows:

```
!
! the domain used
!
REDEFINE DOMAIN TEMP USING TEMP_REC ON DIC.DAT;
!
! the record definition for temp_rec
!
REDEFINE RECORD TEMP_REC USING
01 TOP.
   03 DIC_NAME PIC X(45).
;
!
! the main extract procedure for extracting only highest version
!
REDEFINE PROCEDURE EXTR
DECLARE COMMA PIC 99.
DECLARE NUMBER PC 9999.
DECLARE FIRST_TIME PIC 9.
FIRST_TIME = 1
NUMBER = COUNT OF TEMP REDUCED TO DIC_NAME
ON EXTR.DTR BEGIN
    FOR TEMP REDUCED TO DIC_NAME BEGIN
       IF FIRST_TIME EQ 1 THEN BEGIN
           PRINT "extract on "|FILE_NAME
           FIRST_TIME = 0
           END
       IF RUNNING COUNT NOT EQ NUMBER THEN
           PRINT DIC_NAME||", " ELSE
           PRINT DIC_NAME(-)
       END
    END
END_PROCEDURE
!
! the main extract procedure for extracting all versions
!
REDEFINE PROCEDURE EXTR_ALL
DECLARE NUMBER PIC 9999.
DECLARE END_STR PIC X(40).
END_STR = ", -"
NUMBER = COUNT OF TEMP
ON EXTR.DTR BEGIN
    PRINT "extract -"
    FOR TEMP BEGIN
       IF RUNNING COUNT EQ NUMBER THEN END_STR = " on "|FILE_NAME
```

```
        PRINT DIC_NAME||END_STR
        END
    END
END_PROCEDURE
!
! use this for extraction all version of all databases
!
REDEFINE PROCEDURE EXTRACT_ALL_ALL_DATABASES
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list of dictionary items"
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW DATABASES
CLOSE
SET COLUMNS_PAGE = 85
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME
READY MAGIC$DIC.TEMP.AS NEW WRITE
NEW = OLD
FINISH
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW DATABASES" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
BEGIN
    :MAGIC$DIC.EXTR_ALL
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
FINISH TEMP
END_PROCEDURE
!
! use this for extraction all version of all domains
!
REDEFINE PROCEDURE EXTRACT_ALL_ALL_DOMAINS
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list of dictionary items"
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW DOMAINS
CLOSE
SET COLUMNS_PAGE = 85
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME
```

```
READY MAGIC$DIC.TEMP.AS NEW WRITE
NEW = OLD
FINISH
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW DOMAINS" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
BEGIN
    :MAGIC$DIC.EXTR_ALL
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
FINISH TEMP
END_PROCEDURE
!
! use this for extraction all version of all procedures
!
REDEFINE PROCEDURE EXTRACT_ALL_ALL_PROCEDURES
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list of dictionary items"
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW PROCEDURES
CLOSE
SET COLUMNS_PAGE = 85
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME
READY MAGIC$DIC.TEMP.AS NEW WRITE
NEW = OLD
FINISH
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW PROCEDURES" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
BEGIN
    :MAGIC$DIC.EXTR_ALL
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
```

```
FINISH TEMP
END_PROCEDURE
!
! use this for extraction all version of all records
!
REDEFINE PROCEDURE EXTRACT_ALL_ALL_RECORDS
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list of dictionary items"
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW RECORDS
CLOSE
SET COLUMNS_PAGE = 85
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME
READY MAGIC$DIC.TEMP.AS NEW WRITE
NEW = OLD
FINISH
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW RECORDS" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
BEGIN
    :MAGIC$DIC.EXTR_ALL
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
FINISH TEMP
END_PROCEDURE
!
! use this for extraction all version of all tables
!
REDEFINE PROCEDURE EXTRACT_ALL_ALL_TABLES
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list of dictionary items"
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW TABLES
CLOSE
SET COLUMNS_PAGE = 85
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME
READY MAGIC$DIC.TEMP.AS NEW WRITE
NEW = OLD
```

```
FINISH
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW TABLES" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
BEGIN
    :MAGIC$DIC.EXTR_ALL
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
FINISH TEMP
END_PROCEDURE
!
! use this for extracting the highest version of all databases
!
REDEFINE PROCEDURE EXTRACT_DATABASES
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list databases
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW DATABASES
CLOSE
SET COLUMNS_PAGE = 85
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME (DUP)
READY MAGIC$DIC.TEMP AS NEW WRITE
BEGIN
    DECLARE COMMA PIC 99.
    FOR OLD STORE NEW USING BEGIN
       COMMA = FN$STR_LOC(DIC_NAME,";")
       DIC_NAME = FORMAT (FN$STR_EXTRACT(DIC_NAME,1,(COMMA - 1))) USING X(45)
       END
    END
FINISH
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW DATABASES" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
```

```
BEGIN
    :MAGIC$DIC.EXTR
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
FINISH TEMP
RELEASE FILE_NAME
END_PROCEDURE
!
! use this for extracting the highest version of all domains
!
REDEFINE PROCEDURE EXTRACT_DOMAINS
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list of domains
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW DOMAINS
CLOSE
SET COLUMNS_PAGE = 85
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME (DUP)
READY MAGIC$DIC.TEMP AS NEW WRITE
BEGIN
    DECLARE COMMA PIC 99.
    FOR OLD STORE NEW USING BEGIN
       COMMA = FN$STR_LOC(DIC_NAME,";")
       DIC_NAME = FORMAT (FN$STR_EXTRACT(DIC_NAME,1,(COMMA - 1))) USING X(45)
       END
    END
FINISH
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW DOMAINS" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
BEGIN
    :MAGIC$DIC.EXTR
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
FINISH TEMP
RELEASE FILE_NAME
END_PROCEDURE
```

```
!
! use this for extracting the highest version of all procedures
!
REDEFINE PROCEDURE EXTRACT_PROCEDURES
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list of procedures
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW PROCEDURES
CLOSE
SET COLUMNS_PAGE = 85
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME (DUP)
READY MAGIC$DIC.TEMP AS NEW WRITE
BEGIN
    DECLARE COMMA PIC 99.
    FOR OLD STORE NEW USING BEGIN
       COMMA = FN$STR_LOC(DIC_NAME,";")
       DIC_NAME = FORMAT (FN$STR_EXTRACT(DIC_NAME,1,(COMMA - 1))) USING X(45)
       END
    END
FINISH
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW PROCEDURES" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
BEGIN
    :MAGIC$DIC.EXTR
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
FINISH TEMP
RELEASE FILE_NAME
END_PROCEDURE
!
! use this for extracting the highest version of all records
!
REDEFINE PROCEDURE EXTRACT_RECORDS
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list of records
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW RECORDS
CLOSE
SET COLUMNS_PAGE = 85
```

```
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME (DUP)
READY MAGIC$DIC.TEMP AS NEW WRITE
BEGIN
    DECLARE COMMA PIC 99.
    FOR OLD STORE NEW USING BEGIN
       COMMA = FN$STR_LOC(DIC_NAME,";")
       DIC_NAME = FORMAT (FN$STR_EXTRACT(DIC_NAME,1,(COMMA - 1))) USING X(45)
       END
    END
FINISH
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW RECORDS" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
BEGIN
    :MAGIC$DIC.EXTR
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
FINISH TEMP
RELEASE FILE_NAME
END_PROCEDURE
!
! use this for extracting the highest version of all tables
!
REDEFINE PROCEDURE EXTRACT_TABLES
DECLARE FILE_NAME PIC X(32).
FILE_NAME = "file name"
PRINT "ignore following list of tables
SET COLUMNS_PAGE = 1
OPEN DIC.DAT
SHOW TABLES
CLOSE
SET COLUMNS_PAGE = 85
PRINT "thank you"
READY MAGIC$DIC.TEMP AS OLD
DEFINE FILE FOR MAGIC$DIC.TEMP KEY = DIC_NAME (DUP)
READY MAGIC$DIC.TEMP AS NEW WRITE
BEGIN
    DECLARE COMMA PIC 99.
    FOR OLD STORE NEW USING BEGIN
       COMMA = FN$STR_LOC(DIC_NAME,";")
       DIC_NAME = FORMAT (FN$STR_EXTRACT(DIC_NAME,1,(COMMA - 1))) USING X(45)
       END
    END
FINISH
```

```
READY MAGIC$DIC.TEMP WRITE
FIND TEMP
!erase the first 2 lines
SELECT FIRST
ERASE
!erase the "DTR> SHOW TABLES" LINE
SELECT NEXT
ERASE
SELECT LAST
ERASE
RELEASE CURRENT
BEGIN
    :MAGIC$DIC.EXTR
    END
SET COLUMNS-PAGE = 80
@EXTR.DTR
FINISH TEMP
RELEASE FILE_NAME
END_PROCEDURE
```

Diana Washburn, Hanes Hosiery, Winston-Salem, NC

This is an example of using subscripting or indexing in DATATRIEVE. This is very handy when you need to get information for specific fields within a list, but hove no value within the list to use in the RSE with the 'ANY' Boolean expression. For instance, 12 months of data but no actual month identifier (1 - 12 or JAN - DEC). It should only be used when the normal DATATRIEVE list processing features can not be used. They said it couldn't be done, but watch:

First I'll show you the domain and record that will be used in the example.

```
DEFINE DOMAIN MAGIC USING MAGIC_REC ON DEVICE:MAGIC.DAT;
DEFINE RECORD MAGIC_REC USING
01 MAGIC_REC.
   02 MAGIC_KEY                PIC X(5).
   02 MAGIC_LIST               OCCURS 4 TIMES.
      03 MAGIC_LIST_FIELD      PIC 9(4) QUERY_NAME QTY.
;
```

The data that has been stored in the file looks like this:

```
DTR> PRINT MAGIC

      MAGIC
MAGIC LIST
 KEY  FIELD

11111 0001
      0002
      0003
      0004
22222 0010
      0020
      0030
      0040
33333 0100
      0200
      0300
      0400
44444 1000
      2000
      3000
      4000
55555 1111
      2222
      3333
      4444
```

Here is a very simple example that will allow you to enter one subscript.

```
DEFINE PROCEDURE MAGIC_SUBSCRIPTING
DECLARE COUNTER PIC 99.   ! The counter is used to identify the level
DECLARE SUBSCRIPT PIC 99.
READY MAGIC SHARED
SUBSCRIPT = *.'Subscript' ! The subscript is prompted here but could
                          ! be taken from another source (screen,record)
FOR MAGIC BEGIN
   COUNTER = 1
   FOR MAGIC_LIST BEGIN
      IF COUNTER = SUBSCRIPT THEN PRINT MAGIC_LIST_FIELD
      COUNTER = COUNTER + 1
      END
   END
END_PROCEDURE
```

Here is what happens when you run the procedure:

```
DTR> :MAGIC-SUBSCRIPTING
Enter Subscript : 1

MAGIC
LIST
FIELD

0001
0010
0100
1000

DTR> :MAGIC-SUBSCRIPTING
Enter Subscript : 3

MAGIC
LIST
FIELD

0003
0030
0300
3000
```

```
DTR> :MAGIC-SUBSCRIPTING
Enter Subscript : 2

MAGIC
LIST
FIELD

0002
0020
0200
2000

DTR> :MAGIC-SUBSCRIPTING
Enter Subscript : 4

MAGIC
LIST
FIELD

0004
0040
0400
4000
```

Now let's add a range of subscripts. In this case the values from the specified list levels are accumulated and the total is printed for each record:

```
DEFINE PROCEDURE MAGIC_SUBSCRIPTING
DECLARE COUNTER PIC 99.
DECLARE BEG_SUBSCRIPT PIC 99.
DECLARE END_SUBSCRIPT PIC 99.
DECLARE MAGIC_TOTAL   PIC S9(9).
READY MAGIC SHARED
BEG_SUBSCRIPT = *.'Beginning Subscript or Index'
END_SUBSCRIPT = *.'Ending Subscript of Index'
FOR MAGIC BEGIN
   MAGIC_TOTAL = 0
   FOR MAGIC_LIST BEGIN
      IF COUNTER BT BEG_SUBSCRIPT AND END_SUBSCRIPT
         MAGIC_TOTAL = MAGIC_TOTAL + MAGIC_LIST_FIELD
      COUNTER = COUNTER + 1
      END
      PRINT MAGIC_TOTAL
   END
END_PROCEDURE
```

