PharmaSUG 2019 - Paper BP-029

Lit Value Locator: An Efficient Way to Pinpoint Data

Varunraj Khole, Jefferson University

ABSTRACT

This paper describes a method to locate any alphanumeric text string, in character or numeric format within any dataset present in default as well as user defined SAS® libraries. Often a times, data presented to us comes in different format and is stored in multiple locations. To ensure that all the required data is captured, a user needs to implement smart coding which can find the required data and give its location. This method would be very useful while searching for a particular data string when the user is working with multiple SAS® libraries. This method can be an essential tool for a data analyst working in finance, pharmaceuticals, life sciences, healthcare, banking and various other industries. By combining powerful SAS® macro language, procedures, efficient do loops and handful of extremely useful SAS® functions this method will provide the user with a detailed description of the search value and provide summary statistics for the same. This method is an example that an extremely complicated task can be achieved with very simple and efficient coding.

INTRODUCTION

This paper introduces Lit Value Locator (LVL), a process to locate any character or number from any dataset in SAS® default libraries as well as user defined libraries. At times it becomes difficult to locate a particular value (character or numeric) from multiple libraries and datasets. To find the value, a user needs to review metadata or additional documentation resulting in increased time consumption.

LVL performs detailed checks within multiple SAS® libraries and provides the user with the exact location of the required value within SAS® libraries, datasets and variables. It makes a time-consuming task very simple and automatic.

By combining powerful SAS® macro language, procedures, efficient do loops and handful of extremely useful SAS® functions this method will provide the user with a detailed description of the search value along with an HTML report. Think of it as a “Ctrl + F” for all the user defined and SAS® default libraries. This macro is tested on SAS® Studio and SAS® 9.4.

LVL PARAMETERS

- **Target:**

  This macro parameter defines the value which the macro will search for.

- **Targetval_num:**

  This macro parameter defines whether the target value is numeric or not.

  The default value of this parameter is ‘N’.

  Required only if target value is numeric.

  If the user chooses ‘N’ then the entire target value will be considered as character.

- **Entire_str:**

  This macro parameter defines whether the entire string is to be searched or not.

  If the user chooses ‘Y’ then the entire string will be searched ‘as is’ in the dataset.

  If user chooses ‘N’ then the variable containing the targeted value will be searched.

- **Ignore_case:**

  This macro parameter gives user the option of searching the target ‘as is’ or to ignore case.

  If user chooses ‘Y’ then the macro will ignore the case.

  If user chooses ‘N’ then the macro will search for the value ‘as is’.

- **Inc_sas_def_lib:**

  This macro parameter gives user the option to include or exclude SAS® default library from the search.

- **Inc_sas_wrk_lib:**

  This macro parameter gives user the option to include or exclude SAS® work library from the search.

- **Search_lib:**

  This macro parameter lets user choose the library they wish to search.

  User may leave it blank or enter any user defined library in “ “ . For e.g. %str(“sdtm” “adam” “abc” “xyz”). Cannot be used in combination with “inc_sas_def_lib” or “inc_sas_wrk_lib”. If a user needs to include SAS® work or default libraries then those libraries need to be specified in “search_lib” macro parameter.

- **Outfilepath:**

  This macro parameter defines the path for the html output.

- **Filename:**

  This macro parameter defines the name of the output file.

PARAMETERS: OPTIONS AND DEFAULT SETTINGS

Macro variable	**Option**	**Default**	**Required**
Target			Y
Target_val_num	Y or N	N	N
Entire_str	Y or N	Y	N
Ignore_case	Y or N	Y	N
Inc_sas_def_lib	Y or N	N	N

Inc_sas_wrk_lib	Y or N	N	N
Search_lib	user defined SAS@ library in “ “ or null		N
Outfilepath			Y
Filename			Y
Debug	Y or null		N

Table 1.0 - LVL options and default settings.

