DECISION SUPPORT
Data Integration & Reporting

Frequently Asked Questions

DARGAL and Miscellaneous

Question

Does DARGAL support the use of permanent directories and the propagation of security attributes?

Resolution

DARGAL supports the use of permanent directories if your site has established permanent directories and has set the system option PERMDIR. If your site administrator has set security so that non-privileged users can access the permanent directories, then non-privileged DARGAL users can create files within permanent directories without getting the "NOT ON YOUR USERCODE" messages. DARGAL supports permanent directories only on the local host, not remote hosts. Support for permanent directories was introduced with DARGAL Versions 59.170 and 60.110.

If the PROPAGATESECURITYTOFILES attribute of the permanent directory is set to PROPAGATE, then the security attributes associated with the permanent directory are propagated to the files created within the directory. This means that DARGAL recognizes those attributes and does not set default security attributes. However, if the user explicitly sets any security attributes, then those settings are used and any unset security attributes are set to the DARGAL defaults. Support for the PROPAGATESECURITYTOFILES attribute was introduced with DARGAL Versions 59.230 and 60.160.

Question

How do I create stream files in DARGAL?

Resolution

When you send output to a disk file, you can specify STREAM as a file attribute. It allows you to send output in one long stream within the file. You can use this attribute with all DARGAL commands that use the output menu. It does not apply to output generated by the REport command. When you set this attribute, FILESTRUCTURE is set to STREAM and MAXRECSIZE is set to 1. The following example uses the Write command and you are sending output from multiple Write commands to the same file:

```
WRITE OPEN:F=(SAMPLE)SAMPLES/TSET ON X(STREAM)
WRITE “ABC”
WRITE "DEF"
WRITE "GHI"
WRITE CLOSE
```

If you list the contents of the disk file, the three strings are appended in one long stream file:

```
ABCDEFGHI
```

Question

How do I convert a string that contains a combination of numeric and non-numeric characters to a numeric value in DARGAL?

Resolution

The VAL function is useful when you need to perform calculations involving employee ID numbers, account numbers, or other numbers that, for whatever reason, are stored as strings.

Following are some types of data that might contain both numeric and non-numeric characters:

- Account numbers that include some alphabetic "account type"
- Address fields that contain street numbers, route numbers, box numbers, or ZIP codes
- Telephone numbers
- Dates that contain embedded separators such as hyphens, slashes, or periods
- Automobile license plate numbers

If you need to access the numeric portion of a mixed alphanumeric field, you can use the FREE option of the VAL function. Here are some quick examples to demonstrate this.

From a DARGAL command prompt, enter the following WRITE command (the WRITE command is the quickest way to test or show the results of an expression):

```
W VAL("123")
```

DARGAL displays the number *123*.

Next, try the same thing with a string that contains some non-numeric characters:

```
W VAL("123ABC")
```

DARGAL displays the message "NON NUMERIC DATA IN VAL FUNCTION ARGUMENT."

Now, try the same thing, but add the word FREE as a second argument to the VAL function:

```
W VAL("123ABC", FREE)
```

Again, DARGAL displays the number *123*.

This will work equally well if the numeric portion follows the alpha characters (e.g., "ABC123").

The VAL function lets you convert numeric characters embedded within a string to numbers. But what if there is more than one set of numeric characters embedded? A date, for example, might be represented as "07/20/69." If you apply the VAL function to such a date, even with the FREE option, it will only return the first number (7).

To extract such embedded numbers, indicate, with the FREE option, which set of numeric characters you want to convert.

The following examples show how to extract the first, second, and third occurrences of numeric characters from the string "07/20/69."

```
VAL function                  Result
VAL("07/20/69", FREE1)             7
VAL("07/20/69", FREE2)            20
VAL("07/20/69", FREE3)            69
```

The preceding example shows, in effect, how to extract the numeric portions of a date represented as a combination of alphabetic and numeric characters. You can generalize this for other mixed-type items (street addresses, for example).