Running this procedure gives you:

```
DTR> :MAGIC-SUBSCRIPTING
Enter Beginning Subscript : 1
Enter Ending Subscript : 2

   MAGIC
   TOTAL

 000000003
 000000030
 000000300
 000003000
```

```
DTR> :MAGIC-SUBSCRIPTING
Enter Beginning Subscript : 2
Enter Ending Subscript : 4

   MAGIC
   TOTAL

 000000009
 000000090
 000000900
 000009000
```

If you have data in a list that is not identified in the record but is in a fixed pattern, you can use a table lookup to get at the data without having to remember the order of the data in the list. Here is the table used in the next example:

```
DEFINE TABLE MAGIC_CODE_TABLE USING
'AA':1
'BB':2
'CC':3
'DD':4
END_TABLE

DEFINE PROCEDURE MAGIC_SUBSCRIPTING
DECLARE COUNTER PIC 99.
DECLARE SUBSCRIPT_CODE PIC XX.
READY MAGIC SHARED
SUBSCRIPT_CODE = *.'Translation code for subscript'
FOR MAGIC BEGIN
   COUNTER = 1
   PRINT COL 2, MAGIC_KEY(-)
   FOR MAGIC_LIST BEGIN
      IF COUNTER = SUBSCRIPT_CODE VIA MAGIC_CODE_TABLE
         PRINT COL 10, MAGIC_LIST_FIELD(-)
      COUNTER = COUNTER + 1
      END
   END
END_PROCEDURE
```

When you run the procedure you get:

```
DTR> :MAGIC_SUBSCRIPTING
Enter Translation code for subscript : CC

  11111
          0003
  22222
          0030
  33333
          0300
  44444
          3000
  55555
          3333
```

There are times when you may need to get the data in several of the levels of a list that are not contiguous. In this case you can use a DATATRIEVE function to pull the subscripts from a string:

```
DEFINE PROCEDURE MAGIC_SUBSCRIPTING
DECLARE COUNTER PIC 99.
DECLARE SUBSCRIPTS PIC X(5).
READY MAGIC SHARED
SUBSCRIPTS = *.'Subscript(s) separated by commas'
FOR MAGIC BEGIN
   COUNTER = 1
   PRINT COL 2, MAGIC_KEY(-)
   FOR MAGIC_LIST BEGIN
      IF COUNTER = FN$STR_EXTRACT(SUBSCRIPTS,1,1),
                   FN$STR_EXTRACT(SUBSCRIPTS,3,1),
                   FN$STR_EXTRACT(SUBSCRIPTS,5,1)
         PRINT COL 10, MAGIC_LIST_FIELD(-)
      COUNTER = COUNTER + 1
      END
   END
END_PROCEDURE
```

Here is what happens when this procedure is run:

```
DTR> :MAGIC-SUBSCRIPTING
Enter Subscript(s) separated by commas: 1,4

  11111
          0001
          0004
  22222
          0010
          0040
  33333
          0100
          0400
  44444
          1000
          4000
  55555
          1111
          4444
```

In the last example you would have to set up the subscripts variable to be able to handle the maximum number of subscripts allowed. Remember to add 1 character per subscript for the separator (comma). In the example I allowed for 3 subscripts since the occurs is 4 and I would assume that the normal DTR list processing would be used if all entries in the list were desired. This gets a little more complicated when the subscripts itself can be a variable number of digits. The easiest way to handle this is to establish a fixed number of places for the subscript and require that leading zeros be entered.

The concept of subscripting can be very helpful. I use this technique to build summary files and then report the data in the summarized file. I have not found a way yet to use this within the report command.

To give you an idea of how it applies to a real life application, let me tell you how I use it. I have a record with 27 month worth of data. The record does not indicate the year/month represented in each level. This information is in another domain in a control record. My users request data from one date to another (i.e., December '82 to December '83 or September '82 to January '84). I have to locate the proper levels by using the control file. I convert the year/month to a number (1 to 27) for beginning and ending dates. Then based on a long series of if statements I accumulate the data for the appropriate months.

[The next three magic presentations all have to do with using variable length strings. Each is slightly different, but each gets at the same (or similar) problem.]

Chris Wool, E. I. duPont, Wilmington, DE

The problem is: We need to input a variable length character string and use it in a STARTING WITH Boolean expression. When one uses

```
FIND domain with field STARTING WITH *."prompt_message"
```

it works perfectly well. However, when one need to save the value that has been input to use several times, such as,

```
DECLARE TEXT PIC X(10).
TEXT = *."prompt_message"
```

then

```
FIND domain with field STARTING WITH TEXT
```

no longer works, since the global variable contains blanks. However, it one uses

```
FIND domain with field STARTING WITH TEXT||"" ! the null string
```

everything works perfectly well again since the blank fills are effectively removed.

Stephen Pacheco, Ship Analytics, North Stonington, CT

To create a variable length character string to use in string comparisons, you may use

```
DECLARE TEXT PIC X(40).
TEXT = *."prompt_message"||"**"
DECLARE MATCH COMPUTED BY
  (FN$STR_EXTRACT(TEXT ,1 ,FN$STR_LOC(TEXT, "**") - 1) .
```

Then MATCH may be used in string comparisons.

Philip Dickerson, Northern Telecom Incorporated, Concord NH

In DATATRIEVE-11 where there is no STARTING WITH, a variable length, starting with comparison can be accomplished as follows:

```
DECLARE A PIC X(10).
DECLARE B PIC X(10) COMPUTED BY A||"zzzz" .
A = *."string"
FIND domain with field BETWEEN A AND B
```

Contributions

Contributions for the newsletter can be sent to either of the following addresses:

Editor, DATATRIEVE Newsletter
c/o DECUS U. S. Chapter
249 Northboro Road, BP02
Marlboro, MA 01752

Joe H. Gallagher, Ph. D.
DATATRIEVE Newsletter Editor
Director, Medical and
Research Computing
Cleveland Clinic Foundation
9500 Euclid Avenue
Cleveland, Ohio 44106

Letters and articles for publication are requested from members of the SIG. They may include helpful hints, inquiries to other users, reports on SIG business, summaries of SPRs submitted to Digital or other information for members of the DATATRIEVE SIG. Machine readable input is highly desirable and machine-to-machine transfer of material is preferred, but most anything legible will be considered. However, this newsletter is not a forum for job and/or head hunting, nor is commercialism appropriate.

Table of Contents

Chairman's Corner

Larry Jasmann, U. S. Coast Guard, Burke, VA

The last several months have been very active, both in and out of DATATRIEVE SIG. One of the most pressing matters which has occupied my time, and that of the Steering Committee is the need to change the name and a partial redirection of the focus of the SIG. The following concept paper was submitted and approved by the SIG council in the latest Florida meeting, and it represents a concise statement of what we are doing and why. Please read it carefully, and I welcome your comments to it.

Above all, please be reassured that the Wombat is not going away, he is just expanding his scope of activity a bit.

DATATRIEVE and Fourth Generation Language SIG Proposal

Background:

In order to understand why DATATRIEVE SIG wishes to change it's name and the scope of its activities, it is necessary to understand DEC's predicament with DATATRIEVE. DATATRIEVE at this point is second only to FORTRAN in sales for layered software products. Clearly, it will be around for quite a while. It is useful on many levels, from a novice computer user up to system programmers. It is an extremely powerful and flexible data management tool, making possible application production savings of up to 80%. In spite of this, DEC is actively working on products which would, at least in part, replace DATATRIEVE. Why is this? There are several reasons:

1. DATATRIEVE is a fairly mature product. It has gone through many revisions on the PDP side, and two major rewrites on the VAX side. In terms of functionality, it has expanded about as far as it can go in its present form. The volume of code is large, and it is getting increasingly difficult to add new features without "breaking" old features. In a business that sees a major breakthrough every 6 months, a product that has reached the end of its functionality growth will sooner or later be overtaken by the competition.

2. While DATATRIEVE is indeed "user friendly", it isn't as helpful as it could be to non-computer users. Menu driven packages are easier to use for those who are not computer literate. DATATRIEVE functionality has expanded dramatically on the "top" end, but on the "low" end, other than a better help facility, there hasn't been much growth. Given the structure of the language, this is not likely to change.

3. The cost of getting the bells and whistles incorporated into DATATRIEVE is high. Not only is the software fairly expensive (cheaper than it used to be!), but it is CPU and memory intensive, particularly if used improperly. Unless one really understands DATATRIEVE, it is inevitable

that it be used in an inefficient way.

Another issue is the fact that significant third party software products now exist which do the same sorts of things that DATATRIEVE does. Users of these products belong to DECUS and their needs, to the extent permitted by commercialism guidelines, need to be addressed.

Concerns such as those expressed above have focused the attention of the DATATRIEVE SIG Steering Committee on the needs of users of this type of product, rather than DATATRIEVE exclusively. When we searched our corporate soul, we found that beneath the surface, we represent the needs of DEC computer users who have a need to access data bases for informational purposes. Particularly, we represent the needs of those who use fourth generation products such as DATATRIEVE to do this. By changing the scope to focus on a type of user, rather than a particular product, we find that many of the ambiguities which have plagued us in the past disappear.

So what is a fourth generation language anyway? Isn't this just another ill-defined buzz word that leads down a path rosey at the beginning and full of rocks at the end? I'm not an expert, but to me the term applies to languages that you tell WHAT to do (and get results) rather than HOW to do it. That is clearly an over simplification, but perhaps it will work in a limited frame of reference. Fourth Generation Languages do many things, the ones we are interested in run on DEC hardware, and are primarily used to access databases for decision support. DATATRIEVE certainly falls within that category as does ADE. We are not interested for example, in fourth generation products that control robots, or operate process control systems.

The Proposal:

We propose:

1. To change the name of the DATATRIEVE SIG to be the "DATATRIEVE and Fourth Generation Language SIG", or DTR/4GL" for short.

2. To adopt the following mission:

   The DTR/4GL SIG serves the informational needs of all DECUS members who are concerned with accessing and manipulating information in databases through the use of DATATRIEVE and other fourth generation languages. The SIG also serves as a two-way communications facilitator between DEC and this user community.

SIG membership will initially consist of present DTR SIG members, most of whom are DATATRIEVE users. As DEC announces new products in this same product area, DTR/4GL would assume a leadership role in the user development and use of these products. While the user base will diversify, it is expected that the essential needs and motivations of members of the SIG will stay the same as the present DTR SIG members. DTR/4GL would also pick up users who are interested in third party products which are at present mostly ignored by DECUS. The present DTR steering committee will initially staff the steering committee of the new SIG.

The primary DEC interface for the SIG initially would be with a DATATRIEVE product manager in the VAX Information Architecture development group in

Engineering. The new products under development are currently handled by this same group in DEC, so continuity will be easy to achieve and would seem to be adequate for the new SIG, at least in the initial stages.

The services to be provided by DTR/4GL will consist of the traditional services provided by an applications oriented SIG. A newsletter, symposia sessions, a communications relationship with DEC, library support, and user assistance would be the mainstay products. The present DATATRIEVE SIG is providing these services currently, and as products appear, the services would be gradually expanded to meet growing needs.

Issues:

SIG Overlap:

The primary concern in this area is (as it has always been for DTR) with DMS. In the view of the two present SIG chairs, this change in mission actually clarifies the relationship between the two SIGs. DTR/4GL is interested in users and applications designers who are satisfying end user needs. DMS, on the other hand, is really concerned with database management which implies products such as RMS, DBMS, and RDB and problems such as data architectures and corporate data management. A fundamental fact, which both SIG's recognize, is that the basic user groups are quite separate and distinct, although also clearly related. DTR/4GL will continue to have a unique, close relationship with DMS.

A concern which was largely responsible for starting the movement to change the SIG's role is the need to avoid confusion and overlap caused by a single product SIG (DTR) being overtaken by a new SIG formed to care for the users of a new product in the same general area. Changing the mission of the DTR SIG to a functional orientation from a product orientation precludes this from happening.

Feasibility:

DATATRIEVE is currently a moderately sized SIG with lots of active energetic members. It has a reputation for getting things done effectively. That energy will flow into the new SIG and provide the initial boost needed to keep things moving. Additionally the excitement and user enthusiasm created by a new series of products should add more fuel to the fire. I see no problem in getting this SIG to go.