PROCESS FLOW

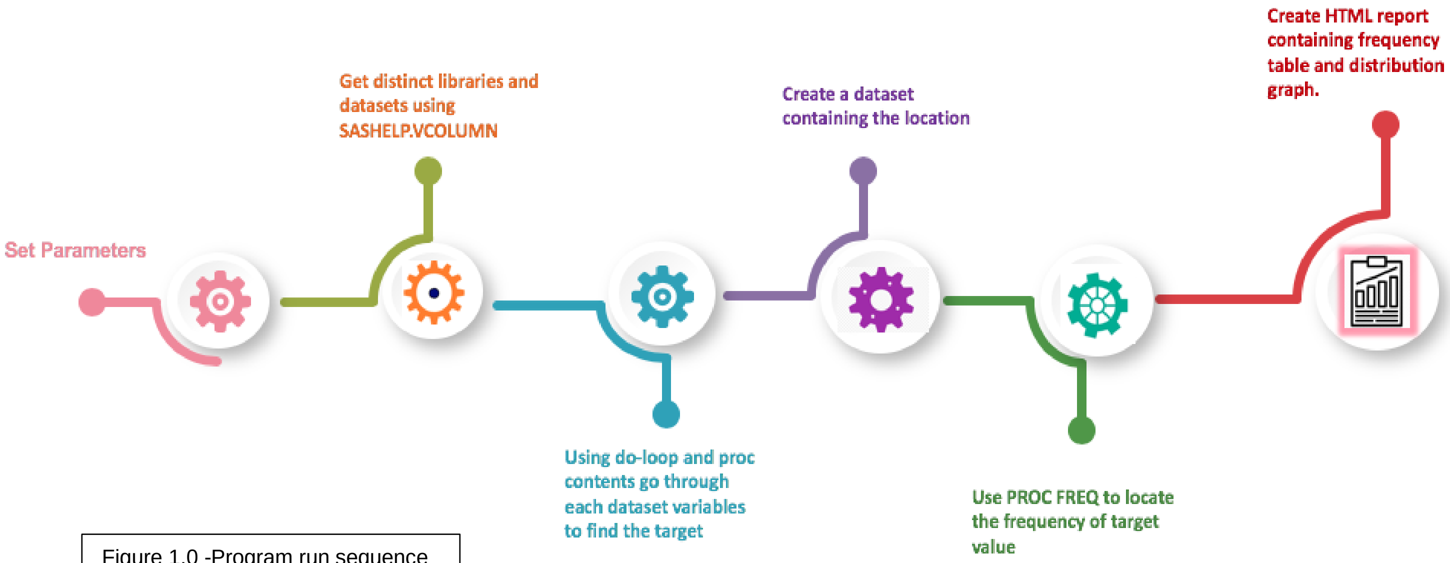

Set Parameters
Get distinct libraries and datasets using SASHELP.VCOLUMN
Using do-loop and proc contents go through each dataset variables to find the target
Create a dataset containing the location
Use PROC FREQ to locate the frequency of target value
Create HTML report containing frequency table and distribution graph.

Figure 1.0 -Program run sequence

EXAMPLE 1: LOCATING CHARACTER VALUE

A situation occurs wherein one needs to find ‘adverse events’ which include the word ‘head’ and isn’t sure which dataset to look into.

By executing LVL, one can find the target value within a short time by defining libraries which they wish to include in the search and setting the ‘target’, ‘outfilepath’, ‘filename’, also selecting special options such as whether to look for ‘entire string’ and whether to ‘ignore case’.

