Paper CT-23

Introducing a New FINDIT Macro: An Efficient Tool for Simultaneously Searching Several Free-Text Fields Using Multiple Keywords

LaTonia Richardson, Centers for Disease Control and Prevention, Atlanta, GA

ABSTRACT

Have you ever faced the daunting task of searching for information within a free-text field? Even with a list of specific keywords, it is tedious to write multiple "where like" statements or to apply the SAS® INDEX function for each keyword. This paper presents a new FINDIT macro that enables users to quickly find information based on many keywords in multiple free-text fields. Using DATA step and PROC SQL commands within an iterative “do loop” procedure, the macro simultaneously searches all the free-text fields and keywords specified. The best part is that the results are presented in a user-friendly summary report listing the specific keyword found, the variable that contained it, and the exact text found by the search. This macro considerably reduces the need for manually reading information stored in free-text fields. It is the next-best thing to perusing with the human eye!

INTRODUCTION

Finding specific information within free-text fields can be a time-consuming task. This effort often requires thoroughly reading several lines of text or lengthy paragraphs to extract needed information. The use of keywords to search for relevant subject matter can markedly reduce the burden of manually reading free-text information. However, most keyword search tools only permit one keyword or one field to be searched at a time. This paper introduces a SAS® macro that offers a more efficient way to search multiple keywords across multiple fields.

IMPORTING THE DATA

The FINDIT macro can accommodate datasets of various formats, including Access, Excel, SAS, and SQL databases. Default import statements for each format are provided. Users may select the appropriate import statement and replace default values with user-specific values as shown below:

```
/*Access*/
%let AccessDB=C:\My Documents\MyDatabase.mdb; /*Replace with the appropriate
location (folder name and file name) of the Access database*/
%let AccessTable=MyTable; /*Replace with the appropriate Access table name*/
proc import out= work.SearchData datatable= "&AccessTable." dbms=access
replace; database="&AccessDB.";run;

/*Excel*/
%let ExcelDB=C:\My Documents\MyExcel.xls; /*Replace with the appropriate
location (folder name and file name) of the Excel file*/
%let ExcelWorksheet=Sheet1; /*Replace with the appropriate Excel worksheet
name*/
proc import datafile="&ExcelDB." out=work.SearchData dbms=excel
replace;sheet="&ExcelWorksheet.";run;

/*SAS*/
libname search "C:\My Documents";/*Replace with the appropriate location
(folder name) of the SAS dataset*/
%let SASDB=MySAS ;/*Replace with the appropriate SAS dataset
data work.SearchData;set search.&SASDB.;run;

/*SQL Database*/
%let SQLServer=MyServer; /*Replace with the appropriate SQL server name*/
%let SQLDB=MyDB; /*Replace with the appropriate SQL database name*/
%let SQLTable=MyTable; /*Replace with the appropriate SQL table name*/
%macro sqlpath;
 libname search oledb provider=SQLOLEDB schema=dbo
 properties=('data source'= "&SQLServer." , 'initial catalog'=
 "&SQLDB." ,'integrated security'=SSPI);
```

```
%mend;

%sqlpath;

 data work.SearchData;set search.&SQLTable.;run;
```

Users should select only one of the import statements above, based upon the dataset to be imported. All free-text fields that the user would like to search should be located in the imported dataset. If free-text fields are located in more than one dataset (e.g., multiple Access tables) then the FINDIT macro should be run separately for each dataset.

SETTING INITIAL MACRO VARIABLES

After importing the data, the next step is to set initial macro variables used by the FINDIT macro. First, users should specify the variable that uniquely identifies each record (the primary key). Then the number of free-text fields to be searched should be specified. The number of fields must be greater than 0 and can be as large as the imported dataset can accommodate. For example, the maximum number of fields for an Access 2007 dataset is 255[1], so a maximum of 255 free-text fields can be searched for an imported Access dataset. Next, users should list each field to be searched, where "Fld1" represents the first field, "Fld2" represents the second field, and so forth. Ten macro variables for the field names are provided by default (Fld1-Fld10), but users can modify the code to create additional macro variables as needed. Next, users should list each keyword separated by commas. For example, to search for dairy-related foods within the specified free-text fields, users may list the following keywords "dairy, milk, cream, butter, cheese". Users may list as many keywords as desired, and this is one of the distinguishing features of the FINDIT macro. However, the length of the keyword list must not exceed the maximum length of SAS character variables (32,767 bytes[2]). Unlike the number of free-text fields, users are not required to enter the number of keywords, as this number is auto-calculated by the tool. Finally, users may specify a location and file name of an auto-generated Excel file that summarizes the search results.