A key leadership challenge -- which is recognized by the steering committee -- is the need to keep the fun loving, dynamic and spontaneous atmosphere of DATATRIEVE SIG alive in the new SIG. We must (and will) emphasize to our old DATATRIEVE membership that the Wombat has not abandoned him, he is just diversifying a bit.

Implementation Strategy:

Provided that the concepts expressed by this paper are accepted by the SIG Council, the Management Council, and the Board, we intend to execute the following steps to implement the new SIG:

1. Announce to DECUS membership during the New Orleans Symposium our intention to modify the mission of the SIG.

2. At a Woods Meeting after New Orleans make the necessary changes to our operating procedures.

3. Following that, to proceed with the relicensing process. It is anticipated that minimal changes to the present DTR budget will be required.

4. Assume the role of the new SIG at the 1985 Fall Symposium.

Impact on DECUS:

We believe that the impact of this action upon DECUS will be positive. It adds no additional management overhead, and it clarifies the roles and missions of SIG's in the area of data management and decision support. It also provides a structure more adaptable to future DECUS growth and more flexible in meeting DEC's changing product marketing plans.

Relationship to Strategic Plan:

We feel that this new structure addresses many of the initiatives expressed in the Strategic Plan:

Membership:

This structure is more adaptable to serving a DECUS membership of 100,000 by 1990. This structure provides a clearer focus on end-user application activities in the database area.

DECUS/Vendor relationship:

The DTR/4GL SIG is more flexible in meeting DEC's changing marketing patterns. It provides a communications link to a market segment rather than the users who have already bought a specific DEC product.

Services:

DTR/4GL SIG will provide more effective and complete services to users because it is addressing the needs of a complete segment of users rather than the needs of the users of a single product.

Leadership:

This SIG will draw people into leadership from areas that have not been previously tapped because their needs were not addressed. This is particularly true for users of third party products.

DECUS Relationships to the Outside World:

Because it addresses products previously ignored by DECUS, this SIG enhances DECUS relationships to the outside world.

DATATRIEVE SIG Sponsored Presentations at New Orleans Symposium

Chris Wool, Symposium Committee Representative, E. I. duPont, Wilmington, DE

The following are the abstracts of presentations which will be sponsored by the DATATRIEVE SIG at the 1985 Spring DECUS Symposium, May 27-31, 1985, in New Orleans, Louisiana. Of particular interest is the **Fourth Generation Language Panel** at 1:00 p.m. on the 27th and **Talking to DATATRIEVE-32 and VIA in English** at 6:00 p.m. on the 27th in light of the "DATATRIEVE and Fourth Generation Language SIG Proposal" by Larry Jasmann which appears on page 2 of this issue of the Wombat Examiner. In addition to the presentations listed below, there are a large number of presentation which are sponsored by the DMS SIG which DATATRIEVE users will find of interest. In particular, there are several sessions on Thursday, May 30, on VAX Information Architecture.

DATA MANAGEMENT SYSTEMS SIG AND DATATRIEVE SIG JOINT OPENING SESSION

Stephen Pacheco, Chairman
Data Management Systems SIG

Lawrence Jasmann, Chairman
DATATRIEVE SIG

DM020 Monday, May 27, 10:00 to 11:00 a.m., Grand Salon C
The DMS and DTR SIG's will outline their activity for the week as well as highlight sessions and other activities of interest to those attending the New Orleans Symposia. The steering committee people for both SIG's as well as the SIG's Digital counter parts will be introduced. The campground locations and schedules will be provided. The DMS SIG will announce which documentation kits will be given away, as well as the time and place of the announcement of the winners. Questions and comments from symposia attendees will be welcome by both SIG's.

FOURTH GENERATION LANGUAGE PANEL

Lawrence Jasmann
U. S. Coast Guard
Burke, VA

Charles Duncan
Digital Equipment Corporation
Nashua, NH

DT008 Monday, May 27, 1:00 to 2:00 p.m., Grand Salon C
This session will be a probing discussion of the meaning, application and use of Fourth Generation Languages as they are applied to data base and decision support tasks. Speakers will be balanced between DEC and the user community, and both DEC's perspective and user views will be examined. DATATRIEVE, DEC's primary product in this area will be discussed, as well as other products. Some questions which will be addressed include:

* What should a Fourth Generation Language do?
* What additional functionality is needed?
* How does one define a Fourth Generation Language and distinguish it from other computer languages?

Orientation: Intermediate/General

THE 20 MOST ASKED DATATRIEVE QUESTIONS

Katherine Wrobel
Digital Equipment Corporation
Colorado Springs, CO

DT019 Monday, May 27, 4:00 to 5:00 p.m., Elmwood
We, at the Customer Support Center in Colorado Springs, have spent some time to come up with a list of the 20 most asked DATATRIEVE questions. Those questions cover everything from common pitfalls to popular how-to's and gotchas, most of which are also in Digital's Software Information Network. We had a lot of fun preparing this list and hope that you will find it interesting and enjoyable.

Orientation: Intermediate/Technical

DATATRIEVE-11 STATUS UPDATE

Robert M. Laham
Digital Equipment Corporation
Stow, MA

DT013 Monday, May 27, 5:00 to 6:00 p.m., Elmwood
DATATRIEVE-11 is an interactive query, report, and data maintenance system that is well suited for low-cost, multi-user systems. At this session, Digital's DATATRIEVE-11 Product Management and Development Group will speak about the most recent releases of products within the family of DATATRIEVE-11, Micro/RSX DATATRIEVE-11, PDP-11 DATATRIEVE/VAX, and PRO/DATATRIEVE. Additionally, they will talk about future offerings for the family and will discuss some of the commonalities and differences among the DATATRIEVE-11 products.

Orientation: Novice/General/Managerial

TALKING TO DATATRIEVE-32 AND VIA IN ENGLISH

Tom Kush
BBN Software Products
Cambridge, MA

DT030 Monday, May 27, 6:00 to 6:30 p.m., Magnolia
This paper discusses an English-language retrieval interface to DATATRIEVE-32 called the IRUS system. IRUS makes it possible for end-users to retrieve information from DATATRIEVE without knowing either DATATRIEVE or the structure of the underlying files: for example, by typing a question such as "Which male

employees have health benefits?" The presentation will provide an overview of the IRUS technology and detailed discussion of implementation strategies in working with DATATRIEVE, the CDD and VAX file systems.

Orientation: Intermediate/General

DATATRIEVE APPLICATION DESIGN WORKSHOP

Margaret M. Racel and Mitzi Ware
E G & G
Las Vegas, NV

DT004 Monday, May 27, 7:00 to 8:00 p.m.
This session will describe the way in which a typical application might be analyzed and implemented. The presentation will include system design and initial considerations such as what data to include, when to use multiple files, and how to use RMS utilities to optimize files. The system described includes a DCL menu to control system entry, a DATATRIEVE/FMS procedure to update the master data file, DATATRIEVE procedures to report on the information, and the RMS 'CONVERT' utility to optimize file structure.

Orientation: Intermediate/Technical

TRAINING THE INEXPERIENCED USER IN DATATRIEVE

Lori Korbas
North Shore Sanitary District
Gurnee, IL

DT012 Monday, May 27, 8:00 to 8:30 p.m., Grand Salon C
Do you feel that your users would benefit greatly from DATATRIEVE but need extra guidance to answer the questions that so many of us have had? DATATRIEVE has so many potentials but many non-DP personnel get frustrated without a helping hand and predetermined examples. The North Shore Sanitary District has devised a 10-lesson introductory DATATRIEVE training program. The lessons step the user through different commands such as SHOW, PRINT, FIND, and SORT. The instructor can help relate how DATATRIEVE can be used by a particular individual. These DATATRIEVE lessons are just one way to broaden your users' capabilities.

Orientation: Intermediate/General

ON THE EDUCATION OF NEW DATATRIEVE USERS

Stephen Pacheco
Ship Analytics
North Stonington, CT

DT016 Monday, May 27, 8:30 to 9:30 p.m., Grand Salon C
Educating people to use DATATRIEVE usually presents very interesting problems. The main reason for these problems is because DATATRIEVE is powerful enough to let you consider the difficult problems very quickly, which

is not the case with most other programming languages. New DATATRIEVE users range in experience from none to system programmers. The range of problems includes simple list management to complete application development (e.g., a financial package) in DATATRIEVE. These factors, user experience and range of problem types, makes education most challenging and possibly frustrating for the instructor.

Mr. Pacheco has been teaching DATATRIEVE for 4 years. This session will outline a number of points which have proved to be significant in terms of educating new users:

* How do you deal with the naive user.
* What are the most difficult concepts for a new user to understand.
* What concepts are most often misunderstood.

This session will be of interest to those people who find themselves in the position to introduce people to DATATRIEVE. Particular examples of problems and means of explaining them will be used.

Orientation: Intermediate/Managerial

SO WHAT IS DATATRIEVE ANYWAY?

Mike Daugherty & Charles Duncan
Digital Equipment Corporation
Nashua, NH

DT021 Tuesday, May 28, 9:00 to 10:00 a.m., Grand Salon C
VAX DATATRIEVE is Digital's fourth generation application development language and programmer productivity tool. DATATRIEVE also has query and reporting capabilities. The purpose of this session is to introduce the features, capabilities, and uses of DATATRIEVE across the various machine architectures that support it.

Orientation: Novice/General

SO WHY USE DATATRIEVE ANYWAY?

Lawrence Jasmann
U. S. Coast Guard
Burke, VA

DT003 Tuesday, May 28, 10:00 to 11:00 a.m., Grand Salon C
Having heard a basic description of what DATATRIEVE is, this session provides an insight on potential uses for DATATRIEVE and a brief discussion on various strategies to use DATATRIEVE to improve user effectiveness.

Orientation: Managerial

DATATRIEVE RECORD DEFINITION WORKSHOP

Diana Washburn
Hanes Hosiery
Winston-Salem, NC

DT009 Tuesday, May 28, 11:00 a.m. to 12:30 p.m., Grand Salon C
The record definition in VAX DATATRIEVE is an extremely powerful tool. The many features of the record definition clauses alone can dazzle programmers who have used record definitions in other languages. This session expands upon the material in the DATATRIEVE manuals and presents some interesting possibilities for enhancing applications through the use of record definitions. Attendees will be exposed to special techniques, potential pitfalls and beneficial alternatives for the definition and utilization of data. To benefit from this session, the attendee should be more than a beginning DATATRIEVE user and should have some experience defining and using record definitions. The material is geared to VAX DATATRIEVE.

Orientation: Intermediate/Technical

CONSERVING AND MAXIMIZING THE USE OF POOL SPACE IN DATATRIEVE-11 AND PRO-DATATRIEVE

Philip Dickerson
Northern Telecom Incorporated
Concord, NH

DT017 Tuesday, May 28, 12:30 to 1:00 p.m., Grand Salon C
The fixed task size creates a limitation of "pool space" in DATATRIEVE on PDP-11 and PRO-300 machines. This pool space limitation places restrictions on the use of large domains, complex record structures, multiple domains, complex report writer statements, and sorting. Digital's suggestions for conserving pool space will be briefly mentioned, and then several techniques developed by the speaker for maximizing the use of pool space will be explained.

Orientation: Intermediate/Technical

USING VAX DATATRIEVE IN AN EARTH SCIENCE RESEARCH AND PUBLIC SERVICE INFORMATION ENVIRONMENT

Steven Cordiviola
KY Geological Survey U of KY
Lexington, KY

DT028 Tuesday, May 28, 2:00 to 2:30 p.m., Magnolia
The Kentucky Geological Survey was established in 1854 as the official geologic research organization in the Commonwealth. Each section at the Survey is charged with collecting various types of data related to the earth sciences. The diverse nature of this data requires it be archived using a number of different record structure formats, yet be interrelated when necessary. To manage this broad based earth science information, the Survey choose VAX DATATRIEVE for data storage and manipulation. DATATRIEVE'S

flexibility and functionality as a layered product provides a comprehensive data management system which will grow as the Survey's applications grow. By CROSSing fields containing identical geographic coordinate information, each section's data are linked together and can be easily accessed. By having standard data definitions, users do not have to learn a variety of query names or procedures to prepare reports or gain access to the data. In addition, with the proper use of the Common Data Dictionary and its associated Access Control Lists, data file access is controlled; tables common to all users are only stored once and can be easily maintained; and all users have access to field names and data definitions.