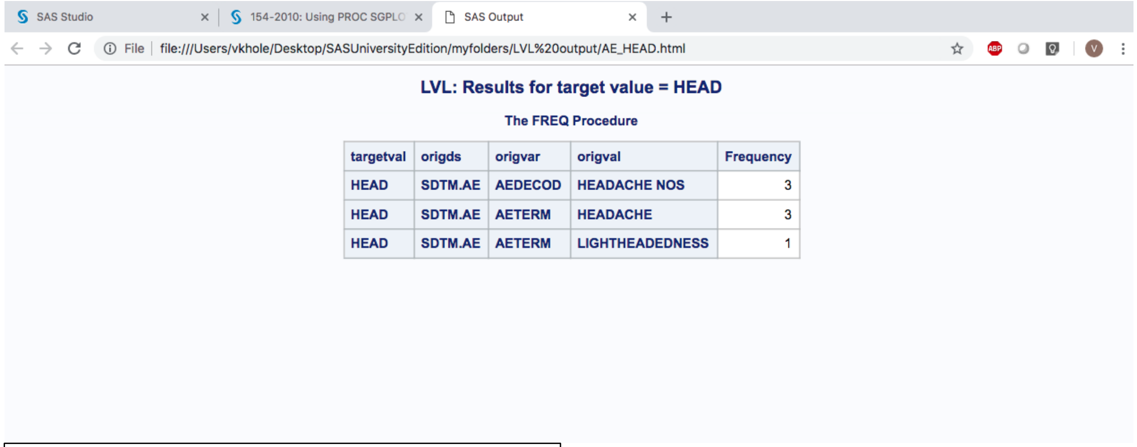
Following is the part of macro code one needs to focus on:

```
%lit_val_loc(target = hEaD
             ,target_val_num = N
             ,entire_str = N
             ,ignore_case = Y
             ,inc_sas_def_lib = N
             ,inc_sas_wrk_lib = N
             ,search_lib = "sdtm"
             ,debug =
             ,outfilepath = /folders/myfolders/lvl output
             ,filename = AE
             )  ;
```

LVL will search for a variable value containing “HEAD” in all the datasets in “sdtm” library. In this example, LVL will search for “HEAD” as a partial string and will ignore its case. If the user does not define search_lib macro parameter it will then search for all the datasets within all the defined libraries except for SAS® default and WORK libraries.

The following HTML output generated by executing LVL includes 4 variables namely, targetval, origds, origvar and origval which provide us with detailed information and the graph contains variable ‘origloc’ which displays the libname, dataset and variable name.

- ‘Targetval’ is the value used for the search.
- ‘Origds’ tells you the libname and dataset in which the targetval was found.
- ‘Origvar’ and ‘Origval’ tell you the variables and the corresponding original values the targetval was found in respectively.

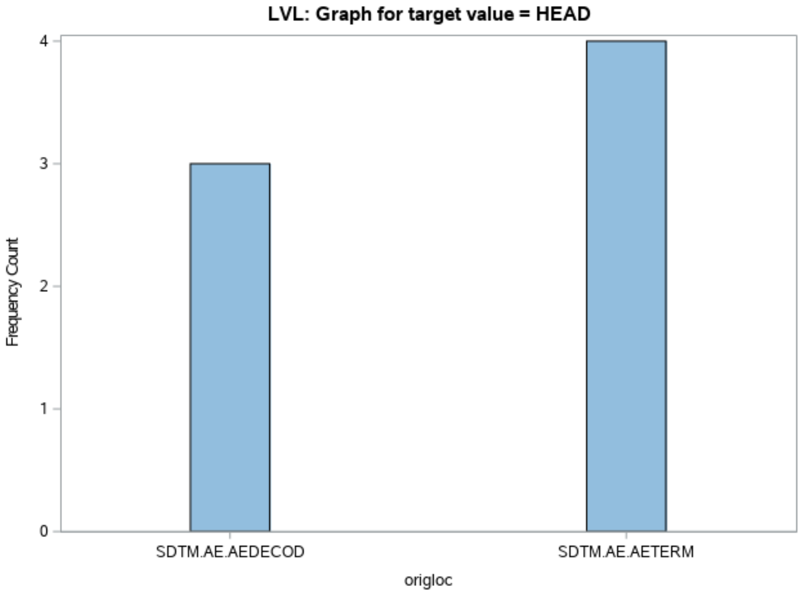

SAS Studio
154-2010: Using PROC SGPLO
SAS Output
File
file:///Users/vkhole/Desktop/SASUniversityEdition/myfolders/LVL%20output/AE_HEAD.html

LVL: Results for target value = HEAD

The FREQ Procedure

targetval	origds	origvar	origval	Frequency
HEAD	SDTM.AE	AEDECOD	HEADACHE NOS	3
HEAD	SDTM.AE	AETERM	HEADACHE	3
HEAD	SDTM.AE	AETERM	LIGHTHEADEDNESS	1

Output 1.1 – Example of output produced by LVL



Output 1.2 -Example of graph of the above output produced by LVL

EXAMPLE 2: LOCATING NUMERIC VALUE

A situation occurs wherein one needs to find a particular number, in this case 111, from numerous datasets. This can be a tedious process, but, by executing LVL, one can find the target value within a short time by defining libraries which they wish to include in the search and setting the 'target', 'target_val_num', 'outfilepath', 'filename'.

