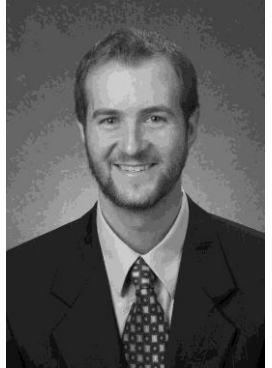


Tips, Tricks, and Techniques from the Experts



SYSTEMS SEMINAR CONSULTANTS, INC.

Presented by Katie Ronk

2997 Yarmouth Greenway Drive, Madison, WI 53711

Phone: (608) 278-9964 • Web: www.sys-seminar.com

Tips, Tricks, and Techniques from the Experts



This paper was written by Systems Seminar Consultants, Inc. SSC specializes in SAS software and offers:

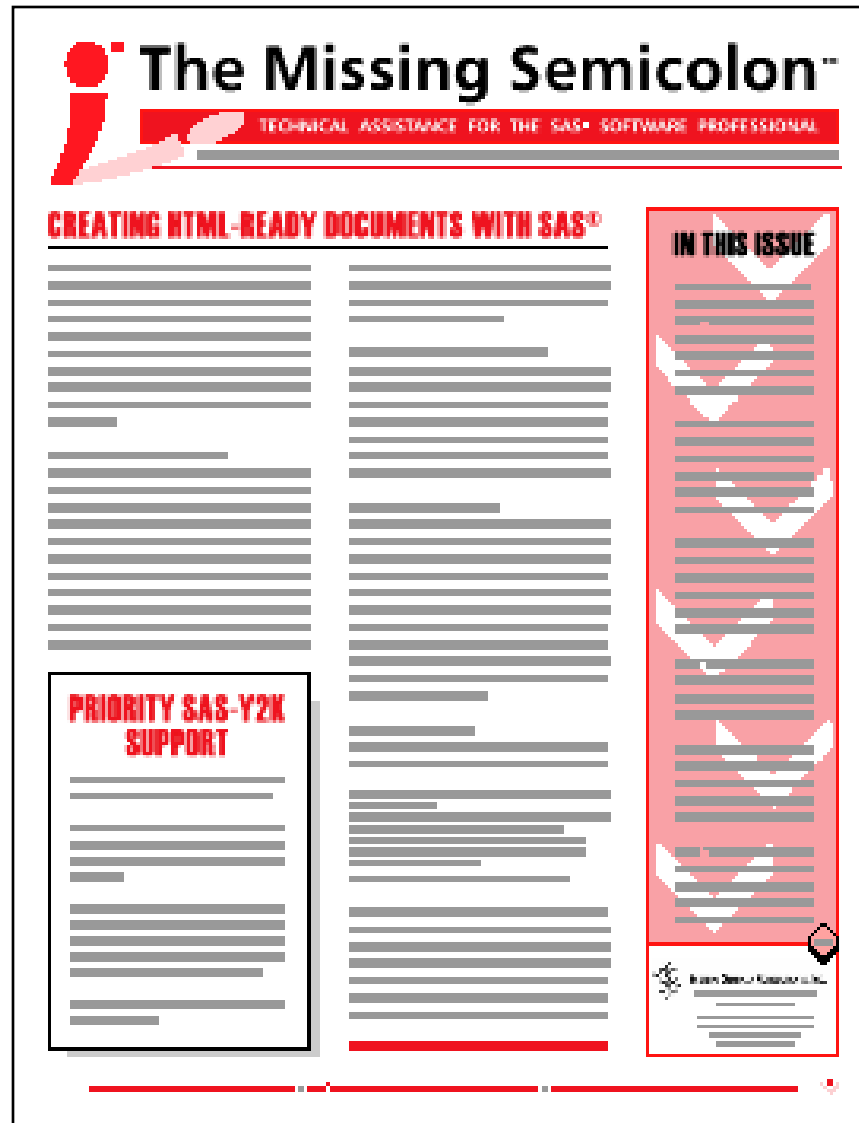
- Training Services
- Consulting Services
- Help Desk Plans
- Newsletter Subscriptions to *The Missing Semicolon*[™]

COPYRIGHT© 2004 Systems Seminar Consultants, Inc.

