

SAS® Programming Tips for Finding Variables In The DATA Step: A Tutorial

Mark MacMullen
MBNA Hallmark Information Services

Introduction

My introduction to the SAS® language was the Introduction to Base SAS® Software class in 1993 here at MBNA where I work. Except for this I have no formal education in any programming languages. My largest frustration when I started reading external files was primarily the approach I was taking. I was attempting to use the exception isolation technique which was used extensively among the SAS® programmers here at MBNA. When I began to use the similarity isolation technique the variables I needed to find were much easier to select, with less effort. I hope you will find this paper informative.

Abstract

This tutorial is based on my own successful experiences in finding variables defined in the DATA step in an MVS environment, running in batch on a mainframe. Many users who work with external files have pre-defined columns from which to retrieve the variables needed. Most of the time I run batch interface commands from the vendor software to create the reports supplied from the vendor. I route the output from the batch commands to a file. This file becomes my "external" file for my SAS® routine. Using the SAS® language I selectively pull only the necessary records that meet the specific criteria. In most cases these reports have more information than I need and the variables are located within a vast output of data within the file. I will show three methods that I have found to be the most helpful. This paper is not intended to be an exhaustive treatment of the topic but rather a spring board from which to exercise your own inventiveness in the use of the SAS® language.

General Considerations

It is assumed that the reader is familiar with JCL and their site recommendations for building and using PDS and flat files.

Method #1 Building Batch Interface Commands

See figure 1 for a sample of the external file to be read.

The following SAS® code example will read a user updated control card and build a batch command that will be executed in the next step of the JCL:

```
data read1;
  infile indd1 firstobs=8;
  file sb010t;
  input @1 jname $char8. @13 id $char3.;
  if jname = ' ' then do;
    delete;
    return;
  end;
else put @1 'fstruc.job='
        @12 jname
        @20 'schid='
        @27 id;
run;
```

In this first example the INFILE statement is pointing to a user updated control file. The FIRSTOBS= option allows me to place instructions in the user updated control file and still read only the pertinent data I need to build the batch interface command. I prefer to use formatted character lengths for all my input statements combined with pointer control so that there is no guess work as to what I can expect in my search of a file with the INPUT statement. The FILE statement works in conjunction with the PUT statement to create an output file that will be read in the next JCL step of the batch job as a SYSIN DD statement under the control of the program that, in this case, will issue the command to CA7 via the batch interface on the mainframe. The IF statement was used to delete all missing variables not found starting at the end of the job name list to the end of file record. Otherwise the PUT statement places a "blank" job name to account for every position remaining from the last job name to the end of file marker. A sampling of the output results are in figure 2.

Method #2 Selecting Variables From The Results of The Batch Interface Command

Once the commands created above were processed via the batch terminal interface the results were put into a file which I will now read as the next external file in the next step. See figure 3 for a sample of the external file to be read.

Keep in mind two fundamental concepts:

- 1) The Program Data Vector (PDV) Operates on One Observation at a Time.
- 2) The SAS® System Executes Programming Statements in the Exact Order They Are Encountered in the Program.

The following SAS® code example will read two external files in preparation to build another batch interface command for further processing in the next step of the JCL job. The first file to be read will be the sample file in figure 3 and the second file will be the control file from figure 1.

```
data read1;
  infile indd1;
  input @6  jname $char20.
        @42 sl1    $char1.
        @53 sl2    $char1.;
  if sl1 = 'P' & sl2 = 'P' then do;
    jname1 = left(jname);
    jname2 = substr(jname1,1,8);
    drop jname jname1 sl1 sl2;
    output;
  end;
  else delete;
run;

proc sort data=read1 out=read1a nodupkey;
  by jname2;
run;

data read2; set read1a;
  file sb030t;
  put @1  'Ijob,list=depj,job='
      @20 jname2;
run;

data read3;
  infile indd2 firstobs=8;
  file sb030t mod;
  input @45 jname $char8.;
  if jname = ' ' then do;
    delete;
    return;
  end;
  else put @1  'Ijob,list=depj,job='
          @20 jname;
run;
```