Following is the part of macro code one needs to focus on:

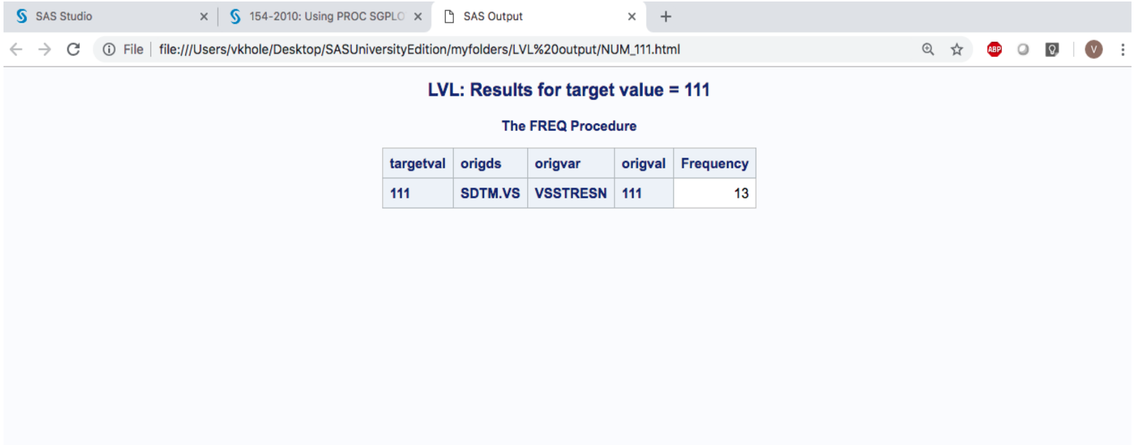
```
%lit_val_loc(target = 111
             ,target_val_num = Y
             ,entire_str =
             ,ignore_case =
             ,inc_sas_def_lib = N
             ,inc_sas_wrk_lib = N
             ,search_lib = "sdtm"
             ,debug = Y
             ,outfilepath = /folders/myfolders/lvl output
             ,filename = NUM
             )  ;
```

LVL will search for a variable value containing "111" in all the datasets in "sdtm" library. In this example, LVL will search for "111" as a number because 'target_val_num' is Y hence 'entire_str', 'ignore_case' options are not needed. If the user does not define search_lib macro parameter it will then search for all the datasets within all the defined libraries except for SAS® default and WORK libraries.

The following HTML output generated by executing LVL includes 4 variables namely, targetval, origds, origvar and origval which provide us with detailed information and the graph contains variable 'origloc' which displays the libname, dataset and variable name.

- 'Targetval' is the value used for the search.

- ‘Origds’ tells you the libname and dataset in which the targetval was found.
- ‘Origvar’ and ‘Origval’ tell you the variables and the corresponding original values the targetval was found in respectively.