RUNNING THE MACRO

Running the FINDIT macro begins with counting the number of keywords specified by the user. First, a mock keyword called "endoflist" is inserted at the end of the keyword list in order to flag the end of the list. Then, the keywords are separated such that each word is listed as a separate record. Next, the mock keyword is deleted and the number of remaining records indicates the number of keywords. Macro variables are then created for each keyword and for the number of keywords.

```
 /*Separate the keywords*/
 data searchtxt;
keep count word;
length word $300;
length string $32767.;
string = compbl("&txtlist."||","||"endoflist");
delim = ',';
modif = 'mo';
nwords = countw(string, delim, modif);
do count = 1 to nwords;
word = scan(string, count, delim, modif);
output;
end;
run;

/*Remove the mock keyword "end of list"*/
 data searchtxt;set searchtxt;length fndend
 8.0;fndend=index(word,'endoflist');if not fndend>0;drop fndend;run;

/*Create macro variables for each keyword and the number of keywords*/
 data _null_;set searchtxt end=lastrec;call symput('txt'||left(_n_),trim(word));
 if lastrec then call symput('nwords',left(_n_));run;
```

Next, a table called "SearchResults" is created using PROC SQL. The variables included in the table are: PrimaryKey, Keyword, Location (field where the keyword was found), and SpecificTextFound (specific text entered in the keyword field). The macro searches for each keyword in each free text field, and if the keyword is found then the

primary key, keyword, location, and specific text found is appended to the “SearchResults” table. The entire process takes place as an iterative “do loop” procedure. After running the keyword search, the results are exported to a user-friendly Excel file. First, a line list of all search results is created within a worksheet called “Complete Results” (Table 1). This is essentially an Excel version of the “SearchResults” table where all keyword results were appended. Then, for each record listed in the “SearchResults” table, a summary of the total keywords found and a list of the keywords found (separated by ‘|’) is provided in the “SummaryResults” worksheet (Table 2).

Table 1. “Complete Results” Excel Worksheet

PrimaryKey	Keyword	Location	SpecificTextFound
AA9887	MILK	FoodItem	Milk with cereal
AA9887	CREAM	FoodDetails	Whipped cream
BB2264	BUTTER	FoodItem	Butter or margerine
CC1234	MILK	FoodDetails	The ingredients included milk, cheese and butter
CC1234	CHEESE	FoodDetails	The ingredients included milk, cheese and butter
CC1234	BUTTER	FoodDetails	The ingredients included milk, cheese and butter

Table 2. “Summary Results” Excel Worksheet

PrimaryKey	TotalKeywordsFound	ListofKeywordsFound
AA9887	2	MILK|CREAM
BB2264	1	BUTTER
CC1234	3	MILK|CHEESE|BUTTER

CONCLUSION

The FINDIT macro offers an efficient way to search several free-text fields using an extensive list of keywords. It is the ideal solution for finding useful information without spending lots of time reading text in multiple variable fields. The macro accommodates multiple file types and provides users the flexibility to modify the code to meet their specific needs.

REFERENCES

1. *Access 2007 Specifications.* (2007). Retrieved Aug 2, 2012, from http://office.microsoft.com/en-us/access-help/access-2007-specifications-HA010030739.aspx
2. *Setting the Length of Character Variables.* (2011). Retrieved Aug 2, 2012, from http://support.sas.com/documentation/cdl/en/basess/58133/HTML/default/viewer.htm#a001336069.htm