Continuing with date examples, let us point out that dates are often represented with hyphens (7-20-69) or, especially in Europe, periods (20.7.69). Since the hyphen (minus sign) and the period (decimal point) are valid numeric characters, you may run into problems when trying to extract the date components. For example, the following expressions return what might be unexpected results:

```
VAL function                  Result
VAL("07-20-69", FREE1)             7
VAL("07-20-69", FREE2)           -20
VAL("07-20-69", FREE3)           -69
```

The REPLACE function replaces a specified character with another. You can use it to replace ambiguous characters with non-numeric characters (such as a blank or a comma). The following expression, for example, replaces hyphens with commas:

```
REPLACE ("07-20-69","-",",")
```

Similarly, the following expression replaces periods with commas:

```
REPLACE ("07.20.69",".",",")
```

You can combine the two REPLACE functions to create a single expression that will replace hyphens or periods with commas:

```
REPLACE(REPLACE ("07-20-69" , "." , ",") , "-" , ",")
```

The program shown below demonstrates using the REPLACE function to insure that input fields do not contain hyphens or periods. It then extracts three date components (mm, dd, yy) and creates a variable C that contains the centuryday representation of the original date. For demonstration purposes, the program then writes the date out in a formatted expression.

In an application you could use the centuryday value (or the mm, dd, yy components) in calculations (an age calculation, for example).

```
  100 SETUP   cutoffyear=30
  200 INPUT   my-date$   : STOP         %  ex 1-12-98
  300 LET     x$=replace(my-date$,"-",",") % change
                           minus sign to comma
  400 LET     x$=replace(x$,".",",")       % change
                           decimal to comma
  500 LET     mm= val(x$,free1)
  600 LET     dd= val(x$,free2)
  700 LET     yy= val(x$,free3)
  800 LET     c= centuryday(yy,mm,dd)
  900 WRITE   my-date$,"is",$(c,date dd mmm yyyy)
 1000 GOTO    200
```

Question

Can I dynamically assign file descriptions to users based on user profiles? I have five copies of a database and I have set up a usercode for each copy. I have user profiles with the same names as the usercodes. I have copies of the file descriptions on each usercode.

Resolution

It's not uncommon for DMSII users to have several copies of each database: a production copy, a test copy, possibly a month-end or year-end copy. The structures in these databases (dataset names, field names, etc.) are usually the same in each copy. To access all the databases, you must create separate file descriptions and interface routines for each database.

The following program dynamically builds a master based on the user profile logon name (which is the same as the database usercode).

```
  100 FDSYS   MASTER DEMO/MASTER
  200 FDSYS   SHOW : F=TEMP/MASTER!
  300 LET     USER$=USERNAME
  400 LET     PACK$=PACKNAME
  500
  600 FILES   *(NAME=TEMP/MASTER)
  650
  700 DEFINE NEWFD$ = 1#.(1 THRU "=" IN 1#) &
        && TRIM(( "(" & USER$ & ")"), BOTH) &
        && #.((")" IN 1#) +1 THRU (" ON" IN 1#) -1) &
        && " ON " & PACK$ & ")"
  800
```

```
1100 LAYOUT 1
1200 - *START OF REPORT
1300 -     "FDSYS" [SKIP 1 LINE AFTER]
1400 -     "MASTER NEW/MASTER" [SKIP 1 LINE AFTER]
1500 -     "REMOVE ALL"
1600 - *DETAIL
1700 -     "ENTER ", NEWFD$ [114]
2000 REPORT 2 THRU END : T0;D1;F=DO/NEW/MASTER!&
          (MAXRECSIZE=120)
2100 DO DO/NEW/MASTER
```

Note: This version of the program uses the demo master distributed with DARGAL.

The first two lines of the program create a text version of the master entries. This text version is placed in a file called **TEMP/MASTER** (line 200). The program reads this text file and modifies the text to replace the usercode for each entry with the current user name.

Following are sample entries from the file **TEMP/MASTER**:

```
MASTER DEMO/MASTER % 12 ENTRIES
TRAINING (FDNAME=(DARGAL)DEMODB/FD/TRAINING ON DISK1)
JOB      (FDNAME=(DARGAL)DEMODB/FD/JOB ON DISK1)
DEPTS    (FDNAME=(DARGAL)DEMODB/FD/DEPTS ON DISK1)
```

Line 300 assigns the current user name (for example, “TEST”) to a variable called **USER$**. Line 400 assigns the current pack (for example, “PACK”) to a variable called **PACK$**. These variables are used later in rebuilding the master entry names.