Orientation: Intermediate/General

USING DOMAIN TABLES TO CONNECT INTERACTIVE AND BATCH DATATRIEVE PROCEDURES

Elliot F. Jaquith
E.I. du Pont de Nemours & Co.
Brevard, NC

DT010 Tuesday, May 28, 2:30 to 3:00 p.m., Magnolia
This paper describes how to use a domain table as a bridge between interactive and batch DATATRIEVE when requesting queries on a very large file with variable search parameters. The need to use this method is because our very large file queries require at least 45 minutes search time. By using a domain table to hold the variable information, which can be entered from the keyboard with interactive DATATRIEVE using two FMS screens, the terminal connect time is reduced to 10 minutes. This then leaves the terminal free for other tasks. The unique use of the domain table is important because this table is used by another batch program to supply variable information to prepare a special report. Before using DATATRIEVE, this report was prepared by hand in four and one half days.

Orientation: Intermediate/Technical

CUSTOMIZING VAX DATATRIEVE

Sandy Kaplan & Kirk Searle
Digital Equipment Corporation
Nashua, NH

DT023 Tuesday, May 28, 3:00 to 4:00 p.m., Magnolia
VAX DATATRIEVE can be customized to suit the needs of individual users or customized to suit all users on a system. DATATRIEVE allows users to modify or create HELP and message texts, user-defined keywords, and user-defined functions. This presentation will focus on the design of customized features and will include examples of customizing DATATRIEVE for the international business environment.

Orientation: Intermediate/Technical

MAILBOXES AND DATATRIEVE

Lali Singh
University Hospital
Madison, WI

DT026 Wednesday, May 29, 9:00 to 9:30 a.m., Elmwood
This presentation discusses a menu driven application for Operating Room scheduling at the University Hospital in Wisconsin. Key features of this discussion are:

* Using the DATATRIEVE call interface for all I/O
* Using FMS for screer
* Using mailboxes to interface to DATATRIEVE

Orientation: Advanced/Technical

USING THE DATATRIEVE CALL INTERFACE

Carolyn Sprague
Dupont/NEN Products
Billerica, MA

DT027 Wednesday, May 29, 9:30 to 10:00 a.m., Elmwood
In this session, the use of the DATATRIEVE call interface will be demonstrated using a sample FORTRAN application. In this application, FORTRAN is used for initial data entry and storage. DATATRIEVE is used for data manipulation and report generation.

Orientation: Intermediate/Technical

USING VAX DATATRIEVE GRAPHICS

Sandy Kaplan & Kirk Searle
Digital Equipment Corporation
Nashua, NH

DT022 Wednesday, May 29, 10:00 to 11:00 a.m., Elmwood
The VAX DATATRIEVE PLOT statement gives users a quick and easy-to-use method of displaying information as a graph. A wide range of predefined plot formats are used to enhance decision support applications and can be produced in hard-copy form for presentations and reports. This talk will present effective techniques for producing graphs from data managed with VAX DATATRIEVE.

Orientation: Intermediate/Technical

VAX DATATRIEVE GRAPHICS INTERNALS

Joe H. Gallagher
Cleveland Clinic Foundation
Cleveland, OH

DT011 Wednesday, May 29, 11:00 a.m. to 12:00 noon, Elmwood
VAX DATATRIEVE plots are implemented in a PASCAL-like language which is internal to the plotting library. This plot language is characterized by multiple entry point routines, strong typed variables, dynamic arrays, a DO-loop construct, arithmetic and relational operators, and a set of functions which are available only from within the plotting language. Complex plots are constructed of logic in the plot language and calls to a set of utility plots which do most of the work of creating ReGIS instructions. A full explanation of the plot language and several new plots will present the attendee with the knowledge and courage to begin modifying and extending the plot library.

Orientation: Advanced/Technical

DATATRIEVE DISTRIBUTED MANIPULATION FACILITY

Mike Daugherty & Andy Schneider
Digital Equipment Corporation
Nashua, NH

DT024 Wednesday, May 29, 2:00 to 3:30 p.m., Elmwood
This tutorial will discuss the DATATRIEVE Distributed Manipulation Facility, its features and use. This facility provides an ability to access and process data that may reside on different systems and in different database storage systems. When using VAX DATATRIEVE and/or DATATRIEVE-11 this facility assure that stored data is manipulated in a consistent manner without regard for the location of the data.

Orientation: Intermediate/Technical

EQUIPMENT MAINTENANCE PROGRAM FOR A MICROELECTRONICS MANUFACTURING FACILITY

James Griffin
Department of Defense/ATT S454
Fort Meade, MD

DT029 Wednesday, May 29, 3:30 to 4:00 p.m., Elmwood
This paper outlines an equipment maintenance program for a micro-electronic manufacturing facility. It is an applications program utilization DATATRIEVE, FMS, and VMS command procedures. There are two data bases; an equipment inventory file, and a maintenance history file. Several DATATRIEVE procedures, with FMS forms, are used to maintain the inventory file and update the history file. DATATRIEVE procedures are used to generate maintenance reports. Access to the system is controlled from an option list in a login command file.

Orientation: Intermediate/General

HOW SYSTEM MANAGERS USE DATATRIEVE

Bert Roseberry
U.S. Coast Guard District (DT)
New Orleans, LA

DT005 Wednesday, May 29, 4:00 to 5:00 p.m., Elmwood
System managers use DATATRIEVE to manage their systems in a multitude of ways. System managers from both PDP-11 and VAX systems will present some of the ways they use DATATRIEVE to make their system management job easier.

Orientation: Intermediate/Technical

DATATRIEVE-11 PERFORMANCE OPTIMIZATION

Robert M. Laham
Digital Equipment Corporation
Stow, MA

DT014 Wednesday, May 29, 5:00 to 6:00 p.m., Elmwood
DATATRIEVE-11 is an interactive query, report, and data maintenance system that is well suited for low-cost, multi-user systems. At this session, a software engineer from Digital's DATATRIEVE-11 Development Group will provide helpful hints on how to optimize DATATRIEVE-11 applications for improved performance.

Orientation: Intermediate/Technical

VAX DATATRIEVE-APPLICATION DESIGN TRADE-OFFS

Steve Baron & Kathy Jo Nelson
Digital Equipment Corporation
Nashua, NH

DT025 Thursday, May 30, 10:00 to 11:30 a.m., Grand Salon D
This will be a presentation and discussion by Digital performance analysts of various aspects of VAX DATATRIEVE application design and implementation trade-offs for improved performance. This session is intended for programmers, systems analysts, and software specialists who currently use DATATRIEVE or are planning to implement an application using DATATRIEVE as an application development language. Discussion will include: general DATATRIEVE application design issues (e.g., FOR vs FIND, query optimization, CROSS, domains in nested FOR loops, dictionary vs domain tables, call interface), VAX RMS file design and "tuning," VAX DBMS database tuning for use with DATATRIEVE, VAX CDD dictionary design, and related system tuning.

Orientation: Advanced/Technical

VAX DATATRIEVE INTERNALS AND OPTIMIZATION

Mike Daugherty & Andy Schneider
Digital Equipment Corporation
Nashua, NH

DT020 Thursday, May 30, 2:00 to 3:30 p.m., Grand Salon C
This session will provide an overview of some of the internals of VAX DATATRIEVE, and provide some suggestions for optimizing DATATRIEVE applications based on these internals. Some of the areas to be covered include DATATRIEVE invocation and start-up, procedure and loop organization alternatives, and trade-offs during the execution phase of DATATRIEVE.

Orientation: Intermediate/Technical

WOMBAT MAGIC

DT001 Thursday, May 30, 8:00 to 10:00 p.m., Grand Salon C
Once again the GREAT WOMBAT wants all you WOMBAT Wizards and novices alike to participate in fun-filled evening sharing DATATRIEVE techniques and experiences which reach beyond the expected. All are asked to come prepared to mystify and amaze other with stores and examples of how to do the impossible with DATATRIEVE. Included in the festivities will be door prizes and a special prizes for the Wizard presenting the best magic.

Orientation: Technical/Novice thru Advanced

MANAGING A LARGE NAVIGATIONAL AIDS SYSTEM WITH DATATRIEVE

David Gosselin
United States Coast Guard
New Orleans, LA

DT018 Friday, May 31, 10:00 to 11:00 a.m., Elmwood
This session will discuss managing a large navigational aids system using DATATRIEVE. Topics to be discussed include:

* Overview of aids to navigation information needs - status of aids, associated costs, inventory, worklists,
* Design of the system - stable in size, partially normalized, use of tables
* FMS benefits - streamlined procedures, readability, faster
* Inquiries made of the system (with and without the report writer) - daily status, listing of equipment, worklists, associated costs
* Use/Inquiries of the system by non-programmers

Orientation: Intermediate/General

VMS ACCOUNTING USING VAX DATATRIEVE

Bert Roseberry
U.S. Coast Guard District (DT)
New Orleans, LA

DT007 Friday, May 31, 11:00 a.m. to 12:00 noon, Elmwood
A combination of VAX DATATRIEVE and VMS command procedures was developed to produce charge back accounting on the VAX. This method of producing charge back accounting was chosen because of its low-cost development and subsequent upkeep. The procedures do not require any programs other than VMS utilities and DATATRIEVE, and only minor modifications in DATATRIEVE will be necessary for future releases of VMS. The procedures are easily customized for site-specific purposes. Users can specify who, what, and when to charge. Bills, statistics, and other reports are all handled by DATATRIEVE and can be easily modified for any purpose.

Orientation: Intermediate/Technical

DATATRIEVE SIG CLOSING SESSION

Lawrence Jasmann
Chairman, DATATRIEVE SIG

DT002 Friday, May 31, 12:00 noon to 1:00 p.m., Elmwood
This is the windup session for the DATATRIEVE SIG. Wish list items will receive comments for DIGITAL and any last minute questions or comments will be attended to. This session also kicks off activities for the following symposium in Anaheim.

Hints and Kinks

---

* Dick Azzi reminds us that the RUNNING TOTAL of field which is USAGE REAL does not work properly. An error message like "Internal error (expected block id NN, encountered id MM)." is obtained.

* Bert Rosebery reminds us that parenthesis around the left half (only) of a Boolean expression does not work properly. Consider the following:

```
IF ( 3 - 2) EQ 1 THEN ...          !Does not work

IF ( 3 - 2 EQ 1) THEN ...          !Works

IF 3 - 2 EQ 1 THEN ...             !Works

IF ((3 - 2) EQ 1) THEN ...         !Works
```

DATATRIEVE Application Development Standards

Williams College Administrative Computing

1.0 STANDARDS AND EGOLESS PROGRAMMING

Four of the most important objectives in application development are to develop modules that are functionally complete, functionally accurate, easily understood, and easily maintained. Although most programmers would agree with these objectives, individual programming style often dominates a programmer's production and applications often fail to meet these common standards. This tendency towards individuality is very natural and appears within almost everyone's work, especially when one remains isolated from one's programming piers for long periods of time.

A proven solution to these very common problems involves group reviews or 'structured walkthroughs'. Individuals should be encouraged to share frequently their portion of the group project with other members of a group. Very often other members will easily recognize erroneous logic, omissions, needed controls, and comments. A sharing of ideas helps to overcome idiosyncratic coding patterns and encourages programmers to keep their work within established standards.

Reviews help overcome the very natural tendency to become possessive and sensitive towards ones work. This possessive tendency is characteristic of 'ego programming'. Overcoming this ownership or possessiveness leads to 'egoless programming'. Egoless programming is characterized by an environment in which programmers have not developed sensitive and personal attachments to 'their' work - a very difficult yet very important objective in all major application development.

Two complete examples of DATATRIEVE code written according to recommended standards have been included at the end of this paper.

2.0 WARNIER ORR

All DATATRIEVE applications, regardless of their size, should be designed as highly structured modules of code. The most effective tool available for developing such code is the Warnier-Orr Structured Diagram. Completed diagrams should be reviewed and critiqued by colleagues before any coding begins. Warnier-Orr diagrams should become part of a systems documentation. They should be referenced and updated whenever enhancements or maintenance 'patches' are applied.

If Warnier-Orr diagrams are maintained they will serve as an excellent overview of application systems. Both beginning and experienced staff will appreciate their clarity, simplicity, structure and ease of use.

An introduction to Warnier-Orr diagrams is presented in **PROGRAM DESIGN AND CONSTRUCTION**, by David A. Higgins, published by Prentice-Hall.

