



CCAPRINT

A Newsletter Excerpt for System 1032 Users

March 2008

USE OF AND ACCESS TO PRODUCTS AND FEATURES ARE IN ACCORDANCE WITH THE TERMS AND CONDITIONS OF THE USER'S SOFTWARE LICENSE. THE PRESENTATION OF MATERIAL HEREIN DOES NOT, IN ANY MANNER, MODIFY SUCH TERMS AND CONDITIONS.

DSDELIM – Extracting Data in Delimited Text Format

By Tym Stegner

A customer had numerous datasets from which he wanted to extract all the data, so he could analyze it in Microsoft Excel. What was needed was a simple approach to performing this extraction from any particular dataset. Thus was born another useful utility, the DSDELIM tool.

Design Considerations

The DSDELIM tool is a utility (or program) that itself creates a program when it is run. In the case of DSDELIM, a dataset-specific DMC file is created that can extract records from a dataset into a vertical-bar (|) delimited text file. The generated program file has the name of the dataset with the DELIMIT file type.

There are two major reasons the DSDELIM tool produces a DMC, instead of directly outputting the delimited text.

1. Firstly, processing becomes very costly when acquiring the value of an attribute, when the attribute name is not itself encoded into the program. To do so, DSDELIM would have to use EXECUTE statements or an API call to read in each attribute value itself. This would produce massive overhead for reading each record.
2. Secondly, the reason is one of simplicity. The DSDELIM tool is only 58 lines long, and other than some FORMAT statements, the code is easy to follow.

DSDELIM Design Details

In this simple case, our requirements are straightforward: output a text file delimited with vertical-bar characters. The heading line for the file should be descriptive, so the defined TITLE for each attribute is used. If the TITLE is not defined, the automatic default is the primary name for the attribute, so there should not be any null titles.

Non-text attributes are handled with the following consideration. Using the A format specifier in a FORMAT statement will automatically suppress trailing spaces in a text field, so in the case of any non-TEXT attributes, they are converted into TEXT. Therefore, the data type of each attribute must be tested so that the program knows when such conversion must take place.

Functionally, the program needed to pass through the attributes in the dataset two times: first time for the titles, second time for the attribute names in the WRITE command. This is because the titles and attribute names are written to the output file in separate places and cannot be written in the same pass. To circumvent two passes through the dataset, DSDELIM makes a single pass through the dataset and stores the necessary information in an array, keyed by the ordinal position of the attribute in the dataset.

The DSDELIM tool makes use of the PL1032 procedure that accesses the API SHOW command, so the S1032_HLI library must be available.

While loading the array, the title information is also written to the output file. This saves the step of storing the title information into the array. Once the array is filled, a second loop is used to output the item list for the WRITE command, including a test for the data type. Those items that are not already text are converted into text using the \$TEXT() function.

The DSDELIM Tool

```

!! DSDELIM - Create a dataset-specify DMC to extract active records
!! from that dataset into a text delimited format.
!!
open library S1032_HLI in S1032_TOOLS readonly
var ax,nx integer
var delimit text varying init "|" !change value to change delimiter
var txbuf text varying
var z array (100) of group of          !increase on -E-SUBRANGE error
      nm text varying
      tx logical init false
end_group

call show_int(nx, "NUMBER", $current_dataset,, "ATTRIBUTE")
init 1 $file($current_dataset&".DELIMIT")
write on 1 $current_dataset $current_dataset format(-
  'set ds ' a / -
  'find all' // -
  'init 8 ' a '.txt' / -
  'write on 8 format( -' )

for ax from 1 to nx do
  call show_text(z.nm(ax), "NAME",,, ax, "ATTRIBUTE")      !Atr Name

  call show_text(txbuf, "TITLE",,, ax, "ATTRIBUTE")        !title
  write on 1 txbuf delimit format (3x ''' a a ' ' $ -')

  call show_text(txbuf, "TYPE",,, ax, "ATTRIBUTE")          !data type
  if txbuf beg "Text" or txbuf beg "User" then !is a text data type?
    let z.tx(ax) = true
  end_if
end_for !attr scan

write on 1 format( -
  3x ''' ') / -
  'write on 8 fmt(" ")'// - !force end-of-line from $ fmt
  'for each record do' / -
  ' write on 8 -' )

for ax from 1 to nx do
  if z.tx(ax) then
    write on 1 z.nm(ax) format( 8t "#NAM'" a "' -" )
  else
    write on 1 z.nm(ax) format( 8t "$text(#NAM'" a "' ) -" )
  end_if
end_for

write on 1 nx delimit format( 4t "format( " i3 '(a "' a '") )' )
write on 1 $current_dataset format( -
  "end_for" / -
  "release 8" / -
  'write format("Created ' a '.TXT")' )
release 1
write $current_dataset format("Created DMC file " a ".DELIMIT")

```

Formatting FORMATS

The DSDELIM tool is primarily creating two specific WRITE commands with associated FORMAT statements.