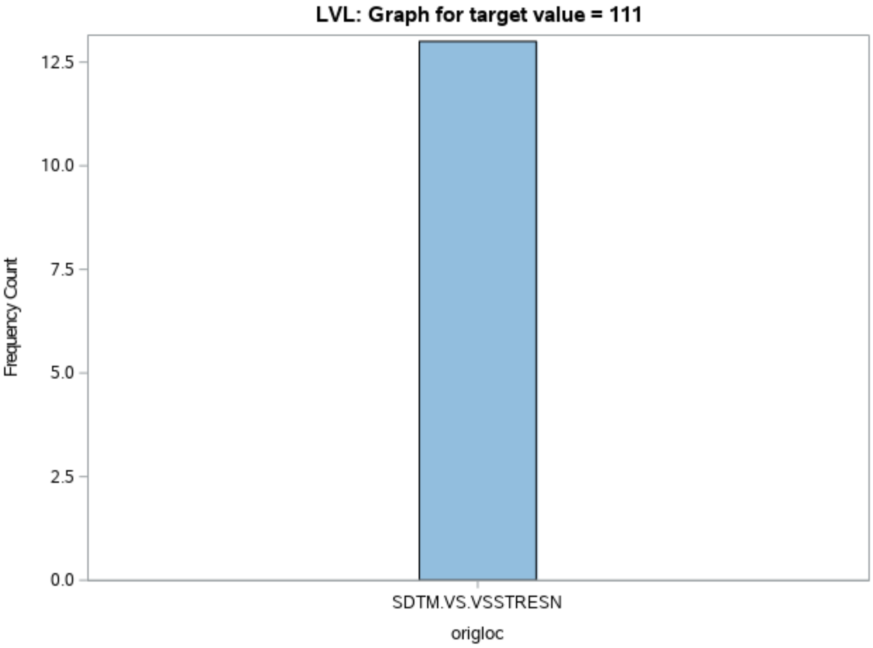

SAS Studio
154-2010: Using PROC SGPLO
SAS Output
File
file:///Users/vkhole/Desktop/SASUniversityEdition/myfolders/LVL%20output/NUM_111.html

LVL: Results for target value = 111

The FREQ Procedure

targetval	origds	origvar	origval	Frequency
111	SDTM.VS	VSSTRESN	111	13

Output 2.1- *Example of output produced by LVL*



Output 2.2- *Example of graph for the above output produced by LVL*

CONCLUSION

In this paper features of LVL are demonstrated. Two examples are given, first one for character and second for numeric value. In addition to the examples, a detailed information about all the macro parameters and how to use them is provided. Lit Value Locator serves the purpose of automatically finding any value in SAS® libraries and datasets in an efficient manner. By using LVL, users no longer have to spend their time trying to find a particular value manually and this macro can come very handy to find any given value within multiple libraries. All the user needs to do is to update the macro parameters and run the macro to get LVL report of the target location.