3.0 DESIGN METHODOLOGIES

While following the requirements for structured design, all procedures should also be developed according to an established methodology. It is

especially important to establish a methodology within each major application package. Guidelines for a methodology are listed below.

3.1 Consistency

Consistency is very important. User responses, error messages, prompting, looping, and other structural components of an application should be consistent throughout the application.

3.2 No 'Bells And Whistles'

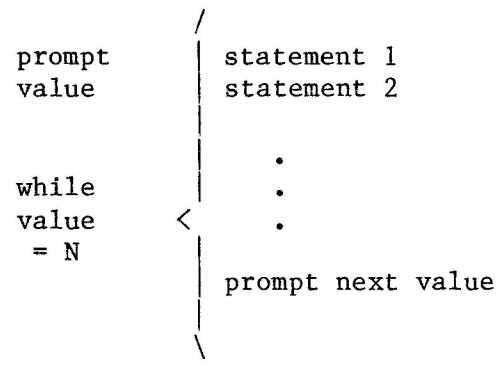
An application should not contain frivolous code. More often than not, these 'special features' appear as inconsistent responses to users and complicate maintenance procedures. Whenever an enhancement is justified, it should be developed as a separate and independent procedure (to be used as a subroutine, described below). Developing enhancements as subroutines enables programmers to eliminate them with a single comment during maintenance.

3.3 Meaningful Error Messages

Error messages should inform the user of the problem and suggest corrective action. Messages should be brief, friendly, and complete.

3.4 Standard Looping Structure

The following looping structure should be applied whenever possible.



Note that if the WHILE clause is not used, the PROMPT NEXT VALUE clause is not required.

3.5 Enclose Subroutine Procedures

All subroutine procedures should be enclosed with BEGIN and END statements. The BEGIN statement should appear immediately following the header block. The END statement should appear just before the END_PROCEDURE statement. This technique guarantees every statement in the subroutine will be executed. Top level procedures invoked by the user are an exception to this recommendation if they contain DATATRIEVE commands such as READY.

3.6 Prompting The User

Whenever a user is given a prompt requesting data entry, the prompt should:

1. Clearly indicate exactly what it is the program is expecting. Often a single field is needed and a field name will suffice.

2. If the format for data entry is not immediately obvious, the expected format should be included in the prompt.

3. The default response should be displayed in brackets and lower case.

4. The users response should be evaluated using the CONTAINING Boolean operator, not the EQUAL Boolean operator. The CONTAINING clause will match on upper and lower case responses.

4.0 PROCEDURES AS SUBROUTINES

Structured design is easily implemented when DATATRIEVE procedures (procs) are used as subroutines. Procs can be coded in such a way that a 'one to one' mapping can be achieved from the Warnier-Orr design to the actual code. Code should be removed from 'the main line' and placed into callable DATATRIEVE proc (subroutine) whenever:

1. A procedure contains more than one or two screens of code.

2. A function is easily coded as a separate entity.

3. A function requiring several lines of code is required in more than one location.

4. A PRINT statement is required that contains several lines of code.

5.0 COMMENTS

Commenting is an extremely important component of any software package. DATATRIEVE is no exception. Every DATATRIEVE proc should begin with a 'header block' and should contain a liberal amount of comment blocks and comment lines.

All comments of any type (header blocks, comment blocks, comment lines) should be formated in the following manner:

1. Enclose all comments with blank comment lines.

2. Indent (or TAB) all comments such that they do not appear flush left. Comments should appear visually separate from all DATATRIEVE code. When comments are flush left they appear to merge into commands and become more difficult to read.

5.1 Header Blocks

Every DATATRIEVE proc should begin with a header block. All header blocks should contain the following format:

1. Header blocks should be enclosed with an 'asterisk block'. Create the asterisk block such that it is TABed in (or indented) at least one tab stop.

2. Begin each header block with a 'proc id'. Include its CDD dictionary name and proc name.

3. Indicate who (or what) calls this proc. If a proc is called by several 'calling procs', list each of the calling procs within the called proc.

4. Briefly describe the function(s) being performed within the proc. If several major and distinct functions are being performed, list and describe (using a numerically sequenced outline format) each major function. Use brief yet accurate descriptions.

5. If the proc requires a calling proc to provide temporary variables or a specific record context, these requirements should be noted within the header block.

5.2 Comment Blocks

Whenever a comment extends to more than one line a comment block should be constructed. A comment block contains the following format:

1. Comment blocks should be enclosed with an 'asterisk block'. Create the asterisk block such that it is TABed in (or indented) at least one tab stop.

2. If two or more functions are being discussed within the comment block, itemize (using a numerical sequence) the comments such that they reflect the logical structure being referenced.

5.3 Comment Lines

Whenever comments require only a single line, enclose the comment with blank lines and format the comment such that it begins with a string of 5 asterisks, a space, the comment, a space, and terminates with a string of 5 asterisks.

6.0 INDENTATION

The visual appearance of code is extremely important. Appearance should reflect logical structure. Indentation is a very simple yet effective technique for visually displaying structure.

Indentation should be applied whenever:

1. Code is subordinate to a looping structure (within FOR, REPEAT, and WHILE loops).

2. Code is subordinate to a conditional statement (within an IF or IF-ELSE statement).

3. The PRINT statement is used. Two very simple standards should be applied to the formatting of PRINT commands. They are:

   a. Every item within a PRINT command that causes a new print line to appear should be formated flush left (relative to its current margin).

   b. Every other item within a PRINT command (those items which cause fields to be printed within a print line) should be formated with a slight indentation, one field per line.

7.0 DECLARATIVES AT THE TOP

All declaratives should be included at the top of DATATRIEVE procs.

8.0 VARIABLE NAMES AND CONDITIONAL VALUES

The purpose and meaning of a variable should be reflected in the name of the variable. Temporary variables (those used within a proc but not stored in a data record) should begin with the character string "TMP-". Two and three character, cryptic, 'fortran type' names should NEVER be used in DATATRIEVE.

Examples of variable names might be: INTEREST-TOTALS, TMP-DEBUG-SWITCH, TMP-SESSION-DATE, PRINCIPAL-DUE

Whenever conditional values are assigned to 'switches', they should equal a meaningful character string such as: "DO NOT RUN INTEREST" or "OK TO PROCESS". An example might be: "IF TMP-INTEREST-FLAG = "OK TO PROCESS" ..."

Conditional values should NOT be equal to something like "ON", "OFF", "YES" "1", or "0".

9.0 PROCEDURE NAMES

Naming conventions for DATATRIEVE procs are determined by their calling mechanism. For ease of use, procs called by users can be given short abbreviations that are easily remembered. However, all other procs should be given full and meaningful names. Names can be as long as 31 characters.

In addition to meaningful names procs should be given functional prefixes. The need for these prefixes is described below.

Major DATATRIEVE applications consist of several related, yet quite independent, functions such as: maintenance tasks, reporting needs, and various processing needs. Each of these functions is defined by a structured set of procs. The highest level proc is usually invoked directly by the user, each subordinate proc is called by the proc above it. Sometimes procs are called by more than one proc within an application.

Quite often the number of procs within an application grows to such an extent that it becomes difficult to locate all procs related to a single function (such as those described above). Occasionally all people within a working group disclaim any knowledge of a proc; no one can remember why the proc was developed!

For this reason all procs should be given a prefix. The prefix enables all procs within a function to be clustered together. Whenever a SHOW PROCEDURES command is issued, all related procs appear together. The prefix that appears to work best contains the following format: FFFN-... . The FFF consists of the initials of the function name and the N indicates the level of the proc (level 0 is the top, level 0 procs call level 1 procs, ...).

An example is the following: A major function within the Loan System is

to report Journal entries by ID. This function, Journal By ID consists of a top level proc (called by the user) and several subordinate, structured procs. The name of the top proc is JBI (Journal By Id). The proc JBI calls procs named JBI1-...-..., JBI1-...-... . Each of these procs calls procs named JBI2-...-...

Procs that are called by more than one unrelated function cannot contain the prefix of each of the calling procs. Therefore these procs are given the prefix of "SHR-" indicating they are shared by more than one calling proc.

The comment blocks of these shared procs must accurately indicate the names of all calling procs. If the shared proc is ever changed, all calling procs may also need modification.

10.0 INCLUDE COMMENTS ON END STATEMENTS

Structured DATATRIEVE code consists of a hierarchy of subroutines. Each subroutine should begin with a BEGIN statement and end with an END statement. When developing procs consisting of multiple subroutines it is quite possible to misplace a single BEGIN or END statement. DATATRIEVE, however, is quite good about matching BEGIN END statements. Unfortunately it is not very friendly when it discovers a mismatch. Normally it responds "... encountered END ... expected statement". If END statements contain comments (an "!" followed by an ACCURATE description of its location) the comment will appear along with the error message, greatly reducing the task of debugging.

The usefulness of the commenting scheme is only as effective as the comment is accurate. Do not comment END statements with phrases such as "END ! end of main line". A more descriptive comment might be "END ! end of JBI3-PRINT-JOURNAL-HEADER".

