Paper SA-07 2014

How Do I …. ? Some Beginners FAQs

Peter Eberhardt, Fernwood Consulting Group Inc. Toronto, Canada
Audrey Yeo, Athene USA, West Des Moines, IA

ABSTRACT

No matter how long you use SAS® you find you still have questions on how to do things in SAS. What you forget is the number of questions you had to overcome when you first started with SAS. If you were lucky, you had a mentor at your place of work or school who helped you with those questions; if you were not so lucky you struggled with a myriad of sources trying to get help. In this paper, we will highlight some of the questions we often get from new SAS users, and offer some answers.

INTRODUCTION

There are several common problems that new, and sometimes more experienced, SAS programmers face. On one hand, given the scope of the SAS system and the vast variety of applications for which it is used, this paper will not even scratch the surface of problems. On the other hand, there are a few common questions we see that would cover many uses of SAS. In this paper we will introduce some that we commonly get from our colleagues. The questions are broken into two basic categories

1. SAS environment
2. Programming

What we are providing are answers based on our experience. With SAS, there are normally many ways to solve the same problem; we are providing solutions we found effective.

We will start with the environment questions.

HOW DO I??

HOW DO I USE A DIFFERENT SAS WORK FOLDER?

You might want to change the SAS WORK folder because of space limitations on the drive, or because a faster drive is available; since the SAS WORK folder is accessed constantly during a SAS session, using the fastest drive available is a good strategy, The SAS WORK folder location is set when SAS is installed; on a Windows machine this is in the temp folder defined by Windows and specified in the Windows environmental variable TEMP (usually C:\Users\{username}\AppData\Local\Temp). Of course the best way to change the location of the folder is to specify the location during the install, but since you are asking the question, it is too late for this.

Option 1 – Change the Configuration File

This is the permanent solution, however there are always risks with changing the configuration file. Once the configuration file is changed, all SAS sessions will use the new location, unless there is a command line override. The first thing you need to do is locate the configuration file; for SAS v9 it is called sasv9.cfg and by default is found in folder C:\Program Files\SASHome\SASFoundation\9.x\nls\en (where 9.x is the SAS release). You can verify this by checking the SAS shortcut on the Start menu; you will see something like this (but all on one line)

```
"C:\Program Files\SASHome\SASFoundation\9.4\sas.exe"
 -CONFIG "C:\Program Files\SASHome\SASFoundation\9.4\nls\en\sasv9.cfg"
```

If you are using SAS with a non-English language, then the file will be in the folder with the 2 character ISO abbreviation (*en* is the ISO English language abbreviation).

BEFORE YOU MAKE ANY CHANGES TO THE CONFIGURATION FILE, MAKE A BACKUP COPY

The first challenge is dealing with the dire warning:

```
/* DO NOT EDIT BELOW THIS LINE - INSTALL Application edits below this line */
```

since the line that specifies the location of the WORK library is located below this line. In the configuration file the following line sets the location of the WORK folder:

```
-WORK "!TEMP\SAS Temporary Files"
```

Note the use of the environmental variable !TEMP (actually TEMP is the environmental variable name and the exclamation point indicates that TEMP is not a literal but a variable). Since the location is set using a variable, we can simply override the value of the variable. To do this, add the following lines BEFORE the dire warning:

```
/* this will override the windows TEMP environmental variable */
-SET TEMP    "C:\WORK"
```

Where C:\WORK would be replaced by the folder you want to use. Now, every time you start SAS from the Start menu shortcut, or any shortcut created from it, you will use the new location.

Option 2 – Create a new Config File

This option is essentially the same as above, except:

1. You use a different name and/or path for the config file.
2. You change the shortcut to specify the new config file.

The advantage of this solution is that you can add project specific SAS options in addition to the WORK location to the config file; in this way, each project can have a custom set of SAS options. If the config file is stored in D:\projects\risk\config\sasv7.cfg, then the shortcut would look like:

```
"C:\Program Files\SASHome\SASFoundation\9.4\sas.exe" -CONFIG " D:\projects\risk\config\sasv7.cfg"
```

Option 3 – Command Line Override

In this option we simply add the ***–WORK*** option to the command line specified in the shortcut (all one line):

```
"C:\Program Files\SASHome\SASFoundation\9.4\sas.exe"
-CONFIG "C:\Program Files\SASHome\SASFoundation\9.4\nls\en\sasv9.cfg"
-WORK "C:\WORK\PETER\SASWORK"
```

In this example we are using the default config file, but specifying the location of the WORK folder to be C:\WORK\PETER\SASWORK.

IN SAS, WHEN I INITIALLY OPEN A PROGRAM IT OPENS THE ‘MY DOCUMENTS\MY SAS FILES\9.4’ FOLDER. HOW DO I GET SAS TO DEFAULT TO ANOTHER LOCATION?

If you do not save your files to the ***My Documents\My SAS Files*** folder, it is a nuisance to have to navigate to the folder where you save your files. Of course, once you navigate to the folder and open a file all further open program dialogues start at this folder.

Option 1 – Change interactively

In your SAS session you can change this folder interactively in two ways:

1. Using the SAS menu Tools… Options… Change Current Folder
2. Double click on the Current Folder icon in the Status bar

In both cases you navigate to the desired folder. Although these both work, this is essentially the same thing you do the first time you open a SAS program, so there is no real advantage.