All rights reserved. Printed in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without prior written permission of SSC. SAS is a registered trademark of SAS Institute Inc. in the USA and other countries. *The Missing Semicolon* is a trademark of Systems Seminar Consultants, Inc.



- **TIPS**
- TRICKS
- TECHNIQUES



Back issues on website: www.sys-seminar.com

Systems Seminar Consultants, Inc • www.sys-seminar.com • 608.278.9964

Tips, Tricks, and Techniques from the Experts



- DATA Step
- PROC Step
- Graphing
- Report Formatting
- Miscellaneous

Data Step Tip: Input @ 'TEXT' (June 1998)



Using INPUT @ ' text' positions the input pointer directly after the word 'text'.

```
DATA NAMES;  
  LENGTH NAME CITY $10;  
  INPUT  @1 NAME  
         @ 'CITY=' CITY;
```

```
DATALINES;  
KATIE CITY=MADISON  
TERESA CITY=MCFARLAND  
STEVE CITY=MONONA  
ANN CITY=WAUNAKEE  
;  
RUN;
```

```
PROC PRINT DATA=NAMES;  
RUN;
```

Data Step Tip: Input @ 'TEXT' (Jun '98)



Output:

Obs	NAME	CITY
1	KATIE	MADISON
2	TERESA	MCFARLAND
3	STEVE	MONONA
4	ANN	WAUNAKEE

- If the text is not found, the pointer will go to a new line.

Data Step Tip: Trailing @ For Efficiency (Oct '98)



Use the trailing "@" with input statements to avoid inputting records you wish to eliminate.

Instead of:

```
DATA MIDWEST;
  INFILE ACCOUNTS;
  INPUT  @1  NAME $19.
         @20 STATE $2.
         @24 ACCTNBR $10.
         @35 BALANCE 8.;
  IF STATE IN ('WI', 'MN', 'IA');
RUN;
```

Data Step Tip: Trailing @ For Efficiency (Oct '98)



Better:

```
DATA MIDWEST;  
  INFILE ACCOUNTS;  
  INPUT @20 STATE $2. @;  
  IF STATE IN ('WI', 'MN', 'IA') THEN  
    INPUT @1 NAME $19.  
    @24 ACCTNBR $10.  
    @35 BALANCE 8. ;  
RUN;
```

- More Efficient
- Different Layouts in One File

Data Step Tip: ATTRIB statement (Feb '99)



Use the ATTRIB statement in the Data Step to associate a format, informat, label, and length with a variable all at once.

Example:

```
ATTRIB NAME LENGTH=$20 FORMAT=$20. LABEL= "EMPLOYEE NAME";
```

Data Step Tip: Alignment Option (Feb '99)



An alignment specification can be coded in the PUT statement. Code -L, -C, or -R after the format.

Example:

```
124 DATA _NULL_;  
125     SET NAMES;  
126     PUT @1 NAME $30. -C;  
127 RUN;
```

```
      KATIE  
      TERESA  
      STEVE  
      ANN
```

NOTE: There were 4 observations read from the data set WORK.NAMES.

Data Step Tip: FIRSTOBS Option (Oct '99)



When reading raw files use the FIRSTOBS=record-number to begin reading the input data at the record number specified.

This is also helpful when you don't want to read a header record, usually stored in the first record of the input file.

Example:

```
INFILE RAWFILE FIRSTOBS=2;
```

Data Step Tip: FIRSTOBS & OBS Options (Oct '99)



To read a range of records from a raw file you can use the FIRSTOBS option combined with OBS=record-number, where record-number specifies the last record that you want to read from an input file.

Example:

```
INFILE RAWFILE FIRSTOBS=10 OBS=30;
```

- Results in 21 records being read, 10 through 30.