EXAMPLE 1

```
DELETE JBI;
DEFINE PROCEDURE JBI
!
!  ************************************************************
!  * CDD$TOP.ADMIN.LOAN  JBI (Journal By ID)
!  *
!  * Report Journal entries for a selected ID, within a selected
!  * date range.
!  *
!  * This report should serve as an audit trail of an individual's
!  * account.  It will show dollar values on a beginning date, all
!  * action against the account, followed by ending values.
!  ************************************************************
!
READY BIO SHARED READ
READY LOANS SHARED READ
READY OLDLOANS SHARED READ
READY JOURNAL SHARED READ
!
DECLARE TMP_DEBUG_FLG   PIC X(10).
DECLARE TMP_KEY     PIC X(7).
TMP_KEY = *."KEY"
!
```

```
SET COLUMNS_PAGE = 132
FN$WIDTH(132)
!
! ********************************************************************
! * 1. print a 'before' record
! * 2. print all action records (add,modify,post) that follow
! * 3. print the last 'after' record
! ********************************************************************
!
WHILE TMP_KEY NOT = " " BEGIN
!
  FOR BIO WITH KEY = TMP_KEY BEGIN
    :JBI1_PRINT_BIO_HEADER
    FOR LOANS WITH KEY = TMP_KEY BEGIN
      :JBI1_PRINT_ORIGINAL_LOAN
      :JBI1_PRINT_JOURNAL_ENTRIES
      :JBI1_PRINT_CURRENT_LOAN
      END           ! FOR LOANS
    END             ! FOR BIO
  TMP_KEY = *."NEXT KEY, <SPACE> TO EXIT"
  END               ! WHILE TMP_KEY
!
SET COLUMNS_PAGE = 80
FN$WIDTH(80)
!
END-PROCEDURE


DELETE JBI1_PRINT_BIO_HEADER;
DEFINE PROCEDURE JBI1_PRINT_BIO_HEADER
!
!  ******************************************************
!  * CDD$TOP.ADMIN.LOAN JBI1-PRINT-BIO-HEADER
!  *
!  * Called By: JBI
!  *
!  * print a biographic header
!  ******************************************************
BEGIN
!
PRINT SKIP,
  COL 32, "***** JOURNAL ENTRIES FOR:   " | KEY | "  " | NAME ||| " *****",
SKIP 2,
  COL 1, "SESS",
  COL 6, "CODE",
  COL 12, "DATE",
  COL 19, "USER",
  COL 24, "KEY",
  COL 32, "ST",
  COL 35, "TY",
  COL 38, "RATE",
  COL 52, "ORIGINAL",
  COL 66, "B PRIN",
  COL 76, "PRIN DUE",
  COL 89, "INT DUE",
```

```
  COL 100, "LAST PAID",
  COL 114, "LATE"
!
END       ! JBI1-PRINT-BIO-HEADER
!
END-PROCEDURE


DELETE JBI1_PRINT_CURRENT_LOAN;
DEFINE PROCEDURE JBI1_PRINT_CURRENT_LOAN
!
!  ************************************************************
!  * CDD$TOP.ADMIN.LOAN JBI1-PRINT-CURRENT-LOAN
!  *
!  * CALLED BY: JBI
!  *
!  * print an original loan (old) loan record
!  * (CALLING PROC MUST SET TERM WIDTH TO 132 ("FN$WIDTH(132)")
!  * AND MUST HAVE A CURRENT LOAN RECORD)
!  ************************************************************
!
BEGIN
!
PRINT COL 51, "================================================",
  COL 101, "================="
!
PRINT COL 32, LOANS.STATUS (-),
  COL 35, LOANS.TYPE (-),
  COL 38, LOANS.RATE (-),
  COL 51, LOANS.ORIGINAL (-) USING $$,$$$.99-,
  COL 63, LOANS.BAL_PRIN (-) USING $$,$$$.99-,
  COL 75, LOANS.PRINCIPAL_DUE (-) USING $$,$$$.99-,
  COL 87, LOANS.INTEREST_DUE (-) USING $$,$$$.99-,
  COL 100, LOANS.PAID_SINCE_LAST_BILL (-) USING $$,$$$.99-,
  COL 112, LOANS.LATE_CHARGE (-) USING $$$.99,
SKIP
!
END       ! JBI1-PRINT-CURRENT-LOAN
!
END-PROCEDURE


DELETE JBI1_PRINT_JOURNAL_ENTRIES;
DEFINE PROCEDURE JBI1_PRINT_JOURNAL_ENTRIES
!
!  ************************************************************
!  * CDD$TOP.ADMIN.LOAN JBI1-PRINT-JOURNAL-ENTRIES
!  *
!  * Called By: JBI
!  *
!  * print all Journal entries for the current loan
!  ************************************************************
!
```

```
BEGIN
!
FOR JOURNAL WITH KEY = TMP_KEY BEGIN
  IF (JOURNAL.TYPE | JOURNAL.RATE = LOANS.TYPE | LOANS.RATE) AND
  (ENTRY_CODE = "ADD", "DEL", "PST", "MOD", "INT") BEGIN
    PRINT COL 1, JOURNAL.SESSION (-),
      COL 6, JOURNAL.ENTRY_CODE (-),
      COL 12, JOURNAL.ENTRY_DATE (-) USING MMM_DD,
      COL 19, JOURNAL.USER (-),
      COL 24, JOURNAL.CLASS | ID (-) USING X(7),
      COL 32, JOURNAL.STATUS (-),
      COL 35, JOURNAL.TYPE (-),
      COL 38, JOURNAL.RATE (-),
      COL 51, JOURNAL.ORIGINAL (-) USING $$,$$$.99-,
      COL 63, JOURNAL.BAL_PRIN (-) USING $$,$$$.99-,
      COL 75, JOURNAL.PRINCIPAL_DUE (-) USING $$,$$$.99-,
      COL 87, JOURNAL.INTEREST_DUE (-) USING $$,$$$.99-,
      COL 100, JOURNAL.PAID_SINCE_LAST_BILL (-) USING $$,$$$.99-,
      COL 112, JOURNAL.LATE_CHARGE (-) USING $$$.99
    END             ! IF (...
  END               ! FOR JOURNAL ...
!
END                 ! JBI1-PRINT-JOURNAL-ENTRIES
!
END-PROCEDURE


DELETE JBI1_PRINT_ORIGINAL_LOAN;
DEFINE PROCEDURE JBI1_PRINT_ORIGINAL_LOAN
!
!  ******************************************************
!  * CDD$TOP.ADMIN.LOAN JBI1-PRINT-ORIGINAL-LOAN
!  *
!  * Called By: JBI
!  *
!  * print a loan record from the old loan domain ONLY IF
!  * if is the same TYPE and RATE
!  ******************************************************
BEGIN
!
FOR OLDLOANS WITH KEY = TMP_KEY BEGIN
  IF OLDLOANS.TYPE | OLDLOANS.RATE = LOANS.TYPE | LOANS.RATE THEN BEGIN
    PRINT COL 32, OLDLOANS.STATUS (-),
      COL 35, OLDLOANS.TYPE (-),
      COL 38, OLDLOANS.RATE (-),
      COL 51, OLDLOANS.ORIGINAL (-) USING $$,$$$.99-,
      COL 63, OLDLOANS.BAL_PRIN (-) USING $$,$$$.99-,
      COL 75, OLDLOANS.PRINCIPAL_DUE (-) USING $$,$$$.99-,
      COL 87, OLDLOANS.INTEREST_DUE (-) USING $$,$$$.99-,
      COL 100, OLDLOANS.PAID_SINCE_LAST_BILL (-) USING $$,$$$.99-,
      COL 112, OLDLOANS.LATE_CHARGE (-) USING $$$.99
    END                     ! IF OLDLOANS.TYPE ...
  END                       ! FOR OLDLOANS WITH KEY ...
END                         ! JBI1-PRINT-ORIGINAL-LOAN
END-PROCEDURE
```

EXAMPLE 2

```
DELETE SHR_DISPLAY_DATA;
DEFINE PROCEDURE SHR_DISPLAY_DATA
!
!  ******************************************************
!  * CDD$TOP.ADMIN.LOAN SHR-DISPLAY-DATA
!  *
!  * Called By: DISP,
!  *            MNT1-MAINT,
!  *            PST1-POST
!  *
!  *
!  * Display the biographic and loan data for a select ID
!  * (the calling proc must supply a current BIO record)
!  ******************************************************
!
BEGIN
!
IF TMP_DEBUG_FLG = "DEBUG ON" DISPLAY "ENTERING PROC SHR-DISPLAY-DATA"
!
PRINT COL 33, "STUDENT LOAN RECORD (CURRENT FILE)"
:SHR_DISPLAY_BIO_DATA
!
!  ****************************************************************
!  * set up 'loan data' header                                    *
!  ****************************************************************
!
PRINT SKIP,
  COL 1, ":",
  COL 2, "-------------------------------------------------------",
  COL 61, "--------------------",
  COL 80, ":"
!
!  ****************************************************************
!  * display variable number of loan records                      *
!  ****************************************************************
!
:SHR_DISPLAY_LOAN_DATA
!
PRINT     COL 1, ":",
  COL 2, "-------------------------------------------------------",
  COL 61, "--------------------",
  COL 80, ":"
!
IF TMP_DEBUG_FLG = "DEBUG ON" DISPLAY "LEAVING PROC SHR-DISPLAY-DATA"
!
END       ! SHR-DISPLAY-DATA
!
END-PROCEDURE
```

```
DELETE SHR_DISPLAY_BIO_DATA;
DEFINE PROCEDURE SHR_DISPLAY_BIO_DATA
!
!  ********************************************************
!  * CDD$TOP.ADMIN.LOAN SHR-DISPLAY-BIO-DATA              *
!  *                                                      *
!  * Called By: SHR-DISPLAY-DATA,                         *
!  *            MNT2-MOD-BIO-REC                          *
!  *            MNT2_ADD_BIO_REC                          *
!  *                                                      *
!  * display biographic data for selected loan            *
!  ********************************************************
BEGIN
!
PRINT     COL 1, ":",
  COL 2, "-------------------------------------------------------------",
  COL 61, "--------------------",
  COL 80, ":",
SKIP,
  COL 1, ":",
  COL 3, "CL:" ||| CLASS,
  COL 21, "NA:" ||| NAME (-),
  COL 62, "SD:",
  COL 65, SEPARATION_DATE (-) USING DD_MMM_YYYY,
  COL 80, ":",
SKIP,
  COL 1, ":",
  COL 3, "ID:" ||| ID (-),
  COL 21, "S1:" ||| STREET1 (-) USING X(35),
  COL 62, "GE:",
  COL 65, GRACE_END_DATE (-) USING DD_MMM_YYYY,
  COL 80, ":",
SKIP,
  COL 1, ":",
  COL 21, "S2:" ||| STREET2 (-) USING X(35),
  COL 62, "DS:",
  COL 65, DEFER_START_DATE (-) USING DD_MMM_YYYY,
  COL 80, ":",
SKIP,
  COL 1, ":",
  COL 3, "HP:" ||| HOME_PHONE (-),
  COL 21, "CI:" ||| CITY | " ST:" | STATE | " ZP:" | ZIP USING X(40),
  COL 62, "DE:",
  COL 65,  DEFER_END_DATE (-) USING DD_MMM_YYYY,
  COL 80, ":",
SKIP,
  COL 1, ":",
  COL 3, "WP:" ||| WORK_PHONE (-),
  COL 21, "FR:" ||| FOREIGN (-),
  COL 62, "PD:",
  COL 65,  POST_DEFER_END_DATE (-) USING DD_MMM_YYYY,
  COL 80, ":",
SKIP,
```

```
  COL 1, ":",
  COL 2, "-------------------------------------------------------------",
  COL 61, "--------------------",
  COL 80, ":"
!
END       ! SHR-DISPLAY-BIO-DATA
!
END-PROCEDURE


DELETE SHR_DISPLAY_LOAN_DATA;
DEFINE PROCEDURE SHR_DISPLAY_LOAN_DATA
!
!  ****************************************************
!  * CDD$TOP.ADMIN.LOAN SHR-DISPLAY-LOAN-DATA
!  *
!  * Called By: MNT2-ADD_LOAN_REC,
!  *            MNT2-MOD_LOAN_REC,
!  *            SHR-DISPLAY-DATA
!  *
!  * 1. compute ARREARS
!  * 2. print loan data
!  ****************************************************
BEGIN
!
DECLARE QUARTERLY_ARREARS      PIC 9(2).
DECLARE MONTHLY_ARREARS        PIC 9(2).
!
FOR LOANS WITH KEY = TMP_KEY BEGIN
!
  :SHR_COMPUTE_ARREARS
!
  PRINT COL 1, ":",
        COL 3, "TYPE:",
        COL 11, TYPE (-),
        COL 21, "ACTIVE:",
        COL 28, ACTIVATION_DATE (-) USING DD_MMM_YY,
        COL 43, "ORIG:",
        COL 50, "$" | ORIGINAL (-),
        COL 65, "TC:",
        COL 71, TEACHER_CANCELLATION (-),
        COL 80, ":",
    SKIP,
        COL 1, ":",
        COL 3, "STATUS:",
        COL 11, STATUS (-),
        COL 21, "FIRST:" ,
        COL 28, DATE_FIRST_PAID (-) USING DD_MMM_YY,
        COL 43, "BAL P: $" | BAL_PRIN,
        COL 65, "MBILL:" | MONTHLY_BILL,
        COL 80, ":",
    SKIP,
```

```
        COL 1, ":",
        COL 3, "RATE:",
        COL 11, RATE | "-" | GRACE_PERIOD,
        COL 21, "LAST:",
        COL 28, DATE_LAST_PAID (-) USING DD_MMM_YY,
        COL 43, "P DUE: $" | PRINCIPAL_DUE (-),
        COL 65, "LATE: $" | LATE_CHARGE (-),
        COL 80, ":",
    SKIP,
        COL 1, ":",
        COL 3, "ARREARS:" | QUARTERLY_ARREARS |"Q " | MONTHLY_ARREARS | "M",
        COL 21, "LAST:",
        COL 28, "$" | PAID_SINCE_LAST_BILL (-),
        COL 43, "I DUE: $" | INTEREST_DUE (-),
        COL 80, ":",
    SKIP,
        COL 1, ":",
        COL 80, ":"
  END                ! FOR LOANS ...
END                  ! SHR-DISPLAY-LOAN-DATA
!
END-PROCEDURE
```

Hints and Kinks

---

* Dick Azzi has the following hint: A lot of my users have a complaint that their entry people often enter wrong dates in fields. This is especially prevalent in January of each year when people haven't gotten used to using the new year. Since all of our entry is done within a week or so of the date to be entered, I came up with a validation check that can be placed in a record definition.

```
05  ENTERED_DATE USAGE DATE
      VALID IF "TODAY" LE ENTERED_DATE + 15.
```

This validation will not allow the entering of a date that is greater than 15 days prior to the current date. If you wish to be able to go back 30 days, just change the 15 to a 30. This little validation sure saves a lot of entry problems.

An Open, Menu-Driven, Graphic Report Generator using DTR and VAX-BASIC

S. Begelman, Digital Equipment-France, Rungis, France

One common difficulty with DATATRIEVE, when considered in an "Information Center" context, is that it is highly unsuited as an intensive query-tool for users who do not wish to learn how to program DATATRIEVE, yet need a users-friendly, sophisticated interface to their own data.

With this in mind, I had the occasion to show a prospective customer that it is extremely easy to go beyond standard DATATRIEVE, and offer the user a tool to generate reports, queries, and graphics. The purpose of this article is to show how DATATRIEVE may be interfaced with any high-level language -- in this case VAX-11/BASIC -- to create an open, menu-driven graphic report-generator. The three most typical plots are included in my example, and it is very simple to add other ones if you so wish. French-speaking readers will be delighted. [Some rough translations have been provided by the Newsletter Editor so that non French speaking readers can follow the logic of the DATATRIEVE and BASIC code. These translations appear in comment lines within square brackets.]