Option 2– Command Line Override

In this option we simply add the ***–SASINITIALFOLDER*** option to the command line specified in the shortcut (all one line):

```
"C:\Program Files\SASHome\SASFoundation\9.4\sas.exe"
-CONFIG "C:\Program Files\SASHome\SASFoundation\9.4\nls\en\sasv9.cfg"
-SASINITIALFOLDER "C:\PROJECT\SASPGMS"
```

In this example we are using the default CONFIG file, but specifying the location of the SASINITIALFOLDER folder to be C:\PROJECT\SASPGMS. By setting up different shortcuts for each project, each project can then automatically open its own program files. Of course you can couple this with a custom CONFIG file to fully customize each project.

I HAD TO FORCE MY SAS SESSION TO TERMINATE. IS THERE ANY WAY I CAN RECOVER THE TABLES IN THE WORK FOLDER, IT TOOK HOURS TO CREATE THEM?

You should be able to recover all the tables except those that SAS was writing to when you forced SAS to shut down; you need to:

1. Locate the WORK folder (see above)
2. Open the most recent work folder. Unless you have several SAS sessions open at once, this should be the folder from the session you forced closed; if you have multiple SAS sessions open then you will have to browse through all the recent WORK folders to find the ‘lost’ folder. If you cannot find the folder then the tables are lost
3. Start a new SAS session
4. A new WORK folder is created, in a new Windows Explorer window navigate to this folder.
5. Move the tables from the ‘lost’ folder to this new folder.
6. In SAS, open the WORK library and verify the tables are not corrupt.

Corollary:

When SAS shuts down normally it will clean up the WORK library, however there may be instances where the WORK folder is not completely removed. From time to time you should check the SAS WORK folder for session tables that were not removed; any ‘orphaned’ folders should be removed since they are needlessly using space and ultimately degrading performance.

I HAD TO FORCE MY SAS SESSION TO TERMINATE. IS THERE ANY WAY I CAN RECOVER THE SAS PROGRAMS I HAD OPEN BUT DID NOT SAVE?

This is a definite maybe. SAS can create auto save files at specified intervals, the default being every 10 minutes; depending upon when the changes are made and the auto save interval, you may not recover all of your changes.

SAS auto save files follow the naming convention:

AutoSaved of {program name}.$AS

The default location is:

C:\Users\{user id}\AppData\Roaming\SAS\EnhancedEditor

Where {program name} is the name if the SAS file you had opened. If you have editor sessions opened that have not been saved the {program name} will be ***Untitled***. If you are making many edits to your program AND you expect you may have to force SAS to close then you should set the auto save interval to a small value (say 1 minute), or explicitly save your work to minimize possible loss of work.

Corollary:

When SAS shuts down normally it will clean up the auto save files, however there may be instances where the auto save files are not completely removed. From time to time you should check for and remove old auto save files.

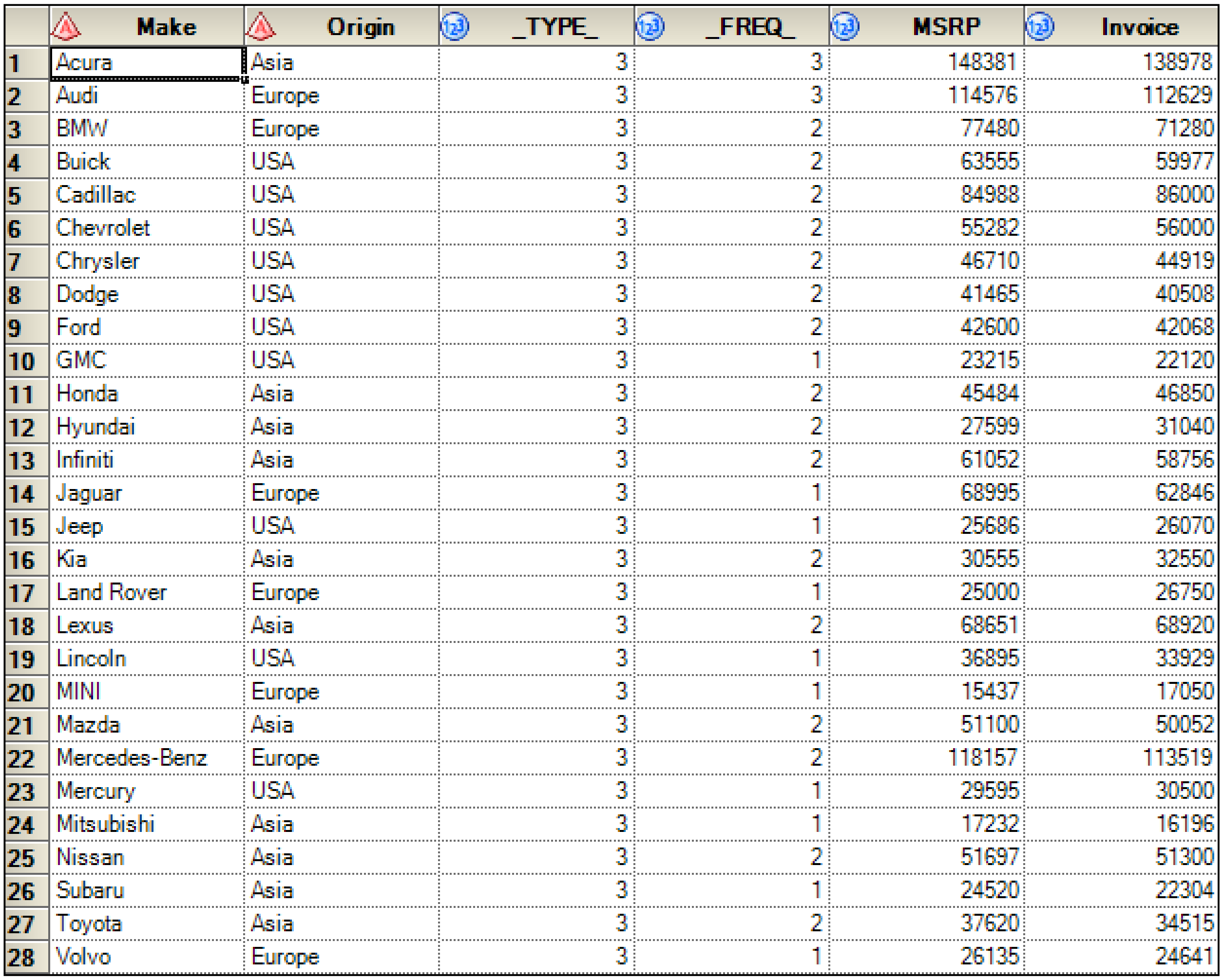
I HAVE A DATASET CALLED CARS AND IT CONSISTS OF THE MAKE, MODEL, TYPE, ORIGIN, MSRP, AND INVOICE. I WANT TO SUMMARIZE THE MSRP AND INVOICE BY ITS MAKE AND ORIGIN, AS SHOWN BELOW. HOW DO I SUMMARIZE DATA AT MORE THAN ONE LEVEL?