Data Step Tip: Adding Variables with Formats

(Jan '00)



A format can be used instead of a join to add a variable.

Example:

Enrollment Dataset:

Obs	NAME	ENRDATE	PLAN
1	KATIE	17SEP2002	A
2	TERESA	17OCT2000	A
3	STEVE	01MAY1998	B
4	ANN	13AUG2001	B

City Dataset:

Obs	NAME	CITY
1	KATIE	MADISON
2	TERESA	MCFARLAND
3	STEVE	MONONA
4	ANN	WAUNAKEE

Data Step Tip: Adding Variables with Formats



Create a User Defined Format from the data:

```
DATA CITYFMT;  
  SET CITY;  
  RETAIN FMTNAME '$CITYFMT';  
  RENAME NAME=START  
         CITY=LABEL;  
RUN;  
  
PROC FORMAT CNTLIN=CITYFMT; RUN;  
PROC PRINT DATA=CITYFMT; RUN;
```

Obs	START	LABEL	FMTNAME
1	KATIE	MADISON	\$CITYFMT
2	TERESA	MCFARLAND	\$CITYFMT
3	STEVE	MONONA	\$CITYFMT
4	ANN	WAUNAKEE	\$CITYFMT

Data Step Tip: Adding Variables with Formats



Create a User Defined Format from the data:

Obs	START	LABEL	FMTNAME
1	KATIE	MADISON	\$CITYFMT
2	TERESA	MCFARLAND	\$CITYFMT
3	STEVE	MONONA	\$CITYFMT
4	ANN	WAUNAKEE	\$CITYFMT

Equivalent to:

```
PROC FORMAT;  
  VALUE $CITYFMT  
    'KATIE' = 'MADISON'  
    'TERESA' = 'MCFARLAND'  
    'STEVE' = 'MONONA'  
    'ANN'   = 'WAUNAKEE';  
RUN;
```

Data Step Tip: Adding Variables with Formats



Apply the User Defined Format:

```
DATA EMP;  
  SET ENROLLMENT;  
  CITY=PUT(NAME,$CITYFMT.);  
RUN;
```

```
PROC PRINT DATA=EMP;  
RUN;
```

Obs	NAME	ENRDATE	PLAN	CITY
1	KATIE	17SEP2002	A	MADISON
2	TERESA	17OCT2000	A	MCFARLAND
3	STEVE	01MAY1998	B	MONONA
4	ANN	13AUG2001	B	WAUNAKEE

Data Step Tip: GETOPTION Function (Apr '00)



Use the GETOPTION function to create variables to hold SAS option values.

Example:

```
353 DATA _NULL_;  
354     YEARCUT=GETOPTION( ' YEARCUTOFF ' );  
355     PUT YEARCUT=;  
356 RUN;
```

```
YEARCUT=1920
```

Data Step Tips: WHERE ALSO operator (Apr '00)



If you have a long WHERE clause which contains an AND, you can break it into two clauses.

The ALSO operator adds requirements to your first WHERE clause.

Example: WHERE SALES>=500;
 WHERE ALSO EXPENSE<=100;

Data Step Tip: COMPBL Function (Jul '00)



To take extra blanks out of a variable (character), use the COMPBL function.

Example: BIGNAME="MARY JANE SMITH"
 SMALLER=COMPBL(BIGNAME);

Resulting in: "MARY JANE SMITH"

Data Step Tip: Stopping Data Errors (Jul '00)



Check denominator to avoid division by zero errors.

Possible Data Errors:

```
360     AVGBAL=TOTBAL/NUMB;  
361     RUN;
```

NOTE: Missing values were generated as a result of performing an operation on missing values.