In Method #2 I start with the INFILE statement which points me to the processed file results from the batch interface in the previous JCL step. The objective of the INPUT statement is to select only the job names starting in position 6. The job names will only be 8 characters long but the starting position in figure 3 shows that the job names can also begin in positions 8, 10, 12, 14, and 16. However, the last position any job name will occupy will be position 20 as noted by the periods in figure 3. Notice also variables SL1 and SL2, which stand for "slash1" and "slash2". This report output is thousands of records long but I want only the records with the job names. By selecting variables that are consistent in every case with only the job name record I am able to isolate and process only the records needed for this DATA step. The IF statement does the isolation and when those particular records are found I left justify the job names with the jname1 assignment. The SUBSTR function allows me to select the job names leaving behind the periods which were picked up in the character length of 20 for jname. Next I drop the unnecessary variables before the DATA step outputs to the SAS® file work.read1. Close the do loop. When the DATA step encounters any and all other records that do not meet the criteria as coded above, it deletes the record. The data is then sorted by jname2 using the collating sequence option of nodupkey. After the sort is complete I SET the SAS® file work.read1a and using the PUT statement I build the next batch interface command with the job names from the report. In this step there was also another source from which I needed to add job names before moving on to the next JCL step in processing the batch interface command. The user updated control file from method #1 comes into play again. This time the DATA step reads at position 45 to pull a different set of job names. Once the job names are read they too are added to the file sb030t with the MOD option on the FILE statement. A sampling of the output results are in figure 4.

Method #3 Selecting Variables From Multiple Lines In The Order of Their Appearance

See figure 5 for a sample of the external file to be read.

The third SAS® code example was designed due to the following criteria:

- 1) Select the primary job name starting in position 2.
- 2) Select the secondary job names that immediately follow the primary job names if in fact they

exist from starting position 12.

- 3) Make provisions for future sorting of the primary job names while keeping the integrity of the secondary names.

```
data read1;
  infile indd1;
  file temp1;
  retain jname1;
  input @2 jname $char8.
        @42 none $char6.
        @75 sl1 $char1.
        @12 sucr $char40.
        @12 job1 $char4.;

  select;
  when (none = "none" & sl1 = ' ') do;
    drop sl1 none;
    jname1 = jname;
    put @1 jname;
  end;
  when (job1 = 'job=') do;
    drop job1;
    put @1 jname1 @12 sucr;
  end;
otherwise delete;
end;
run;
```

In Method #3 I start with the INFILE statement which points me to the processed file from the batch interface in the previous JCL step. The FILE statement will work in conjunction with the PUT statement found in the SELECT group and the RETAIN is needed in order for the 2nd WHEN statement to successfully add the job name along side of the sucr variable for sorting purposes later in the SAS® routine (not shown). The INPUT statement is self explanatory. I found the SELECT statement to work best in this scenario because after the 1st WHEN statement is true the RETAIN statement contains the value of the last iteration. The 2nd WHEN will now be true each time there is an observation for the secondary job name no matter how many there are. Using the PUT statement to assign the variables to the output file makes it possible for me to build the file in the precise manner in which I need it. If the data from the external file does not meet the criteria as defined in either WHEN statements it is deleted in the OTHERWISE statement of the SELECT group. Once a new observation of the primary job name is found in the 1st WHEN statement the RETAIN of

jname1 is reassigned to the new observation name. A sampling of the output results are in figure 6.

Conclusion

This tutorial presents some of the ways SAS® System users can overcome the challenge of selecting data in an external file from specific locations while entrenched in a sea of less than useful data. Learning the multiple functions and vast choices of SAS® statements ready to be used by the SAS® programmer will make using this language all the more fun. The versatility of the SAS® language is limited only to the user's imagination.

Note:

SAS is a trademark of SAS Institute Inc. in the United States and other countries. ® indicates USA registration.