Code:

```
 data cars;
      input Make $ 1-13 Model $ 15-46 Origin $ 47-53 MSRP 54-59 Invoice;
      datalines;
Acura         MDX                             Asia   36945 37000
Acura         TL 4dr                          Asia   21671 22000
Acura         NSX coupe 2dr manual S          Asia   89765 79978
Audi          A4 3.0 4dr                      Europe 28846 29500
Audi          A6 3.0 4dr                      Europe 36640 33129
Audi          S4 Avant Quattro                Europe 49090 50000
BMW           530i 4dr                        Europe 44995 41170
BMW           325xi Sport                     Europe 32485 30110
Buick         Regal LS 4dr                    USA    22835 23050
Buick         Park Avenue Ultra 4dr           USA    40720 36927
Cadillac      SRX V8                          USA    43523 44000
Cadillac      Tahoe LT                        USA    41465 42000
Chevrolet     Tahoe LT                        USA    36287 35000
Chevrolet     Malibu 4dr                      USA    18995 21000
Chrysler      PT Cruiser 4dr                  USA    17985 16919
Chrysler      Pacifica                        USA    28725 28000
Dodge         Caravan SE                      USA    21250 20508
Dodge         Ram 1500 Regular Cab ST         USA    20215 20000
Ford          Escape XLS                      USA    15670 17570
Ford          Freestar SE                     USA    26930 24498
GMC           Safari SLE                      USA    23215 22120
Honda         Pilot LX                        Asia   27560 27000
Honda         Accord LX 2dr                   Asia   17924 19850
Hyundai       Accent GL 4dr                   Asia   11839 13450
Hyundai       XG350 4dr                       Asia   15760 17590
Infiniti      G35 4dr                         Asia   26157 27000
Infiniti      FX35                            Asia   34895 31756
Jaguar        Vanden Plas 4dr                 Europe 68995 62846
Jeep          Grand Cherokee Laredo           USA    25686 26070
Kia           Rio 4dr auto                    Asia   11155 12050
Kia           Sedona LX                       Asia   19400 20500
Land Rover    Discovery SE                    Europe 25000 26750
Lexus         GS 300 4dr                      Asia   36196 37470
Lexus         IS 300 SportCross               Asia   32455 31450
Lincoln       LS V6 Premium 4dr               USA    36895 33929
MINI          Cooper                          Europe 15437 17050
Mazda         MPV ES                          Asia   28750 29570
Mazda         B4000 SE Cab Plus               Asia   22350 20482
Mercedes-Benz C240 4dr                        Europe 31187 32580
Mercedes-Benz S500 4dr                        Europe 86970 80939
Mercury       Grand Marquis LS Premium 4dr    USA    29595 30500
Mitsubishi    Lancer OZ Rally 4dr auto        Asia   17232 16196
Nissan        Quest S                         Asia   22958 24000
Nissan        Murano SL                       Asia   28739 27300
Subaru        Baja                            Asia   24520 22304
Toyota        Sequoia SR5                     Asia   24050 21450
Toyota        Corolla CE 4dr                  Asia   13570 13065
```

```
Volvo         V40                              Europe 26135 24641
;
run;

proc summary data=cars nway;
      class Make Origin;
      var MSRP Invoice;
      output out=Cars_sum sum=;
run;
```