You will find below: my DATATRIEVE DTR$STARTUP file, as well as the BASIC source code, which illustrates how to connect to the default CDD, verify the existence of domains, make selections, choose plots, and print reports to a graphics terminal. A word to the wise: unlike a VT125, do not declare the type of device (SET TERM/DEV=VT125), when you are using a VT240/41, PRO-350 or Rainbow; trouble could result for this... [Declare the device to be a VT100 rather than a VT125]. The program verifies, of course, the status of each call between BASIC and DATATRIEVE for success or failure, handles errors, reprompts, et cetera.

A complete application would allow the operator to "JOIN" multiple domains or Rdb relations. This is left as a exercise for those who are interested in completing the logic of this application.

The compile options command file, as well as the "INCLUDE" file are included in Version 3.0 release of DATATRIEVE, and should be copied to your own account before compilation. I am using: VMS V3.7, TDMS V2, CDD V2.0, BASIC V2.2, and, naturally, DATATRIEVE version 3.0.

In another context, I was confronted with the problem of showing how to do controlled, field-by-field data input using Rdb, DATATRIEVE, and TDMS. This is done in the following way: the fields <DEGREE_FIELD> and <DEGREE> are defined as display only in the TDMS form definition state; the input-output information is then masked with PUTS and GETS, as you see from the DATATRIEVE procedure that I have included. The magic is handled by TDMS: the Rdb relations are in UPDATE or MODIFY mode, yet TDMS will not let you position the cursor in these fields to do updating ....

It is important to notice that the records must be "streamed" with a FOR statement: in other words, FINDs and SELECTs either work very poorly or do not work at all. The FIND/SELECT syntax works very poorly - sometimes not at all - with Rdb.

I hope that these two simple examples will allow other VMS/DATATRIEVE users to get to the bottom of two know DATATRIEVE issues.

[Some of the code in BASIC program and in the DATATRIEVE procedure below has been slightly modified to decrease the number of lines of material.]

```
!DTR$STARTUP FILE
SET NOVERIFY
PRINT " "
PRINT "  Demarrage d'une session avec DATATRIEVE V3"
!         [Starting a session with DATATRIEVE]
PRINT " "
PRINT "Le Dictionnaire contient les elements suivants ..... "
!      [The dictionary contains the following elements ...]
SHOW ALL
SET PLOTS CDD$TOP.DTR$LIB.PLOTS
SET DICTIONARY DTR$DEFAULT$DIC

! A BASIC program to drive the graphics
100     ! Generateur de Graphiques avec <DATATRIEVE> V3
        !
        ! S. Begelman, Digital-France, 10-Dec-84, Rungis
        !
110     %INCLUDE "DTR$LIBRARY:DAB.BAS"
        OPEN "TT:" AS FILE 1%
        ! Declarer l'initialisation fonction.
        ! Declarer le status success/echec. [success or failure]
        EXTERNAL INTEGER FUNCTION DTR$INIT
        EXTERNAL LONG CONSTANT SS$_NORMAL
        ! declare the options for the DTR$DTR call.
        DECLARE INTEGER DTR_OPTIONS
        DTR_OPTIONS = DTR$M_OPT_CMD + DTR$M_OPT_LINE
        DECLARE INTEGER RET_STATUS
120     PRINT ESC + "[2J"
        PRINT ESC+"[10;1H"+ESC+"#3  DTR :  Generateur d'Etats Graphiques "
        PRINT ESC+"[11;1H"+ESC+"#4  DTR :  Generateur d'Etats Graphiques "
        PRINT ESC+"[12;1H"+ESC+"#3  ------------------------------------"
        PRINT ESC+"[13;1H"+ESC+"#4  ------------------------------------"
        !                                [Graphics Report Generator]
        ! pause boucle     [pause loop]
        !
        FOR ZZ = 1 TO 50
        ZZ = ZZ + 10
        NEXT ZZ
130     RET_STATUS = DTR$INIT (DAB BY REF, 100% BY REF, MSG_BUFF, &
            AUX_BUFF, DTR$K_SEMI_COLON_OPT BY REF)
        ! Verify that DATATRIEVE was initialized successfully.
        IF RET_STATUS <> SS$_NORMAL THEN
            PRINT "DATATRIEVE initialisation A ECHOUE."
        !         [DATATRIEVE initialization has failed.]
            GOTO 290
140     CALL DTR$COMMAND (DAB BY REF,"SET PLOTS CDD$TOP.DTR$LIB.PLOTS")
        CALL DTR$DTR BY REF (DAB, DTR$M_OPT_CMD)
        ! Clear the screen.
        ! Show the user the domains available.
150     CALL LIB$ERASE_PAGE BY REF ( 1% , 1% )
        PRINT ESC + "[10;10H Voici les domaines dans votre C.D.D.:"
        !                    [Here are the domains in your CDD.]
        PRINT ESC + "[15;1H  "
```

```
160     CALL DTR$COMMAND ( DAB BY REF, "SHOW DOMAINS" )
        ! Use DTR$DTR to display the results of SHOW DOMAINS.
        CALL DTR$DTR BY REF (DAB, DTR$M_OPT_CMD)
        ! Prompt the user for the domain name and ready the domain.
170     INPUT "Quel domaine souhaitez-vous analyser"; DOM$
        !     [Which domain do you wish to analyze?]
        CALL DTR$COMMAND (DAB BY REF, "READY !CMD", DOM$)
        ! check for errors.
        IF DAB$L_CONDITION <> DTR$_SUCCESS THEN
            CALL LIB$ERASE_PAGE BY REF ( 1% , 1% )
            PRINT ESC + "[12;12H Veuillez entrer un domaine valide ..."
        !                        [You have to enter a valid domain.]
            PRINT "        "
            GOTO 160
180     NUE:       ![Me thinks Begelman is having fun with us non-French  ]
                   ![speaking persons.  In my old French dictionary "Nue" ]
                   !translates to bare, exposed, naked, nude, plain,      ]
                   ![denuded, destitute, or cloud.  Did I miss the joke?  ]
        PRINT " "
        PRINT ESC+"[1m";"   La touche [NO SCROLL] arretera l'affichage "
        !                  [The NO SCROOL key will stop the display]
        PRINT ESC+"[0m"
        PRINT " "
        PRINT ESC+"[23;20H Frapper une touche + [R/C] pour continuer "
        !                 [Press a key and carriage return to continue]
        INPUT " ";CE$
        CEL = LEN(CE$)
        IF CEL > 1 THEN GOTO NUE
190     CALL DTR$COMMAND (DAB BY REF, "SHOW FIELDS !CMD",DOM$)
        CALL DTR$DTR BY REF (DAB, DTR$M_OPT_CMD)
        !SELECTION RSE OU PAS?            [RSE or not?]
200     REPRISE:                        ! [resumption or repetition]
        PRINT " "
        PRINT "Voulez-vous faire une selection dans le domaine "
        !      [Do you want to make a selection in the domain?]
        LINPUT "OUI/NON, S.V.P"; CHOIX_CONT$
        !      [[Enter] yes or no, please]
        SELECT CHOIX_CONT$
            CASE = "OUI"
               GOTO 210
            CASE = "NON"
               GOTO 220
            CASE ELSE
               GOTO REPRISE
            END SELECT
210     LINPUT "Quelle selection"; SEL$  ! [what selection]
        CALL DTR$COMMAND (DAB BY REF, "FIND !CMD WITH !CMD", DOM$, SEL$)
        ! Use DTR$DTR to handle messages and return at DTR$K_STL_CMD
        IF DAB$W_STATE = DTR$K_STL_CMD THEN
            CALL LIB$ERASE_PAGE BY REF ( 1% , 1% )
            PRINT " ", " ", " "
            PRINT ESC + "[12;12H Champs inexistants; Recommencez ... "
        !                       [Field doesn't exist, begin again]
            PRINT " "
            GOTO 200
```

```
        ELSE
        CALL DTR$DTR BY REF (DAB, DTR$M_OPT_CMD)
        CALL DTR$COMMAND (DAB BY REF, "SELECT")
        CALL DTR$DTR BY REF (DAB, DTR$M_OPT_CMD)
        END IF
    !
220 CALL DTR$COMMAND(DAB BY REF, "FIND !CMD",DOM$)
    CALL DTR$DTR BY REF (DAB, DTR$M_OPT_CMD)
    CALL DTR$COMMAND (DAB BY REF, "SELECT")
    CALL DTR$DTR BY REF (DAB, DTR$M_OPT_CMD)
    !
230 CALL LIB$ERASE_PAGE BY REF ( 1% , 1% )
    PRINT " ------------------------------------------------ "
    PRINT "     Quel type de graphique souhaitez-vous faire?   "
    !          [What type of graph do you wish to make?]
    PRINT " "
    PRINT " 1.        Histogramme              [Bar-Chart]"
    PRINT " 2.        Camembert                [Value_Pie]"
    PRINT " 3.        Nuage de points simples  [Scattergraph]"
    PRINT " "
    PRINT " ------------------------------------------------ "

240 KUNG_FU:
    INPUT "Votre Choix ( 1, 2, ou 3)";D      ! [[enter] your choice]
    SELECT D
        CASE = 1
           GOTO 250
        CASE = 2
           GOTO 260
        CASE = 3
           GOTO 270
        CASE ELSE
           GOTO 230
        END SELECT
    !
250 HISTOGRAMME:
    PRINT "1er Champ=(Critère de trie), 2e Champ = (Value representee)"
    !   [First field = sort criterion, Second field = represented value]
    PRINT " "
    LINPUT "PLOTTER Quels Champs";C$  ! [Plot which fields]
    CALL DTR$COMMAND(DAB BY REF,"PLOT BAR ALL !CMD",C$)
    CALL DTR$DTR BY REF (DAB, DTR$M_OPT_CMD)
    GOTO 280
260 CAMEMBERT:
    PRINT "1er Champ=(Critère de regroupement), ";
    PRINT "2e Champ =(Value representee)"
    ! [First field=grouping criterion, Second field = represented value]
    LINPUT "PLOTTER Quels Champs";C1$
    CALL DTR$COMMAND(DAB BY REF,"PLOT VALUE_PIE ALL !CMD",C1$)
    CALL DTR$DTR BY REF(DAB, DTR$M_OPT_CMD)
    GOTO 280
```

```
270     NUAGE_DE_POINTS:
        PRINT "1er Champ = AXE-X , 2e Champ = AXE-Y"
        !     [First Field = X Axis, Second Field = Y Axis]
        PRINT " "
        LINPUT "PLOTTER quels champs";C2$
        CALL DTR$COMMAND(DAB BY REF,"PLOT X_Y ALL !CMD",C2$)
        CALL DTR$DTR BY REF (DAB, DTR$M_OPT_CMD)
280     CALL DTR$FINISH (DAB BY REF)
290     END


!
!S. Begelman   Digital-France  4-Jan-85 demonstration Rdb-TDMS-DTR
!
declare encore                          pic x.    ! [encore -> again]
encore = 1
declare choix_id                        pic x(5). ! [choix_id -> chosen id]
declare reponse                         pic x(3). ! [reponse  -> response]
!
declare errcont                         pic x.    ! [errcont  -> error continue]
declare cont                            pic xxx.  ! [cont     -> continue]
!
ready personnel using employees         modify
ready personnel using degrees           modify
ready personnel using colleges          modify
!
while encore = 1 begin
!
!  boncle majeur: ----------------------------------------------[ major loop ]
!
   reponse = "OUI"                                          ! [OUI -> yes]
   while reponse = "OUI" begin
     reponse = "NON"                                        ! [NON -> no]
     display_form rdbform2 in rdb retrieve using begin
       choix_id = get_form choix_id
       reponse  = get_form reponse
       end
!
     if reponse eq "OUI" then begin
       encore = 1
       end
!
     if reponse eq "NON" then begin
       encore = 0
       end
     end
!
   for e in employees cross d in degrees over employee_id cross -
     c in colleges over college_code with e.employee_id = choix_id begin
```

```
    modify using display_form rbdform1 in rbd using begin
      put_form social_security = social_security
      put_form last_name       = last_name
      put_form degree_field    = degree_field
      put_form college_name    = college_name
      end retrieve using begin
      degree_field             = get_form degree_field
      degree                   = get_form degree
      end
    end
  commit
!
!  continuer or pas?                    ! [continue or not?]
!
  errcont = 1
  while errcont = 1 begin
    errcont = 0
    cont = *."Voulez-vous continuer ? (OUI ou NON)"
    !        [Do you wish to continue (yes or no)?]
    cont = fn$upcase(cont)
    if cont ne "OUI" and cont ne "NON" then begin
      print " "
      print "Repondez par OUI ou par NON, s'il vous plait."
      !     [Respond with yes or with no, please.]
      print " "
      end
    if cont eq "OUI" then begin
      encore = 1
      end
    if cont eq "NON" then begin
      encore = 0
      end
!
    end
  end
!-------------------------------------------------------------
print " "
print "             Saisie/consultation terminee"
!
```

A Useful DATATRIEVE Date Function

John Briggs, SPACECOM, Gaithersburg, MD

We often use DATATRIEVE to access files created and maintained from non-DATATRIEVE applications. One problem we had often encountered was the incompatibility of date formats used by these applications. Searches that involved date comparisons were incredibly difficult. In addition, the output format of the date fields could not be changed.