The author can be contacted at:
MBNA Hallmark Information Services
P.O. Box 809051
Dallas, Texas 75380-9051
972-701-6308

References:

- SAS Institute Inc., SAS® Language: Reference, Version 6, First Edition, Cary, NC: SAS Institute Inc. 1990. 1042 pp.
- SAS Institute Inc., SAS® Procedures Guide, Version 6, Third Edition, Cary, NC: SAS Institute Inc. 1990. 705 pp.
- Cody, Ronald P., and Pass, Raymond, SAS® Programming by Example, Cary, NC: SAS Institute Inc., 1995. 337 pp.
- Aster, Rick, Professional SAS® Programmer's Pocket Reference, Cary, NC: SAS Institute Inc., 1993. 188 pp.
- Karp, Andrew K., (1996). SAS® System Programming Tips and Efficiencies: A Tutorial Proceedings of the Sixth Annual South-Central Regional SAS® User's Group Conference. SCSUG '96 Austin, TX. pp. 119 - 126

>>>>>>> ATTENTION USER <<<<<<<<<

JOBS STARTING IN POSITION 1 ARE TO BE SCHDMODED OFF.

JOBS STARTING IN POSITION 22 ARE SWAPPING ONE SCHID FOR ANOTHER

AND ONLY THE EXCEPTION JOBS FROM THE FSTRUC MATCHING ID=4

WILL BE ADDED TO THE LDEP COMMAND IN STEP 30 OF PCPCONV1.

JOBS STARTING IN POSITION 45 ARE TO BE SET EXEC=N.

```
-----1-----2-----3-----4-----5-----6-----7
CII4CLO1 ID=008      IWP601DJ ID=001      CWP300W3 ID=051
CVBTRAN1 ID=001      IWP601DJ ID=004      SRP390W0 ID=004
CVB008DJ ID=008      W904614B ID=004
CVP001W3 ID=008      CWP240W3 ID=004
CVP004DJ ID=004
CVP008DJ ID=008
CVP230W3 ID=008
CVP340W3 ID=004
CVP384W1 ID=008
CVSCHDYR ID=108
CVSYSTEM1 ID=008
CV01CLO1 ID=008
CV01CLO4 ID=008
CWP400W4 ID=008
FMZ699W1 ID=108
IWP640DJ ID=004
PO19CLO1 ID=008
PRP609W2 ID=004
SAP413W3 ID=001
TSP001DJ ID=004
VISCHDYR ID=104
NOSCHDY1 ID=004
SMP039W2 ID=004
AMP600W0 ID=004
CUSCHDYR ID=004
```

Figure #1 - Method #1

FSTRUC, JOB=CII4CLO1, SCHID=008
FSTRUC, JOB=CVBTRAN1, SCHID=001
FSTRUC, JOB=CVB008DJ, SCHID=008
FSTRUC, JOB=CVP001W3, SCHID=008
FSTRUC, JOB=CVP004DJ, SCHID=004
FSTRUC, JOB=CVP008DJ, SCHID=008
FSTRUC, JOB=CVP230W3, SCHID=008
FSTRUC, JOB=CVP340W3, SCHID=004
FSTRUC, JOB=CVP384W1, SCHID=008
FSTRUC, JOB=CVSCHDYR, SCHID=108
FSTRUC, JOB=CVSYSTEM1, SCHID=008
FSTRUC, JOB=CV01CLO1, SCHID=008
FSTRUC, JOB=CV01CLO4, SCHID=008
FSTRUC, JOB=CWP400W4, SCHID=008
FSTRUC, JOB=FMZ699W1, SCHID=108
FSTRUC, JOB=IWP640DJ, SCHID=004
FSTRUC, JOB=PO19CLO1, SCHID=008
FSTRUC, JOB=PRP609W2, SCHID=004
FSTRUC, JOB=SAP413W3, SCHID=001
FSTRUC, JOB=TSP001DJ, SCHID=004
FSTRUC, JOB=VISCHDYR, SCHID=104
FSTRUC, JOB=NOSCHDY1, SCHID=004
FSTRUC, JOB=SMP039W2, SCHID=004
FSTRUC, JOB=AMP600W0, SCHID=004
FSTRUC, JOB=CUSCHDYR, SCHID=004