Output Dataset:



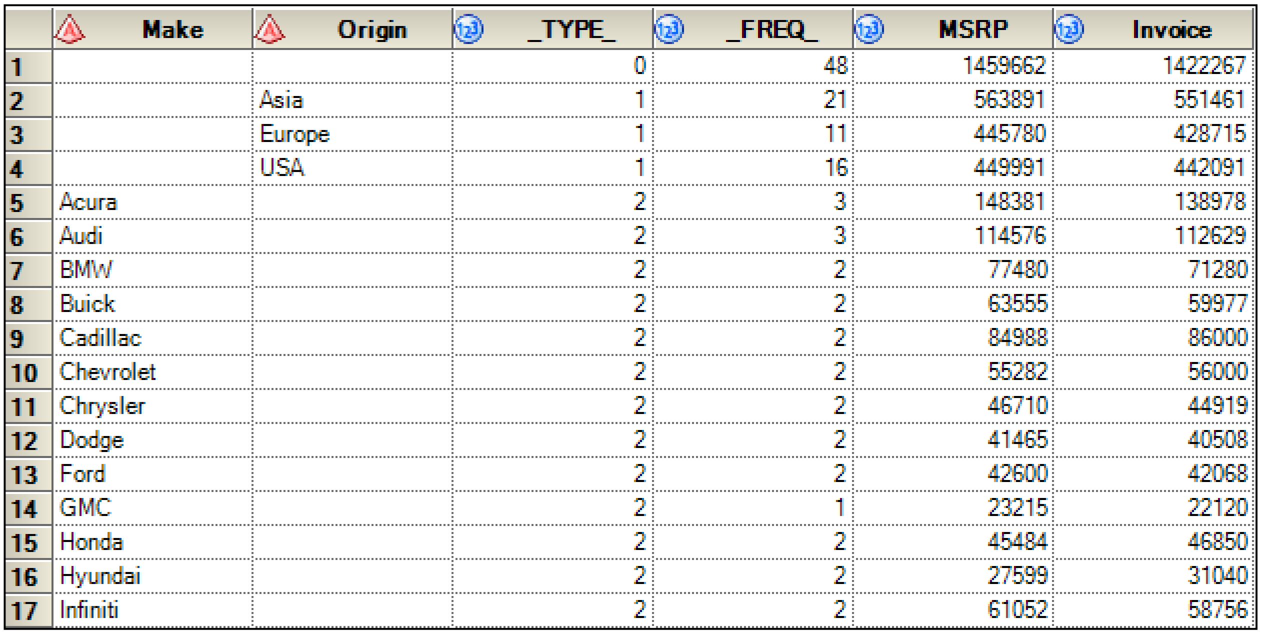
	Make	Origin	_TYPE_	_FREQ_	MSRP	Invoice
1	Acura	Asia	3	3	148381	138978
2	Audi	Europe	3	3	114576	112629
3	BMW	Europe	3	2	77480	71280
4	Buick	USA	3	2	63555	59977
5	Cadillac	USA	3	2	84988	86000
6	Chevrolet	USA	3	2	55282	56000
7	Chrysler	USA	3	2	46710	44919
8	Dodge	USA	3	2	41465	40508
9	Ford	USA	3	2	42600	42068
10	GMC	USA	3	1	23215	22120
11	Honda	Asia	3	2	45484	46850
12	Hyundai	Asia	3	2	27599	31040
13	Infiniti	Asia	3	2	61052	58756
14	Jaguar	Europe	3	1	68995	62846
15	Jeep	USA	3	1	25686	26070
16	Kia	Asia	3	2	30555	32550
17	Land Rover	Europe	3	1	25000	26750
18	Lexus	Asia	3	2	68651	68920
19	Lincoln	USA	3	1	36895	33929
20	MINI	Europe	3	1	15437	17050
21	Mazda	Asia	3	2	51100	50052
22	Mercedes-Benz	Europe	3	2	118157	113519
23	Mercury	USA	3	1	29595	30500
24	Mitsubishi	Asia	3	1	17232	16196
25	Nissan	Asia	3	2	51697	51300
26	Subaru	Asia	3	1	24520	22304
27	Toyota	Asia	3	2	37620	34515
28	Volvo	Europe	3	1	26135	24641

In the output above, we see that the MSRP and Invoice are summed up by Make and Origin. However, what I want is the sum of MSRP and Invoice by Make, and the sum of MSRP and Invoice by Origin. In order to do this, we need to remove the *NWAY* option from the code.