Line 600 opens the file **TEMP/MASTER** for input.

The DEFINE command in line 700 defines a variable called **NEWFD$** in terms of the input buffer (1#) for the input file.

The following illustration shows the locations in a sample buffer as they are referenced in the different parts of the define.

```
TRAINING (FDNAME=(DARGAL)DEMODB/FD/TRAINING ON DISK1)
                |        |                 |
            "=" IN 1#    |           (" ON" IN 1#) - 1
                 (")" IN 1#) + 1
```

For the sample buffer shown above, with **USER$** = “TEST” and **PACK$** = “PACK”, the DEFINE command assigns the variable **NEWFD$** to be the following:

```
TRAINING (FDNAME=(TEST)DEMODB/FD/TRAINING ON PACK)
```

The net effect is to replace the usercode and pack name with the current values for **USER$** and **PACK$**.

The LAYOUT (line 1100) and REPORT (2000) commands created a file that contains the FDSYS commands that will create or recreate the master **NEW/MASTER**. The REPORT command places these files in a file called **DO/NEW/MASTER**.

The commands in the Start of Report section activate FDSYS, assign the master **NEW/MASTER**, and remove all the entries in it (if there are any). The Detail section adds an enter command for each input record (from **TEMP/MASTER**). For each file description in the original master, the report creates an ENTER command with the new file description name as stored in **NEWFD$**.

The REPORT command starts with the second record in the temporary master (the first record contains the master name).

The DO command in line 2100 processes the commands in the file **DO/NEW/MASTER** to create a master that contains the same entries as the original master. The entries, however, now point to a different usercode and pack.

Question

We moved our data from DMSII to SQL. Will our existing specs and programs continue to work? What happens if column names contain hyphens?

Resolution

Yes, your specs and programs will continue to work. Create an FD for each SQL table using URSA Administrator, EZFD, or FDSYS and save them under the same nicknames you used previously. If you do not want to overwrite your original DMSII versions of the FDs for these files, specify different FDNodes for the new FDs. Also, save the new FDs in a different master from the one that contains your original DMSII versions. The column names need to be exactly the same in the new SQL tables as they were in the old DMSII tables. If any of the column names used in your specs and programs contain hyphens, DARGAL automatically converts the hyphens to underscores when sending the query to the SQL database, since hyphens are not permitted in SQL column names.

Question

Regardless of which of your reporting products I use, I need to create a report every day where the value in a particular field is equal to today's date. How do I do this?

Resolution

You need to add a selection condition to your report. Specify that records where the value in the field equals today's date be selected, for example:

```
CENTURYDAY(1#START-DATE) = CENTURYDAY
```

Date comparisons that use the CENTURYDAY function on a flaged date field or literal date are optimized. When you create selection conditions in EZSPEC, DARGAL uses the centuryday values of the dates you entered, since centuryday values are more efficient.

Question

What do all those end of report error messages in DARGAL, EZSPEC, and URSA mean? Specifically, what are "records excluded on WHERE errors"?

Resolution

There are three types of errors that can occur in a report: selection (WHERE) errors, sequencing errors, and until errors. Errors occur when a field used in a selection condition, sequencing logic, or until condition contains invalid or missing data.

Since Version 57.420, DARGAL has treated dates that contain a value of zero as invalid (unless the date field is a centuryday date type). Consequently, users have seen more of the "excluded on WHERE errors" messages. This change was made, along with other changes, to allow some date selection conditions to be evaluated by the turbo library. Other changes were made at that time to make the evaluation of conditions involving dates as efficient as possible. In selection conditions involving dates, zero is generally not a value you want to include. Records excluded because of errors usually don't meet the condition anyway.

You can use the SETUP command to specify how DARGAL handles errors in selection conditions, sequencing logic, and until conditions (used in 10-record test reports, for example). You can also set these options in EZSPEC on the **Configuration Options Screen** and in URSA in the **Report Options** dialog box.

Here are the SETUP command options and their default and possible values:

Option	**Default value**	**Other values**
SELecterrors	EXClude	INClude, ERRor
UNTILerror	CONTinue	STOP, ERRor
SEQuenceerror	SKIP	BEGinning, END, ERRor

To include records that have invalid or missing data, set the **SELecterrors** option to **INClude**.