SAS® GLOBAL FORUM 2017

April 2 – 5 | Orlando, FL

Text Generational Data Sets (TextGDS)

Macro

USERS PROGRAM

Text Generational Data Sets

Dr. Kannan Deivasigamani

HSBC

ABSTRACT ICLICK TO EDIT)

- Offers a way to create GDG like structure for Text files using SAS in Linux & Unix environment
- Allows users to point to current or historical generations
- Allows users to read or write to historical generations
- Allows a maximum of 10 million generations before a need to recycle
- Capable of altering the code to rollover to go over the 10 million limit

MACRO – PARAMETERS

- The TextGDS macro call is shown in this example below:

```
%TextGDS (f1=/data, f2=Test_, n=+1, m=5);
```

Such a call would result in populating a macro variable *&fn* with a value that could be assigned to the *file-ref* name in the *filename* statement. This value assigned to the macro variable *&fn* will be named in such a way that this resembles a *GDG* structure.

- An example of a read is demonstrated below. In this code below, the parameter *"n"* has a value of *"+0"* which indicates that the user is requesting the macro to point to the current generation of the text file.

```
%TextGDS(f1=/data,f2=Test_,n=+0,m=5);
Filename f1 &fn;
```

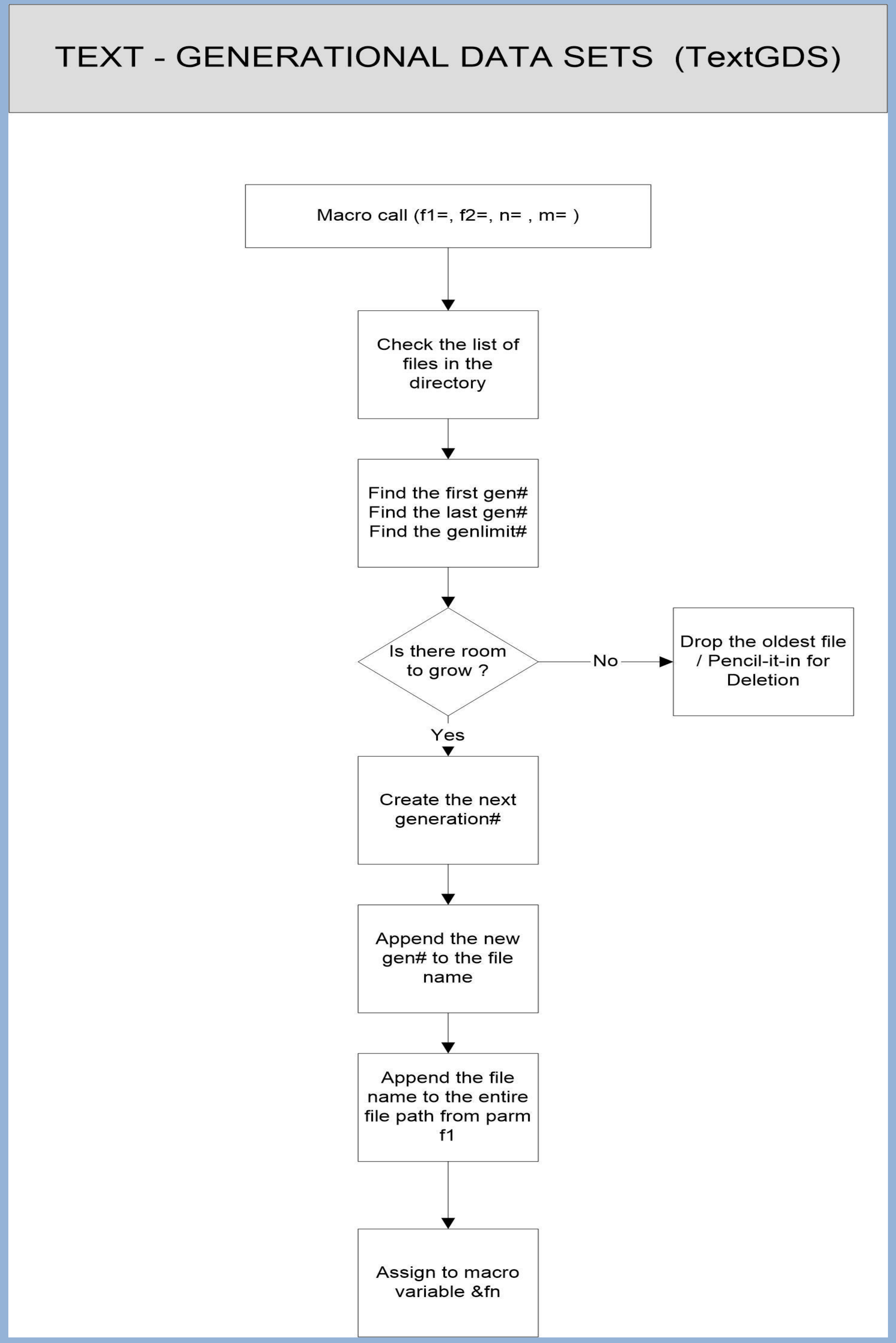
- Parms *f1* and *f2* point to file location and file name respectively. The next parameter n can either have a positive or a negative value. It cannot have an unsigned numeric value. The last parameter m in this case holds a value of 5 which indicates that this macro is set to hold a max of 5 generations. This is equivalent to the *GENLIMIT* parm on the mainframes while defining a *GDG.*

```
%TextGDS (f1=/data, f2=Test_, n=+1, m=5);
Filename f1 &fn;
Data _null_; File f1;Put 'the TextGDS# is:' &fn;
Run;
```