Figure #2 - Method #1

1BSTR-00 *** INPUT TO BATCH TERMINAL ***

/LOGON ***** * GENERATED LOGON *
FSTRUC,JOB=IWP601DJ,SCHID=001
FSTRUC,JOB=IWP601DJ,SCHID=004
/LOGOFF * GENERATED LOGOFF *
OBSTR-00 *** END OF INPUT ***
1 97129 175755 /LOGON *****

CA-7.023 - V3L1 (9505) OPERATOR IS LOGGED ON TO TERMINAL BTT1 AT 17:57:55 ON 97.129
- 97129 175755 FSTRUC,JOB=IWP601DJ,SCHID=001
1FSTRUC DATE 05-09-97 PAGE 0001

FORECAST FOR CA-7 JOBS

START TIME : 05-09-97 AT 1757 HRS

JOB(S) : IWP601DJ

SYSTEM(S) : ALL JOBNET(S) : ALL

OPTIONS : BOTH JOB- AND DSN-TRIGGERED JOBS INCLUDED
CONNECTED OUTPUT NETWORKS NOT INCLUDED

HIGHEST JOB DATE AND TIME : 97130/0010
HIGHEST JOB NAME : PCP700D2

1FSTRUC NETWORK STRUCTURE FOR CA-7 JOBS DATE 05-09-97 PAGE 0001

START TIME : 05-09-97 AT 1757 HRS

LEV#	JOB NAME	SYS	START DTTM	END DTTM	TRIGGERING JOB/DSN/SID
***	IWP601DJ.....		97129/1757	97129/1757	
001	IWP601D2.....		97129/1757	97129/1822	IWP601DJ :001
002	IWP917D2.....		97129/2042	97129/2053	IWP601D2 :001
001	IWP650D2.....		97129/1757	97129/1825	IWP601DJ :001
002	AAP216D2.....		97129/1825	97129/1826	IWP650D2 :001
002	CVP615D2.....		97129/1825	97129/1836	IWP650D2 :001
003	IWP615D2.....		97129/1836	97129/1837	CVP615D2 :001
002	D907414S.....		97129/1825	97129/1845	IWP650D2 :001
002	FMP605D1.....		97129/2045	97129/2123	IWP650D2 :001
002	IKP300D2.....		97129/1905	97129/1940	IWP650D2 :001
003	DMPMC104.....		97129/1940	97129/1941	IKP300D2 :001
003	ICP070D3.....		97129/1950	97129/1951	IKP300D2 :001
002	IWP602D2.....		97129/1835	97129/1836	IWP650D2 :001
003	IWP603D2.....		97129/1856	97129/1900	IWP602D2 :001
004	PRP469D3.....		97129/1930	97129/1930	IWP603D2 :001
002	IWP650#2.....		97129/1825	97129/1825	IWP650D2 :001
002	IWP651D2.....		97129/1825	97129/1859	IWP650D2 :001
002	IWP652D2.....		97129/1825	97129/1826	IWP650D2 :001
002	IWP660D2.....		97129/1835	97129/1836	IWP650D2 :001
002	IWP918D2.....		97129/1825	97129/1833	IWP650D2 :001
002	IW650BAL.....		97129/2025	97129/2025	IWP650D2 :001
002	MT01CLO1.....		97129/1835	97129/1836	IWP650D2 :001
003	MTP000D1.....		97129/1836	97129/1841	MT01CLO1 :001
004	MT01OPE1.....		97129/1841	97129/1842	MTP000D1 :001
005	MTP005D2..		97129/1842	97129/1856	MT01OPE1 :001
005	MTP006D2..		97129/1842	97129/1846	MT01OPE1 :001
002	PCP105D2.....		97129/1825	97129/1839	IWP650D2 :001
002	PCP380D2.....		97129/1925	97129/1935	IWP650D2 :001
002	PCP611D2.....		97129/2025	97129/2029	IWP650D2 :001
002	PEP410D2.....		97129/1835	97129/1839	IWP650D2 :001
003	NJPACHLB.....		97129/2039	97129/2039	PEP410D2 :001