Code:

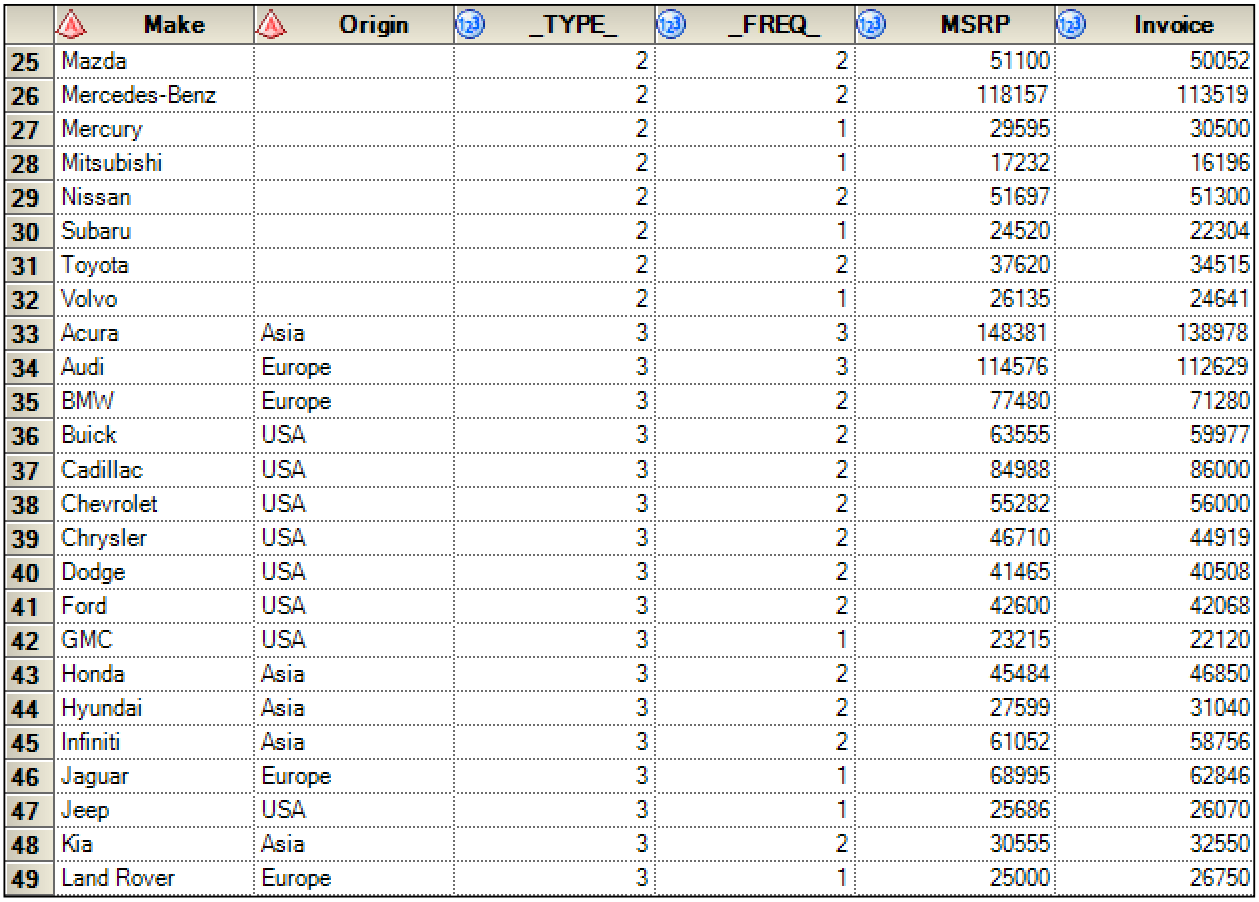
```
proc summary data=cars;
      class Make Origin;
      var MSRP Invoice;
      output out=Cars_sum sum=;
run;
```

Output Dataset (subset 1):



	Make	Origin	_TYPE_	_FREQ_	MSRP	Invoice
1			0	48	1459662	1422267
2		Asia	1	21	563891	551461
3		Europe	1	11	445780	428715
4		USA	1	16	449991	442091
5	Acura		2	3	148381	138978
6	Audi		2	3	114576	112629
7	BMW		2	2	77480	71280
8	Buick		2	2	63555	59977
9	Cadillac		2	2	84988	86000
10	Chevrolet		2	2	55282	56000
11	Chrysler		2	2	46710	44919
12	Dodge		2	2	41465	40508
13	Ford		2	2	42600	42068
14	GMC		2	1	23215	22120
15	Honda		2	2	45484	46850
16	Hyundai		2	2	27599	31040
17	Infiniti		2	2	61052	58756

Output Dataset (subset 2):



	Make	Origin	_TYPE_	_FREQ_	MSRP	Invoice
25	Mazda		2	2	51100	50052
26	Mercedes-Benz		2	2	118157	113519
27	Mercury		2	1	29595	30500
28	Mitsubishi		2	1	17232	16196
29	Nissan		2	2	51697	51300
30	Subaru		2	1	24520	22304
31	Toyota		2	2	37620	34515
32	Volvo		2	1	26135	24641
33	Acura	Asia	3	3	148381	138978
34	Audi	Europe	3	3	114576	112629
35	BMW	Europe	3	2	77480	71280
36	Buick	USA	3	2	63555	59977
37	Cadillac	USA	3	2	84988	86000
38	Chevrolet	USA	3	2	55282	56000
39	Chrysler	USA	3	2	46710	44919
40	Dodge	USA	3	2	41465	40508
41	Ford	USA	3	2	42600	42068
42	GMC	USA	3	1	23215	22120
43	Honda	Asia	3	2	45484	46850
44	Hyundai	Asia	3	2	27599	31040
45	Infiniti	Asia	3	2	61052	58756
46	Jaguar	Europe	3	1	68995	62846
47	Jeep	USA	3	1	25686	26070
48	Kia	Asia	3	2	30555	32550
49	Land Rover	Europe	3	1	25000	26750

If you notice in the first dataset output, there is a column called _TYPE_, and it has a number 3 in this type. The *NWAY* option in the PROC SUMMARY code tells SAS to output only rows with a combination of the variables in the *CLASS* statement, which in this case, is the Make and Origin.