- The first WRITE command creates the heading for the output file making use of the *prompt* (\$) format specifier to suppress the carriage return on each line. This use lets DSDELIM simply output one title per line in the loop.
- The third WRITE command does the heavy lifting of the generated DMC, because it contains the item list of attributes to be output.

For both WRITE commands, you must correctly envision the desired commands in the finished DMC to correctly specify the FORMAT statements in DSDELIM. The first WRITE command must include double quotes to surround the title text, as well as encompass the delimiter string. The third WRITE command must correctly include the required single quotes to encircle the attribute names as part of the #NAM specifier.

One of the tricks to writing these types of format statements is to know that single quotes can also be used to delimit text in a FORMAT statement. This is especially useful when you need to output text containing double quotes.

Generated Code

The following code is generated by the DSDELIM tool for the DIVISIONS dataset.

```

set ds DIVISIONS
find all

init 8 DIVISIONS.txt
write on 8 format( -
  "Division Name|" $ -
  "Div./ID|" $ -
  "City|" $ -
  "State//Province|" $ -
  "Country|" $ -
  "Division Budget|" $ -
  "Profits|" $ -
  " ")
write on 8 fmt(" ")

for each record do
  write on 8 -
    #NAM'DIVISION_NAME' -
    $text(#NAM'DIVISION_ID') -
    #NAM'CITY' -
    #NAM'STATE' -
    #NAM'COUNTRY' -
    $text(#NAM'DIVISION_BUDGET') -
    $text(#NAM'PROFITS') -
    format( 7(a "|" ) )
end_for
release 8
write format("Created DIVISIONS.TXT")

```

An output channel is initialized, followed by a WRITE command to create the heading line for the output data. A record loop performs another WRITE command on the selected records in the dataset. All non-text data is converted to text, thus making the FORMAT portion of the WRITE command very simple. The “#NAM’...” operator is used to ensure operation of the DMC file in the event that an attribute name happens to be a keyword, in which case an error would ensue.

Using the DSDELIM Tool

To use the DSDELIM tool, choose the dataset from which you need to extract data, and open it in a System 1032 session. The following example uses the DIVISIONS dataset located in the S1032_DEMO directory.

```

$ s1032
Computer Corporation of America System 1032 Version V9.81-1
Copyright 2002, Computer Corporation of America
1032> open ds divisions in s1032_demo read
Current dataset is now DIVISIONS
1032>
1032> use DSDELIM
Initializing channel 1
Created DMC file DIVISIONS.DELIMIT
1032>
1032> use DIVISIONS.DELIMIT
6 DIVISIONS records found
Initializing channel 8
Created DIVISIONS.TXT
1032> exit
$
$ type DIVISIONS.TXT
Division Name|Div./ID|City|State//Province|Country|Division Budget|Profits|
Eastern Entertainment| 1|New York City|New York|USA| 1000000.00| 200000.00|
California Classic Films| 2|Hollywood|California|USA| 1200000.00| 225000.00|
Autre Chose Cinema| 3|Montreal|Quebec|Canada| 1250000.00| 225000.00|
Cine Quebec| 4|Montreal|Quebec|Canada| 1230000.00| 240000.00|
Cinemexico| 5|Mexico City|Mexico|Mexico| 2750000.00| 300000.00|
Vanguard Films| 6|Boston|Massachusetts|USA| 3000000.00| 275000.00|
$

```

The DSDELIM tool operates on DIVISIONS, the current dataset, creating the DIVISIONS.DELIMIT command file. It locates the active records in the DIVISIONS dataset, and applies the WRITE commands described in [“Generated Code”](#) to output the headings, followed by the data from within the dataset.

If the extraction needs to be repeated later after dataset updates or upon another dataset having the same structure as the DIVISIONS dataset, the DSDELIM program need not be run again, because the DIVISIONS.DELIMIT program can be reused.

Note that DSDELIM does not work upon a dataset containing arrayed attributes.

Other Extraction Solutions

If you have this sort of extraction problem, but are uncertain that using the DSDELIM tool is the best solution for you, please have a look at previous articles on this topic, which have resolved the problem a bit differently.

- One option is to create a custom Record Descriptor incorporating per-field delimiters for each dataset, as mentioned in the CCAPrint article [“Synchronizing System 1032 Data with another DBMS”](#), (March, 2004); and as described in the CCAPrint article, [“ADD Command Generator: ADDGEN”](#) (January, 2007).
- In a similar fashion, a PRINT command could be used to create a more fixed format delimited file, rather than using RDs.

Keep in mind, however, that these solutions assume that you have some familiarity with System 1032's programming environment.

In Summary

The DSDELIM tool presents a programmatic method of extracting the data from a generic dataset into a delimited text file, via a two-step process.

Using DSDELIM is only a brick in the programs designed for extraction, however, one brick at a time is how things get built.

© 2008 Computer Corporation of America
200 West Street, 3rd Floor West, Waltham, MA 02451