Figure #3 - Method #2

LJOB,LIST=DEPJ,JOB=VII510W4
 LJOB,LIST=DEPJ,JOB=VII512W4
 LJOB,LIST=DEPJ,JOB=VII520W4
 LJOB,LIST=DEPJ,JOB=VIJ510W4
 LJOB,LIST=DEPJ,JOB=VIJ512W4
 LJOB,LIST=DEPJ,JOB=VIJ520W4
 LJOB,LIST=DEPJ,JOB=VIK510W4
 LJOB,LIST=DEPJ,JOB=VIK512W4
 LJOB,LIST=DEPJ,JOB=VIK520W4
 LJOB,LIST=DEPJ,JOB=VIL510W4
 LJOB,LIST=DEPJ,JOB=VIL512W4
 LJOB,LIST=DEPJ,JOB=VIL520W4
 LJOB,LIST=DEPJ,JOB=VIM510W4
 LJOB,LIST=DEPJ,JOB=VIM512W4
 LJOB,LIST=DEPJ,JOB=VIM520W4
 LJOB,LIST=DEPJ,JOB=VIN510W4
 LJOB,LIST=DEPJ,JOB=VIN512W4
 LJOB,LIST=DEPJ,JOB=VIN520W4
 LJOB,LIST=DEPJ,JOB=VIO510W4
 LJOB,LIST=DEPJ,JOB=VIO512W4
 LJOB,LIST=DEPJ,JOB=VIO520W4
 LJOB,LIST=DEPJ,JOB=VIP510W4
 LJOB,LIST=DEPJ,JOB=VIP512W4
 LJOB,LIST=DEPJ,JOB=VIP513W4
 LJOB,LIST=DEPJ,JOB=VIP520W4
 LJOB,LIST=DEPJ,JOB=VIQ510W4
 LJOB,LIST=DEPJ,JOB=VIQ512W4
 LJOB,LIST=DEPJ,JOB=VIQ520W4
 LJOB,LIST=DEPJ,JOB=VIR510W4
 LJOB,LIST=DEPJ,JOB=VIR512W4
 LJOB,LIST=DEPJ,JOB=VIR520W4
 LJOB,LIST=DEPJ,JOB=VISCHDYR
 LJOB,LIST=DEPJ,JOB=VIS510W4
 LJOB,LIST=DEPJ,JOB=VIS512W4
 LJOB,LIST=DEPJ,JOB=VIS520W4
 LJOB,LIST=DEPJ,JOB=VIZ511W4
 LJOB,LIST=DEPJ,JOB=VI05CLO3
 LJOB,LIST=DEPJ,JOB=VI05OPE3
 LJOB,LIST=DEPJ,JOB=W906821S
 LJOB,LIST=DEPJ,JOB=CWP300W3
 LJOB,LIST=DEPJ,JOB=SRP390W0
 LJOB,LIST=DEPJ,JOB=PTP225W2
 LJOB,LIST=DEPJ,JOB=CWP240W3
 LJOB,LIST=DEPJ,JOB=IK#300D2
 LJOB,LIST=DEPJ,JOB=IWP64#RQ
 LJOB,LIST=DEPJ,JOB=IWP640W2
 LJOB,LIST=DEPJ,JOB=IWP641W2
 LJOB,LIST=DEPJ,JOB=IWP642W2
 LJOB,LIST=DEPJ,JOB=IWP643W2
 LJOB,LIST=DEPJ,JOB=IWP644W2
 LJOB,LIST=DEPJ,JOB=IWP649W2
 LJOB,LIST=DEPJ,JOB=IWP650W2
 LJOB,LIST=DEPJ,JOB=IWP656W2
 LJOB,LIST=DEPJ,JOB=PEP409W2

Figure #4 - Method #2

1BSTR-00 *** INPUT TO BATCH TERMINAL ***

```
/LOGON ***** * GENERATED LOGON *
LJOB,LIST=DEPJ,JOB=IWP641W2
LJOB,LIST=DEPJ,JOB=IWP642W2
LJOB,LIST=DEPJ,JOB=IWP643W2
LJOB,LIST=DEPJ,JOB=IWP644W2
LJOB,LIST=DEPJ,JOB=IWP649W2
LJOB,LIST=DEPJ,JOB=IWP650W2
LJOB,LIST=DEPJ,JOB=IWP656W2
LJOB,LIST=DEPJ,JOB=PEP409W2
/LOGOFF * GENERATED LOGOFF *
OBSTR-00 *** END OF INPUT ***
1 97129 175815 /LOGON *****
```