By removing the *NWAY* option from the PROC SUMMARY, this tells SAS to output all combinations of the variables in the *CLASS* statement. This means, without the *NWAY* option, SAS will output the grand summed by total (regardless of by Make or Origin), which has _TYPE_ = 0. Next SAS will output the total summed the last variable in the *CLASS* statement, which in this case, is the Origin, and assigns that as _TYPE_ = 1. This is then followed by _TYPE_ = 2, which is the sum of MSRP and Invoice by Make. Finally, we see _TYPE_ = 3, which is the sum of MSRP and Invoice by Make and Origin.

So, to sum the data by more than one level, using PROC SUMMARY, all I need to do is to remove the *NWAY* option from PROC SUMMARY and I have what I'm looking for (_TYPE_ = 1 and _TYPE_ = 2).

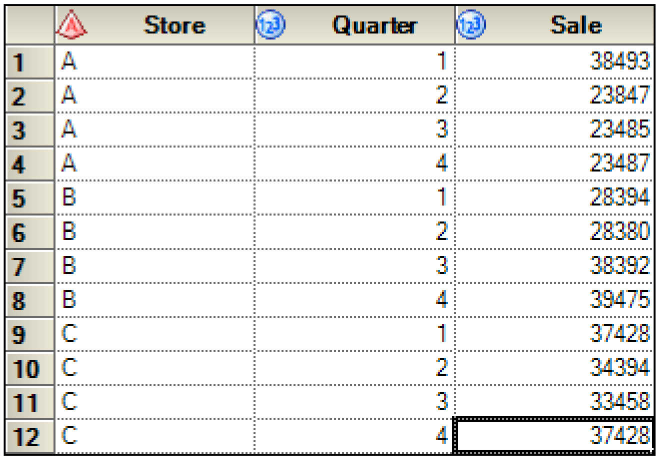
HOW DO I DETERMINE IF THE LAST VALUE IS SMALLER THAN THE FIRST VALUE IN A GROUP?

I have a dataset called sales and I want to compare and see if the last quarter sale has improved compared to the first quarter sale for store A, store B, and store C.

Code:

```
data sales;
      input Store $ 1 Quarter 3 Sale;
      datalines;
A 1 38493
A 2 23847
A 3 23485
A 4 23487
B 1 28394
B 2 28380
B 3 38392
B 4 39475
C 1 37428
C 2 34394
C 3 33458
C 4 37428
;
run;
```

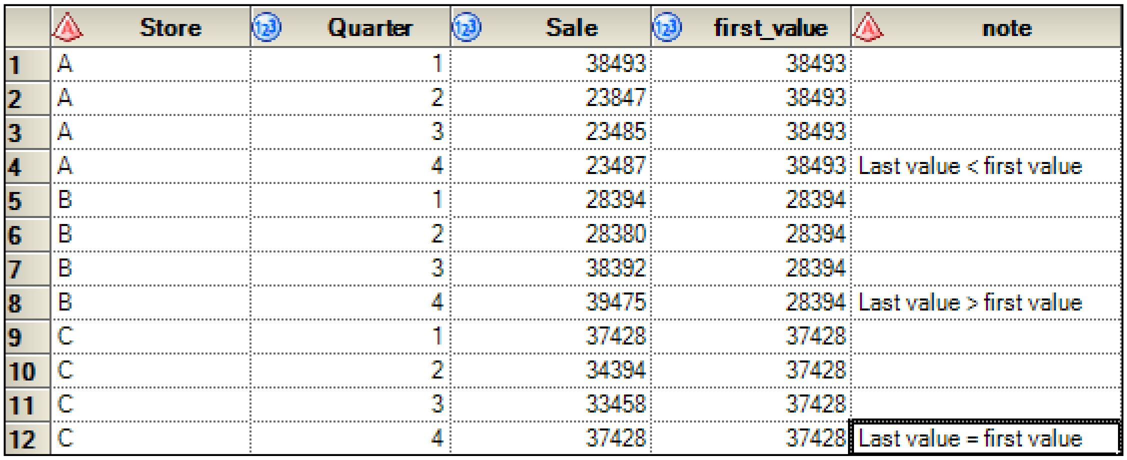
Dataset:



	Store	Quarter	Sale
1	A	1	38493
2	A	2	23847
3	A	3	23485
4	A	4	23487
5	B	1	28394
6	B	2	28380
7	B	3	38392
8	B	4	39475
9	C	1	37428
10	C	2	34394
11	C	3	33458
12	C	4	37428

```
data sales2;
      set sales;
      retain first_value;
      format note $25.;
      by Store;
      if first.store then first_value = sale;
      if last.store then do;
         if sale < first_value then note = "Last value < first value";
          else if sale > first_value then note = "Last value > first value";
          else note = "Last value = first value";
      end;
run;
```

Output Dataset:



	Store	Quarter	Sale	first_value	note
1	A	1	38493	38493	
2	A	2	23847	38493	
3	A	3	23485	38493	
4	A	4	23487	38493	Last value < first value
5	B	1	28394	28394	
6	B	2	28380	28394	
7	B	3	38392	28394	
8	B	4	39475	28394	Last value > first value
9	C	1	37428	37428	
10	C	2	34394	37428	
11	C	3	33458	37428	
12	C	4	37428	37428	Last value = first value

The *RETAIN* statement in the code tells SAS to keep the value that it was assigned to across the observations. In addition, we used the *FIRST.*var and *LAST.*var to keep track of whether we are accessing the first or the last observation. In order to *FIRST.*var and *LAST.*var, we must first make sure that the variables are sorted. We also need a *BY* statement in order to use *FIRST.*var and *LAST.*var. The *BY* statement creates the temporary variables: *FIRST.*var and *LAST.*var.

In this example, we are comparing the first and the last quarter's sale for stores A, B, and C. We see in the dataset above that stores and quarters are sorted in ascending order.

Next, we use the *RETAIN* statement to tell SAS that we want to retain the variable first_value. If *FIRST.*store equals to 1 then we set the variable first_value equals the sale. If *FIRST.*store does not equal to 1, the *RETAIN* statement will keep the original value that was assigned to it.

On the 1st iteration, when SAS is processing the 1st quarter for store A, *FIRST.*store will be equal to 1 and SAS will assign the value in sale to the variable first_value. *LAST.*var will be equal to 0 and SAS will not do anything.

On the 2nd iteration, both *FIRST.*store and *LAST.*store will be equal to 0 and SAS will not do anything. However, because of the *RETAIN* statement, SAS will retain the value in sale and assign it to first_value. The same thing happens on the 3rd iteration.

On the 4th iteration, when SAS is processing the 4th quarter for store A, *FIRST.*store will be equal to 0 but *LAST.*store will then be equal to 1. Since *LAST.*store equals to 1, it will process the conditional statements to check and see if the sale is less than, greater than or equal to the first_value.

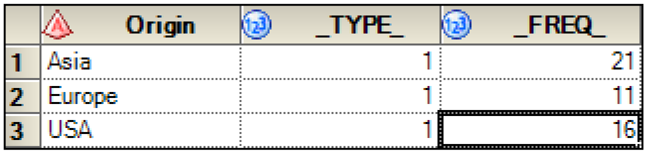
HOW DO I SPLIT ONE DATASET INTO MANY DATASETS?

Using the cars dataset above, let's say that we want to split that one dataset into multiple datasets by Origin.

Code:

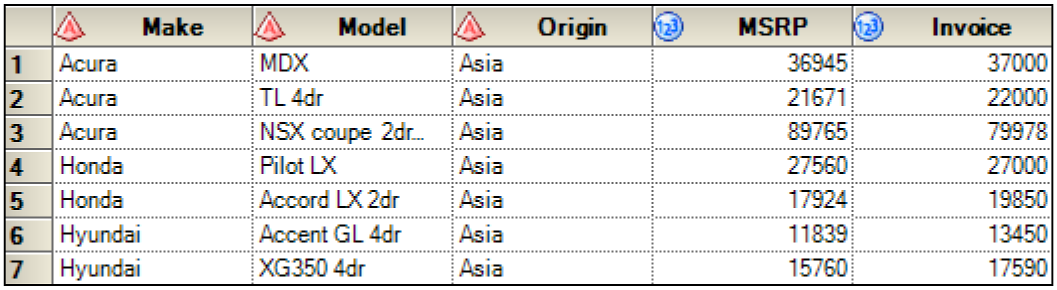
```
proc summary data=cars missing nway;
      class Origin;
      output out=cars_origin;
run;

data Asia Europe USA;
      set cars;
      if Origin = 'Asia' then output Asia;
            else if Origin = 'Europe' then output Europe;
            else if Origin = 'USA' then output USA;
run;
```

Output Dataset:



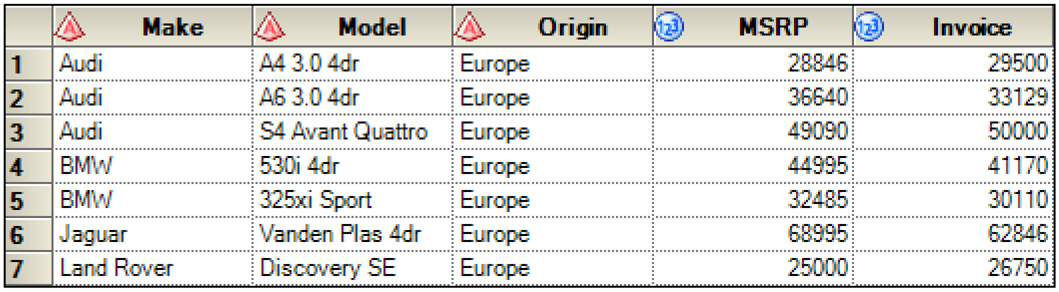
	Origin	_TYPE_	_FREQ_
1	Asia	1	21
2	Europe	1	11
3	USA	1	16

Output Dataset (Asia):