Our solution to the problem was to create a user defined function that converts text string to a standard DATATRIEVE date/time. We call this conversion function "DATE". It works like this:

```
DTR> PRINT DATE("TODAY")
   DATE
 6-Nov-1984
DTR>
```

When we have to use a file that has non-standard date fields, we use a COMPUTED BY field that uses the DATE function to translate the date to the DATATRIEVE format. Then we can do date comparisons and output formatting in the usual way with the computed field.

The magical part of all this is that we didn't have to write any conversion routine. We let DATATRIEVE do it!

When you add a user defined function to DATATRIEVE you put a description of the function into DTRFND.MAR [in DTR$LIBRARY]. This description specifies the type of the arguments and the type of the result. When DATATRIEVE invokes the function it automatically converts input arguments to the type specified in the function description.

The following is the function description we added to DTRFND.MAR:

```
$DTR$FUN_DEF DATE, TEST_TO_DATE, 2
  $DTR$FUN_OUT_ARG TYPE = FUN$K_STATUS
  $DTR$FUN_IN_ARG  TYPE = FUN$K_REF, DTYPE = DSC$K_DTYPE_ADT, ORDER = 1
  $DTR$FUN_IN_ARG  TYPE = FUN$K_REF, DTYPE = DSC$K_DTYPE_ADT, OUT_PUT = TRUE
$DTR$FUN_END_DEF
```

And this is the FORTRAN routine that is called.

```
      INTEGER FUNCTION TEST_TO_DATE ( DATE_IN, DATE_OUT )
      INTEGER * 4    DATE_IN ( 2 ), DATE_OUT ( 2 )
      DATE_OUT ( 1 ) = DATE_IN ( 1 )
      DATE_OUT ( 2 ) = DATE_IN ( 2 )
      TEXT_TO_DATE = 1
      RETURN
      END
```

Details on how to add functions to DATATRIEVE are in Chapter 6 of the DATATRIEVE Guide to Programming and Customizing (Version 3).

DATATRIEVE Masters

The following list of names identifies people who have agreed to provide assistance to other DATATRIEVE users.

Operating System	Phone	Name and Address
RSX & P/OS	(212) 775-3567	Bart Z. Lederman Bankers Trust Co., 28th Floor One Bankers Trust Plaza New York, NY 10006
RSX & VMS	(617) 839-4411 X5480	Joe Kelly Waymon Gordon Worchester Road North Grafton, MA 01536
IAS & VMS	(509) 376-2227	Chuck Watson Battelle Northwest Richland, WA 99352
VMS	(202) 426-1345	Larry Jasmann 10067 Marshall Pond Rd. Burke, VA 20015
VMS	(312) 982-7430	James Swanger G. D. Searle & Co. 4901 Searle Parkway Skokie, IL 60077
VMS	(206) 396-2501	Darrell Eade Naval Undersea Water E. S. Hayport, WA 98345
VMS	(302) 366-4610	Chris Wool E. I. DuPont Engineering Dept., Louviers Bldg. Wilmington, DE 19898
VMS	(602) 244-04316	Dick Azzi Motorola SPS Box 2953 MS: P116 Phoenix, AZ 85062

DATATRIEVE SIG Officers and Steering Committee

SIG Chairman	Lawrence Jasmann U. S. Coast Guard 10067 Marshall Pond Rd. Burke, VA 20015 202-426-1345
Symposium Committee Representative	Chris Wool E. I. duPont Eng. Dept., Louviers Bldg. Wilmington, DE 19898 302-366-4610
Newsletter Editor & Communications Committee Representative (Acting)	Joe H. Gallagher Cleveland Clinic Foundation 9500 Euclid Avenue Cleveland, OH 44106 216-444-2551
Library Committee Representative	Katherine M. Tamer University of Texas Med.School 6431 Fannin RM 5.020 Houston, TX 77030 713-792-5566
Symposium Session Note Editor	Elaine McWilliams Boeing Company P.O Box 24346, M/S 9c-55 Seattle, WA 98124 206-773-0102
Pre-Symposium Seminar Coordinator	Dana J. Schwartz US Dept. of Defense 9800 Savage Road Attn. R822 Fort George G. Meade, MD 20755 301-859-6277
Artist and Media Specialist	Bart Z. Lederman Bankers Trust Co., 38th Floor P.O.Box 318 Church St. Station New York, New York 10015 212-775-3567

Promotions & Campground Coordinator	Robert R. Lott E. I. Dupont N6514, 1007 Market St. Wilmington, DE 19898 302-774-2752
Large Systems (DEC 10/20) Liason	Lynn Duncan Oak Ridge National Lab P.O. Box Y, Bldg 4500N, MS H32 Oak Ridge, TN 37831 615-576-5992
Data Management System SIG Liaison	Keith Hare P.O. Box 463 Granville, Ohio 43023 614-587-0051 (Home)
Finance Committee	Joan Hilton Biogen Research Corp. 14 Cambridge Center Cambridge, MA 02142 617-864-8900 x392
Hardware Coordinator	Alex L. Lamb Jr. U. S. Army 16 Wake Road Eatontown, NJ 07724 201-532-3318/1843
Hardware Specialist, Black Book	Bert Roseberry U.S. Coast Guard District (DT) 500 Camp Street New Orleans, LA 70130 504-589-4934
Steering Committee Member	Harold T. Glaser J. Montgomery Cons. Engrs. P.O. Box 7009 Pasadena, CA 911097009 818-796-9141
Digital Counterpart	Chuck Duncan Digital Equip Corp ZKO2-2/M28 110 Spit Brook Road Nashua, NH 030622642

From the Editor's Pen

Joe H. Gallagher, Cleveland Clinic Foundation, Cleveland, OH

We were pleased to publish three original articles in this issue of the Wombat Examiner. The activities of the semi-annual symposia generates sufficient material for two of the four issue per year; however, it is from you, the DATATRIEVE user community, that the remaining material must come. DATATRIEVE with its wide range of machine implementations and wide range of applications can not help but create interesting, unusual, and unique examples, problems, and computer solutions. It is your responsibility and duty to share your triumphs and troubles with your fellow DATATRIEVE users. You will, of course, notice that one of the articles is in French/English/DATATRIEVE/BASIC. Since we, as computer professionals, are already multi-lingual, we would be please to publish articles written by our European colleagues in French, German, or Spanish as long as the English translations are provided. We will publish the bi-lingual articles in a side-by-side presentation.

This is the time of the year when you should be thinking about renewing your subscription to DECUS newsletters. The subscription service for many subscribers starts at the beginning of July. Subscribers will be sent renewal notices, renewal forms will be available at the New Orleans Symposium, and in the next issue of the Wombat Examiner. There are many fine advertiser supported magazines such as **Hardcopy**, **The DEC Professional**, **The VAX Professional**, **Digital Review**, and others which provide users with very valuable information at a small or no cost. However, these magazines will never provide the detailed technical information that is available within the wide range of DECUS SIG newsletters. Support the activities of your Special Interest Group and DECUS; renew your subscription when it comes due.

At a meeting of the SIG Publications Committee of the Communications Committee which was held March 11-13, 1985, the newsletter offerings for 1986 were carefully and thoughtfully reviewed. Very, very careful considerations was given to combining all DECUS newsletters into one big newsletter which would be published monthly. While there are many good reasons why such a combined publications should be attempted, the cost of such a publication was variously estimated at $60 to $100 per year. Since it is unlikely that some support mechanism (such as DEC subsidy or advertising) could be worked out within the present framework of DECUS, it was felt that a user subscription fee of at least $60 would not be acceptable to DECUS members with small systems (RT, PC, micro-RSX, APL, etc). It was therefore decided to keep the newsletters as separate publications. However, at that meeting, there was some plans made to have several of the current newsletter publish with the Page Swapper (VAX SIG Newsletter). The new combined newsletter offerings will be announced at the New Orleans Symposium and in the subscription renewal notices. The Wombat Examiner will remain as a separate newsletter, at least through July of 1986. The information contained within the Wombat Examiner, however, is likely to be expanded in light of the Fourth Generation Language Proposal which Larry Jasmann, Chairman, DATATRIEVE SIG, has made. His proposal is the first article in this issue.

News Briefs

* The DATATREIVE SIG Steering committee has prepared a special item for the DECUS store at the New Orleans Symposium. The item is a "secret" and no disclosure will be made until the items goes on sale in the DECUS store on Monday, May 27, 1985, when the DECUS Exhibit Hall opens. Those attending the New Orleans Symposium may wish to bring along a few extra bucks. Many may wish to acquire one or more of these limited addition items.

* Glenn Everhart has sent a new version of PortaCalc to the DECUS library which contains a DATATRIEVE interface. Glenn says that it seems to work pretty well and could be used in shops that don't have FMS where it could let the PortaCalc spreadsheets be used for data entry forms. It will, however, also work with FMS if you have forms attached to domains ... the necessary enabling bit is set. Glenn doesn't I don't know how long it will take to become available through to DECUS library. Those interested in these new capabilities should check with the DECUS Library.

* Chuck Duncan reports that the first phases for VAX DATATRIEVE Versions 3.2 and 3.3 are about done. Versions 3.2 and 3.3 will both make a number of bug fixes. In addition, support is planned for RMS Quad word keys (for dates), support for Rdb varying text strings, full cluster support, real support for VT240/241 terminals, and an interface to the VMS 4.0 SORT routine. It is anticipated that Version 3.2 will ship late in the summer. In addition, installation of DATATRIEVE will not require the presence of DECnet (this is a real issue right now on the uVAX!). Support for Rdb segmented strings and more bugs fixes are planned for VAX DATATRIEVE Version 3.3 which is expected to ship in late 1985. [Editor's note: The information related here is NOT a commitment by Digital. It is merely an attempt to let users know what they might expect in the future with regard to the VAX DATATRIEVE product.]

DATATRIEVE User Command File

Joe H. Gallagher, Cleveland Clinic Foundation, Cleveland, OH

After you have read this double issue of the Wombat Examiner, please execute the following command file on your machine.

```
ready datatrieve_action WRITE!
for all_datatrieve_users in datatrieve_action begin
  :think_about_an_interesting_datatrieve_problem
  :think_about_datatrieve_call_interface
  :think_about_interaction_with_programming_languages
  if machine eq "VAX" then begin
    :think_about_interaction_with_CDD
    if ((fn$status("FMS") eq "AVAILABLE") or
      (fn$status("TDMS") eq "AVAILABLE")) then
      :think_about_interaction_with_forms_tools
    end else begin
    :think_about_conserving_pool_space
    end
  if ((problem eq "EXISTS") or (solution eq "EXISTS") or
          (hint eq "EXISTS")) then
    :write_article
  modify article with topic containing "DATATRIEVE" using begin
    style = "Standard English"
    content = "Information"
    end
  if ((fn$status("KERMIT") eq "AVAILABLE") or
      (fn$status("XMODEM") eq "AVAILABLE") or
      (fn$status("MAGTAPE") eq "AVAILABLE")) then begin
    :send_article_electronically
    end else begin
      :send_article_via_hardcopy
      end
    end
  store treasures using begin
    respect = "Much"
    praise = "Undying"
    thanks = "Well deserved"
    end
  end
```

Printed in the U.S.A.