COMPLETE CODE

For simplicity with running the code on your own.

```
/*IMPORT DATA: CHOOSE ONLY ONE FILE FORMAT*/

/*Access*/
%let AccessDB=C:\My Documents\MyDatabase.mdb; /*Replace with the appropriate
location (folder name and file name) of the Access database*/
%let AccessTable=MyTable; /*Replace with the appropriate Access table name*/
proc import out= work.SearchData datatable= "&AccessTable." dbms=access
replace;database="&AccessDB.";run;

/*Excel*/
```

```
%let ExcelDB=C:\My Documents\MyExcel.xls; /*Replace with the appropriate
location (folder name and file name) of the Excel file*/
%let ExcelWorksheet=Sheet1; /*Replace with the appropriate Excel worksheet
name*/
proc import datafile="&ExcelDB." out=work.SearchData dbms=excel
replace;sheet="&ExcelWorksheet.";run;

/*SAS*/
libname search "C:\My Documents";/*Replace with the appropriate location
(folder name) of the SAS dataset*/
%let SASDB=MySAS ;/*Replace with the appropriate SAS dataset (eg.,
MySAS.sas7bdat)*/
data work.SearchData;set search.&SASDB.;run;

/*SQL Database*/
%let SQLServer=MyServer; /*Replace with the appropriate SQL server name*/
%let SQLDB=MyDB; /*Replace with the appropriate SQL database name*/
%let SQLTable=MyTable; /*Replace with the appropriate SQL table name*/
%macro sqlpath;
 libname search oledb provider=SQLOLEDB schema=dbo
 properties=('data source'= "&SQLServer." , 'initial catalog'=
"&SQLDB." ,'integrated security'=SSPI);
 %mend;

 %sqlpath;

 data work.SearchData;set search.&SQLTable.;run;

/*SET INITIAL MACRO VARIABLES*/
%let primarykey=rootid; /*Specify the primary key (aka unique identifier) of
the imported dataset*/
%let numflds=1; /*Specify the number of free-text fields you wish to search*/
%let fld1=FoodName; /*List each free-text field you wish to search. To search
more than 10 fields, modify the code accordingly*/
%let fld2=MyField2;
%let fld3=MyField3;
%let fld4=MyField4;
%let fld5=MyField5;
%let fld6=MyField6;
%let fld7=MyField7;
%let fld8=MyField8;
%let fld9=MyField9;
%let fld10=MyField10;

%let txtlist=dairy, milk, cream, butter, cheese; /*List each keyword separated
by a comma*/

%let outputFolder= C:\My Documents;/*Specify the location of the output Excel
file generated by the macro*/
%let outfilename=AccessTest;/*Specify the name of the Excel file generated by
the macro*/


/*RUN THE MACRO*/

%macro FINDIT;

data SearchData;set SearchData;length primarykey2
$255.;primarykey2=put(&primarykey.,8.0);run;

data searchtxt;
keep count word;
```

```
 length word $300;
 length string $32767.;
 string = compbl("&txtlist."||","||"endoflist");
 delim = ',';
 modif = 'mo';
 nwords = countw(string, delim, modif);
 do count = 1 to nwords;
word = scan(string, count, delim, modif);
output;
 end;
 run;

 data searchtxt;set searchtxt;length fndend
 8.0;fndend=index(word,'endoflist');if not fndend>0;drop fndend;run;

 data _null_;set searchtxt end=lastrec;call symput('txt'||left(_n_),trim(word));
 if lastrec then call symput('nwords',left(_n_));run;

 proc sql;create table searchresults
 (PrimaryKey char(255),Keyword char(20),Location char(100), SpecificTextFound
 char(3000));quit;


 %do j=1 %to &numflds.;

       %do i=1 %to &nwords.;
       data _null_;length uppertext
       $100.;uppertext=compress("%"||upcase("&&txt&i.")||"%");call
       symput('uptext',compress(uppertext));run;
       data temp; set SearchData;
       length keyword1 $20.;keyword1="&uptext.";
       length keyword $20.;keyword=tranwrd(keyword1,"%"," ");
       where upcase(compress(&&fld&j.)) like "&uptext." ;
       length location $100.;location="&&fld&j.";
       length sptext $3000.;sptext=&&fld&j.; if not (keyword1 in ("%
       %","%*%"));run;

       proc sql;insert into searchresults(PrimaryKey,Keyword,Location,
       SpecificTextFound)
       select a.primarykey2 as PrimaryKey,a.keyword as Keyword,a.location as
       Location,a.sptext as SpecificTextFound
       from temp a;quit;
       %end;
 %end;


 proc sql;create table kwcount as
 select primarykey, count (distinct keyword) as TotalKeywordsFound
 from searchresults where not (missing(keyword)) group by primarykey;quit;

 data searchresults0;set searchresults;length primkey
 $100.;primkey=compbl(primarykey||keyword);run;

 proc sort data=searchresults0 out=searchresults0 nodupkey;by primkey;run;

 proc sort data=searchresults0;by primarykey keyword;run;

 proc transpose data=searchresults0 out=kwlist(drop=_name_);var keyword;by
 primarykey;run;

 data list2;set kwlist;length type $4.;type="list";run;

 ods listing close;
```

```
ods output Contents.DataSet.Variables=list2out;
proc contents data=kwlist;run;

proc sql;create table numvars as select "list" as type, max(num)-1 as cols from
list2out;quit;

proc sort data=list2;by type;run;
proc sort data=numvars;by type;run;

data list3;merge list2 numvars(keep=type cols);by type;if not (cols=.) then
call symput('numcols',cols);run;

%macro getlist;
%do i=1 %to &numcols.;
col&i.,
%end;
" "
%mend;

data kwlist2;set list3; length ListofKeywordsFound $32767.;
ListofKeywordsFound=catx('|',%getlist);keep primarykey ListofKeywordsFound;run;

data searchresults;set searchresults;if not (keyword in ("","  ","
","*"));run;

proc export data=searchresults outfile="&outputfolder.\&outfilename..xls"
dbms=excel replace;sheet="Complete Results";run;

proc sort data=searchresults out=searchresults nodupkey;by primarykey;run;
proc sort data=kwcount;by primarykey;run;
proc sort data=kwlist2;by primarykey;run;

data searchsummary;merge searchresults (drop=location specifictextfound)
kwcount kwlist2(keep=primarykey ListofKeywordsFound);if not (keyword in ("","
","    ","*"));
drop keyword;run;

proc export data=searchsummary outfile="&outputfolder.\&outfilename..xls"
dbms=excel replace;sheet="Summary Results";run;

run;

%mend;

%FINDIT;
```

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

LaTonia Richardson, MS
Statistician
Centers for Disease Control and Prevention
1600 Clifton Road, MS C-09
Atlanta, GA 30333
LCRichardson@cdc.gov