CA-7.023 - V3L1 (9505) OPERATOR IS LOGGED ON TO TERMINAL BTT1 AT 17:58:15 ON 97.129
- 97129 175815 LJOB,LIST=DEPJ,JOB=AMP600W0
1LJOB,LIST=DEPJ,JOB=AMP600W0
JOB=AMP600W0 LIST=DEPJ DATE=97.129 PAGE 0001

JOB NAME	----JCL----	SYSTEM -NAME-	USR -ID	MAIN -ID	PROSE -ID-	SCHED DSNBR	--NUMBER OF- STP DDS RUNS	LAST-RUN DATE/TIME
-------------	-------------	------------------	------------	-------------	---------------	----------------	------------------------------	-----------------------

AMP600W0 000 AMP600W0 010 ALL *NONE* 000278 004 017 0028 97123/0744

SLIA-00 REQUEST COMPLETED AT 17:58:15 ON 97.129
- 97129 175816 LJOB,LIST=DEPJ,JOB=AMP955W0
1LJOB,LIST=DEPJ,JOB=AMP955W0
JOB=AMP955W0 LIST=DEPJ DATE=97.129 PAGE 0001

JOB NAME	----JCL----	SYSTEM -NAME-	USR -ID	MAIN -ID	PROSE -ID-	SCHED DSNBR	--NUMBER OF- STP DDS RUNS	LAST-RUN DATE/TIME
-------------	-------------	------------------	------------	-------------	---------------	----------------	------------------------------	-----------------------

AMP955W0 000 AMP955W0 010 ALL *NONE* *NONE* 001 016 0012 97123/1700

----- SUCCESSOR JOBS -----
JOB=/CVP424W1 SCHID=000
JOB=AMP970D0 SCHID=020

SLIA-00 REQUEST COMPLETED AT 17:58:16 ON 97.129
- 97129 175816 LJOB,LIST=DEPJ,JOB=ASP030W1
1LJOB,LIST=DEPJ,JOB=ASP030W1
JOB=ASP030W1 LIST=DEPJ DATE=97.129 PAGE 0001

JOB NAME	----JCL----	SYSTEM -NAME-	USR -ID	MAIN -ID	PROSE -ID-	SCHED DSNBR	--NUMBER OF- STP DDS RUNS	LAST-RUN DATE/TIME
-------------	-------------	------------------	------------	-------------	---------------	----------------	------------------------------	-----------------------

ASP030W1 000 ASP030W1 010 ALL *NONE* *NONE* 010 097 0331 97123/1037

----- SUCCESSOR JOBS -----
JOB=/ASP860D3 SCHID=000
JOB=ASP020S1 SCHID=004
JOB=AS010PE1 SCHID=008

SLIA-00 REQUEST COMPLETED AT 17:58:16 ON 97.129
- 97129 175816 LJOB,LIST=DEPJ,JOB=ASP941W1
1LJOB,LIST=DEPJ,JOB=ASP941W1
JOB=ASP941W1 LIST=DEPJ DATE=97.129 PAGE 0001

JOB NAME	----JCL----	SYSTEM -NAME-	USR -ID	MAIN -ID	PROSE -ID-	SCHED DSNBR	--NUMBER OF- STP DDS RUNS	LAST-RUN DATE/TIME
-------------	-------------	------------------	------------	-------------	---------------	----------------	------------------------------	-----------------------

FIGURE #5 - Method #3

AMP600W0
AMP955W0
AMP955W0 JOB=/CVP424W1 SCHID=000
AMP955W0 JOB=AMP970D0 SCHID=020
ASP030W1
ASP030W1 JOB=/ASP860D3 SCHID=000
ASP030W1 JOB=ASP020S1 SCHID=004
ASP030W1 JOB=AS01OPE1 SCHID=008

Figure #6 - Method #3