CONTACT INFORMATION

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

VARUNRAJ RAMAN KHOLE
267-576-9439
khole7937@mail.philau.edu

SOURCE CODE

Following is the code used to create LIT VALUE LOCATOR:

```
 options mprint nomlogic ;

 %macro lit_val_loc(target =
                        ,target_val_num =
                        ,entire_str =
                        ,ignore_case =
                        ,inc_sas_def_lib =
                        ,inc_sas_wrk_lib =
                        ,search_lib =
                        ,debug =
                        ,outfilepath =
                        ,filename =
                        ) ;



/* SET DEFAULT VALUES */


%if &target_val_num. = %nrbquote() %then %do ;
       %let target_val_num = N ;
%end;

%if &entire_str. = %nrbquote() %then %do ;
       %let entire_str = Y ;
%end;

%if &ignore_case. = %nrbquote() %then %do ;
       %let ignore_case = Y ;
%end;
```

```
%if &inc_sas_def_lib. = %nrbquote() %then %do ;
       %let inc_sas_def_lib = N ;
%end;

%if &inc_sas_wrk_lib. = %nrbquote() %then %do ;
       %let inc_sas_wrk_lib = N ;
%end ;


%if &search_lib. ne %nrbquote() %then %do ;
       %let search_lib = %upcase(&search_lib.) ;
%end ;




%if &ignore_case. = Y %then %do ;
    %let target = %upcase(&target.) ;
%end ;

%if &debug. = Y %then %do;
   %put target = &target. ;
%end;




/*        STEP 1  GET library information in a dataset        */


data libraries1;
    set sashelp.vcolumn;

    %if &inc_sas_def_lib. = N %then %do ;
       if upcase(libname) not in ('SASHELP' 'SASUSER' 'WEBWORK') ;
    %end;

run;

data libraries2;
       set libraries1;

    %if &inc_sas_wrk_lib. = N %then %do ;
       if upcase(libname) not in ('WORK') ;
    %end ;

run;



data libraries;
       set libraries2;

    %if &search_lib. ne %nrbquote() %then %do ;
       if upcase(libname)  in (&search_lib.) ;
    %end ;

    lib_ds= strip(libname)||'.'||strip(memname) ;

run;
```

```
proc sql noprint;
     select distinct(lib_ds) into: alllib separated by '~'
     from libraries
     ;
     %let libcnt = &sqlobs ;

quit;

%if &debug. = Y %then %do ;
       %put libcnt = &libcnt ;
       %put alllib = &alllib ;
%end ;




/*        STEP 2   GET ALL SAS DATASETS IN INDIVIDUAL LIBRARY       */
options mprint ;

  %do i = 1 %to &libcnt. ;
            %let indlib = %scan(&alllib., &i., '~') ;
            %if &debug. = Y %then %do;
                     %put indlib = &indlib ;
               %end;


       /*        STEP 3    GET ALL VARIABLES IN EACH DATASET IN EACH INDIVIDUAL
LIBRARY     */

              proc contents data= &indlib.
                    out=%substr(&indlib.,%index(&indlib., .)+1)_cont noprint;
              run;

              proc sql noprint;
                  select distinct(name) into: chrvar separated by '~'
                  from %substr(&indlib.,%index(&indlib., .)+1)_cont
                     %if &target_val_num. = N %then %do ;
                    where type=2
                  %end;
                     %if &target_val_num. = Y %then %do ;
                    where type=1
                  %end;
                  ;
            %let chrcnt = &sqlobs;

             quit;
           %if &debug. = Y %then %do;
              %put chrcnt = &chrcnt ;
                 %put chrvar = &chrvar ;
           %end;


               %do j = 1 %to &chrcnt ;
                  %let chrvarval = %scan(&chrvar., &j., "~") ;
                  %if &debug. = Y %then %do;
                            %put chrvarval = &chrvarval. ;
                  %end;


                   data chkchr_&chrvarval.;
                   length targetval origds origvar origval $1024. ;
```

```
                        set &indlib. ;

                            %if &target_val_num. = N %then %do ;


                              %if &entire_str. = N %then %do ;
                                        %if &ignore_case. = Y %then %do;
                                        where index(upcase(&chrvarval.),
"&target.") > 0 ;
                                  %end;
                                        %if &ignore_case. = N %then %do;
                                        where index(&chrvarval., "&target.") >
0 ;
                                  %end;
                              %end;

                              %if &entire_str. = Y %then %do ;
                                        %if &ignore_case. = Y %then %do;
                                        where upcase(&chrvarval.) = "&target.";
                                  %end;
                                        %if &ignore_case. = N %then %do;
                                        where &chrvarval. = "&target.";
                                  %end;
                              %end ;

                              targetval = "&target." ;

                            %end ;


                            %if &target_val_num. = Y %then %do ;
                                where &chrvarval. = &target.;
                              targetval = &target. ;
                            %end ;

                    origds = "&indlib." ;
                    origvar = "&chrvarval." ;
                    origval = &chrvarval. ;
                   run;

              %end;

  %end;

   data __all_ds(keep=targetval origds origvar origval origloc) ;
       set chkchr_: ;
       length origloc $100.;
       origloc = strip(origds)||"."||strip(origvar);
   run;



   ODS Listing Close;
   ODS HTML
   path="&outfilepath."
   body="&filename._&target..html";
   title "LVL: Results for target value = &target." ;
   proc freq data = __all_ds ;
   tables targetval*origds*origvar*origval/ list missing nocum nopercent
out=___final_ds ;
   run;
   run;
   title;
   ODS HTML Close;
   ODS Listing;


   proc freq data=__all_ds noprint;
   table  origloc / out=graphds ;*plots=freqplot( orient=vertical  ) ;
   run;


   ODS Listing Close;
   ODS HTML
   PATH = "&outfilepath."
   body="&filename._&target._graph.html";
   title "LVL: Graph for target value = &target." ;
   proc sgplot data=graphds ;
   vbar origloc / response=count  barwidth=0.2 fillattrs=graphdata7
   ;
   run;
   title;
   ODS HTML Close;
   ODS Listing;


%mend lit_val_loc;




 %lit_val_loc(target =
             ,target_val_num = N
             ,entire_str =
             ,ignore_case =
             ,inc_sas_def_lib = N
             ,inc_sas_wrk_lib = N
             ,search_lib = "  "
             ,debug = Y
             ,outfilepath =
             ,filename =
             )  ;



proc datasets lib=work kill ;
run;
```