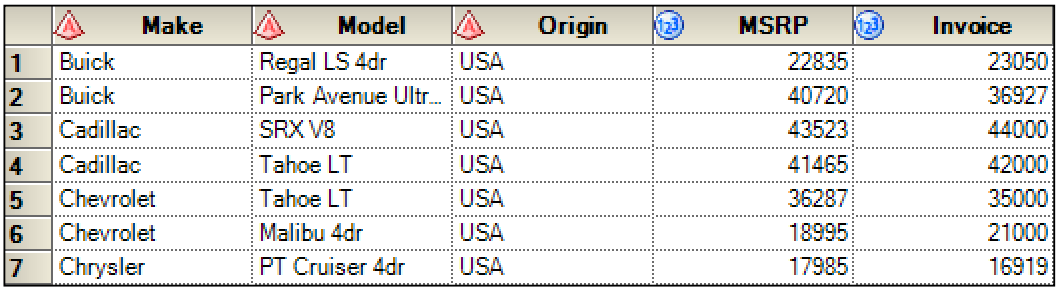
	Make	Model	Origin	MSRP	Invoice
1	Acura	MDX	Asia	36945	37000
2	Acura	TL 4dr	Asia	21671	22000
3	Acura	NSX coupe 2dr...	Asia	89765	79978
4	Honda	Pilot LX	Asia	27560	27000
5	Honda	Accord LX 2dr	Asia	17924	19850
6	Hyundai	Accent GL 4dr	Asia	11839	13450
7	Hyundai	XG350 4dr	Asia	15760	17590

Output Dataset (Europe):



	Make	Model	Origin	MSRP	Invoice
1	Audi	A4 3.0 4dr	Europe	28846	29500
2	Audi	A6 3.0 4dr	Europe	36640	33129
3	Audi	S4 Avant Quattro	Europe	49090	50000
4	BMW	530i 4dr	Europe	44995	41170
5	BMW	325xi Sport	Europe	32485	30110
6	Jaguar	Vanden Plas 4dr	Europe	68995	62846
7	Land Rover	Discovery SE	Europe	25000	26750

Output Dataset (USA):



	Make	Model	Origin	MSRP	Invoice
1	Buick	Regal LS 4dr	USA	22835	23050
2	Buick	Park Avenue Ultr...	USA	40720	36927
3	Cadillac	SRX V8	USA	43523	44000
4	Cadillac	Tahoe LT	USA	41465	42000
5	Chevrolet	Tahoe LT	USA	36287	35000
6	Chevrolet	Malibu 4dr	USA	18995	21000
7	Chrysler	PT Cruiser 4dr	USA	17985	16919

One way we can do that is to summarize the data to see what Origins we have. In this example, we used the PROC SUMMARY to find the Origins in the dataset cars. Once we have all the Origins - Asia, Europe, and USA, we are able to split the one big dataset into three different datasets. We do this by naming the new datasets and using the conditional statements, split the original dataset into multiple datasets.

Code:

```
proc sql noprint;
      select distinct trim(Origin)
      into :Country separated by ' '
      from cars;
quit;

%macro split;
%do i=1 %to %sysfunc(countw(&Country));
    %let Origin=%sysfunc(scan(&Country,&i,' '));
    data &Origin;
        set cars(where=(Origin="&Origin"));
    run;
%end;
%mend;
%split
```

Sometimes we do not want to explore the dataset prior to splitting them, or that the list is too long and we do not want to type out all the dataset names. It could also be as simple as trying to automate code or making code look short and simple. The code above does just that. First, we use PROC SQL to summarize all the Origins that are in the cars dataset and put it into a list called Country. The variables in the Country list are separated by a space.

Next, we create a macro to split the dataset. In this macro code, the first thing we need to do is to count how many Origins are there in the list Country. As shown in the code above, we use the *COUNTW* function to calculate the total number of Origin in the list Country. We then use a *SCAN* function to extract the different Origins from the list Country. *SCAN* function is used since our list contains a space delimiter.

Going through the 1st iteration, we have i equals to 1. The *SCAN* function then extracts the first word until it sees a space and assigns Origin to the first extracted word, which in this case, is Asia. Next, we create a dataset named Asia, and set the original dataset cars with a *WHERE* condition. The *WHERE* condition states that we only want Origin equals to Asia in this dataset. We end the *DO LOOP* after the dataset Asia has been created.

Next, i will increase to 2 and the *SCAN* function will extract the second word (Europe, in this case) from the list Country until it sees a space again and stops. SAS creates a new SAS dataset name Europe, set the original cars dataset with a *WHERE* condition again (where Origin = “Europe”) and ends the loop after the dataset has been created.

Also note that we have to use the *%SYSFUNC* macro function here. This is because we are working on a macro variable &Country (first to get the number of Origins in the list and next, to extract the word using the *SCAN* function).

CONCLUSION

In this paper we looked at some common questions new, and sometimes experienced programmers may have. By no means is this review comprehensive – there are many more questions than can be covered in this short time, however we feel it does cover some of the more common questions users may have. In addition, the solutions are based upon our experience; other programmers will have other solutions. With SAS there is rarely only one right answer.

CONTACT INFORMATION

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

Peter Eberhardt
Fernwood Consulting Group Inc.
Toronto, ON, Canada
peter@fernwood.ca
twitter: @rkinRobin

Audrey Yeo
Athene USA,
West Des Moines, IA
AYeo@athene.com