The data folder will have the following file: /data/Text_000001m005.dat

As shown above, the first part of the file name is from parm f2=Test_ and the second part is from parm n=+1 which resulted in 000001. The last part is from the parm m=5 that resulted in 005 which is appended with the file extension ".dat"

MACRO - OPERATION


TEXT - GENERATIONAL DATA SETS (TextGDS)
Macro call (f1=, f2=, n= , m=)
Check the list of files in the directory
Find the first gen# Find the last gen# Find the genlimit#
Is there room to grow ?
No
Drop the oldest file / Pencil-it-in for Deletion
Yes
Create the next generation#
Append the new gen# to the file name
Append the file name to the entire file path from parm f1
Assign to macro variable &fn

Text Generational Data Sets

Dr. Kannan Deivasigamani

HSBC

MACRO – PARAMETERS (CONTINUED)

If the code segment shown above with the macro is invoked again, then the next generation is created and the ls –altr command in the /data folder will show the following 2 files:

```
Text_000001m005.dat
Text_000002m005.dat
```

If the code segment is executed again, the 3rd generation will be created and will have 3 files as shown below:

```
Text_000001m005.dat
Text_000002m005.dat
Text_000003m005.dat
```

A repeat of the same will result in the 4th file and finally another repetition will result in 5 files as shown below:

```
Text_000001m005.dat
Text_000002m005.dat
Text_000003m005.dat
Text_000004m005.dat
Text_000005m005.dat
```

Here is the interesting part, if the code is executed again, the 6th generation will be created and most recent 5 generations will be available for use as shown below.

```
Text_000002m005.dat
Text_000003m005.dat
Text_000004m005.dat
Text_000005m005.dat
Text_000006m005.dat
```

MACRO – PARAMETERS (CONTINUED)

The oldest generation #000001 will be deleted and the most recent 5 generations will be available. The oldest generations will keep dropping off as new generations are created. The macro internally manages these generation retention process using the work space mentioned earlier on in the paper. The macro reads the Meta data in the current directory which is nothing but the list of generations matching the file name provided in the macro parameter and stores it for file management operations in the macro. The maximum and minimum generations are identified and their generations numbers are handled depending on the max gen limit that the macro initially defined it to hold. If the user is requesting a +1, then the generation number is incremented to the next higher number and the oldest generation is deleted within the macro code. So far, we have seen examples about creating new TextGDS files but not reading in existing current and historical generations which is discussed in the next section.

Text Generational Data Sets

Dr. Kannan Deivasigamani

HSBC

TEXT GDS – READ OPERATION

A macro call with a negative sign to parameter “n” will result in a read operation of a historic generation as specified by the numeric value assigned to the parameter “n”.

```
%TextGDS (f1=/data, f2=Test_, n=-1, m=5);
Filename f1 &fn;
Data _null_;
          File f1;
          Put 'the TDS# is:' &fn;
Run;
```

A macro call with a value of -1 will result in reading the previous generation of the file. As an example if we have the following 5 files in the /data folder if a read is attempted with a -1 as described, Text_000005m005.dat will be referenced by the &fn macro variable and assigned in the filename statement.

```
Text_000002m005.dat
Text_000003m005.dat
Text_000004m005.dat
Text_000005m005.dat
Text_000006m005.dat
```

Similar to this, if a -2 is the value to the parameter m, then Text_000004m005.dat will be the file name that the &fn macro variable will be pointing as a result. If the macro is referencing a generation that is out of scope, then as one would expect, the code would not be successful in pointing to the generation.

The macro is intelligent to recognize what the user is requesting and perform the operation in the UNIX region using a temporary work space for internal calculation and populates the final output variable which is nothing but a path followed by the file name that includes the fixed name along with the variable generation number followed by an extension indicating the generation limit that the file structure was defined for.

LIMITATIONS & DELIMITATIONS

- Highest gen possible with the current setup is about 10 million beyond which it requires a reset.
- The maximum value that can be held by the 3 digit gen-limit value is 999 which is identical to a SAS GDS.
- The extension of the file which is ".dat" may also be modified to ".txt" or another type depending on the need of a project. On the other hand, all these 3 limitations discussed may also be converted to parms that one may input to the macro to accept the values dynamically as the macro is invoked. However, all these enhancements call for a code change in the macro and the way the macro will be invoked.
- Another limitation or requirement is, this macro requires a temporary work storage for the computation where the "meta" file is written and cleared out at the end of the macro execution and will be left with no trace. The workspace or the temporary folder is a requirement in the current design. One might choose to have this provided in the parm if desired; however, in the current setup, the invoking job will have a subfolder within the work folder and will be used for the computation. The user, depending on the setup might choose to add a command "mkdir tmp" in place of the work folder as one deems appropriate.
- The macro does not prevent anyone from invoking to point to a "+5" or "> +1 " without warning the user of getting ahead of the gen numbers. Therefore, the macro needs to be used with caution and it will be the responsibility of the user to invoke with appropriate parameters.

CONCLUSION

This tool can be enhanced to make it more robust but is a good start and can be utilized by developers. The TextGDS feature, someday hopefully becomes available as part of native SAS language for developers to use it for maintaining generational text files when organizations need them.

RECOMMENDED READING

Base SAS® Procedures Guide

SAS® For Dummies®

Linux/Unix literature

SAS® GLOBAL FORUM 2017

April 2 – 5 | Orlando, FL