Each place is given by: (Number of times) at (Line):(Column)

1 at 360:16

Better:

```
IF NUMB NOT IN (0,.) THEN AVGBAL=TOTBAL/NUMB;  
ELSE AVGBAL=0;
```

Data Step Tip: Comment Out Code (Oct '00)



A simple way to block a section of SAS code from being processed is to make it look like it is a piece of macro code but never invoke it:

```
%MACRO COMMENT;      /* Starts a MACRO definition */  
DATA WHATEVER;  
    .....  
RUN;  
PROC PRINT DATA=WHATEVER;  
    .....  
%MEND COMMENT;      /* End of block to ignore */
```

Data Step Tip: Safe Space (Oct '00)



Numeric variables default to 8 bytes in SAS. If you have a numeric date, these can be safely stored in a 4 byte numeric variable (5 bytes for WINDOWS/UNIX), such as:

```
362 DATA _NULL_;
363     LENGTH TODAY TODAY2 4;
364     TODAY=TODAY();
365     TODAY2=TODAY();
366     FORMAT TODAY2 DATE9.;
367     PUT _ALL_;
368 RUN;
```

```
TODAY=16391 TODAY2=16NOV2004 _ERROR_=0 _N_=1
```

Data Step Tip: Safe Space (Oct '00)



Numeric variables default to 8 bytes in SAS. If you have a numeric date, these can be safely stored in a 3 byte numeric variable (4 bytes for WINDOWS/UNIX), such as:

	Windows	MVS
2	NA	256
3	8,192	65,536
4	2,097,152	16,777,216
5	536,870,912	4,294,967,296
6	137,438,953,472	1,099,511,627,776
7	35,184,372,088,832	281,474,946,710,656
8	9,007,199,254,740,990	72,057,594,037,927,900

Data Step Tip: Random Sample (Oct '00)



To read a random sample of roughly 15% of the data from any file, use the RANUNI(0) function:

```
DATA TESTING;
  INFILE RAWIN;
  INPUT    @;          /* read a record and hold it */
  IF RANUNI(0) LT .15; /* allows about 15% of
                        rows to be used */

  INPUT    rest of program.....
```

Data Step Tip: MORT Function (Jan '02)



Use the MORT function to calculate payments on loans.

```
MORT (AMOUNT, PAYMENT, RATE, NUMBER)
```

Should you refinance your home loan? Comparing two different interest rates:

```
377     data _null_;
378         oldpay=mort(100000, ., .08/12, 30*12);
379         newpay=mort(100000, ., .06/12, 30*12);
380         savings=oldpay-newpay;
381         put _all_;
382         format oldpay newpay savings dollar10.2;
383     run;
```

```
oldpay=$733.76 newpay=$599.55 savings=$134.21 _ERROR_=0 _N_=1
```

DATA Step: FILEEXIST Function (Jan '01)



The "FILEEXIST()" function can tell you if a file exists on the system.

It returns 0 if the file does NOT exist and 1 if it does.

Examples:

<u>SAS Code</u>	<u>file exist?</u>	<u>X value</u>
X=FILEEXIST('ABC.XYZ.TEST');	Yes	1
X=FILEEXIST('C:\MYDATA\XYZ.DAT');	No	0
X=FILEEXIST('ABC.XYZ.PDS(GOOD)');	Yes	1

PROC Step Tip: PROC PRINTTO (Jun '99)



Use the PROC PRINTTO procedure to output a report to a file.

Example:

```
FILENAME DUMP 'C:\FILE.TXT';
PROC PRINTTO PRINT=DUMP NEW;
PROC PRINT;
          FORMAT VAR1 MMDDYY8.;
RUN;
PROC PRINTTO; /*TURN OFF */
```

The LOG option on the PROC PRINTTO statement can direct the SASLOG to a file.



To rename the label on the OBS column in PROC PRINT code the OBS= option.

Example:

```
PROC PRINT DATA=TEST OBS='Survey Number';  
  Title 'Results of December Survey';  
RUN;
```

Results of December Survey

Survey Number	Question	Result
1	12	Yes
2	12	No
3	12	Unsure

Graphing: Descending Option (Jun '99)



Use the descending option within the VBAR or HBAR statement to produce bar charts in descending order of magnitude.

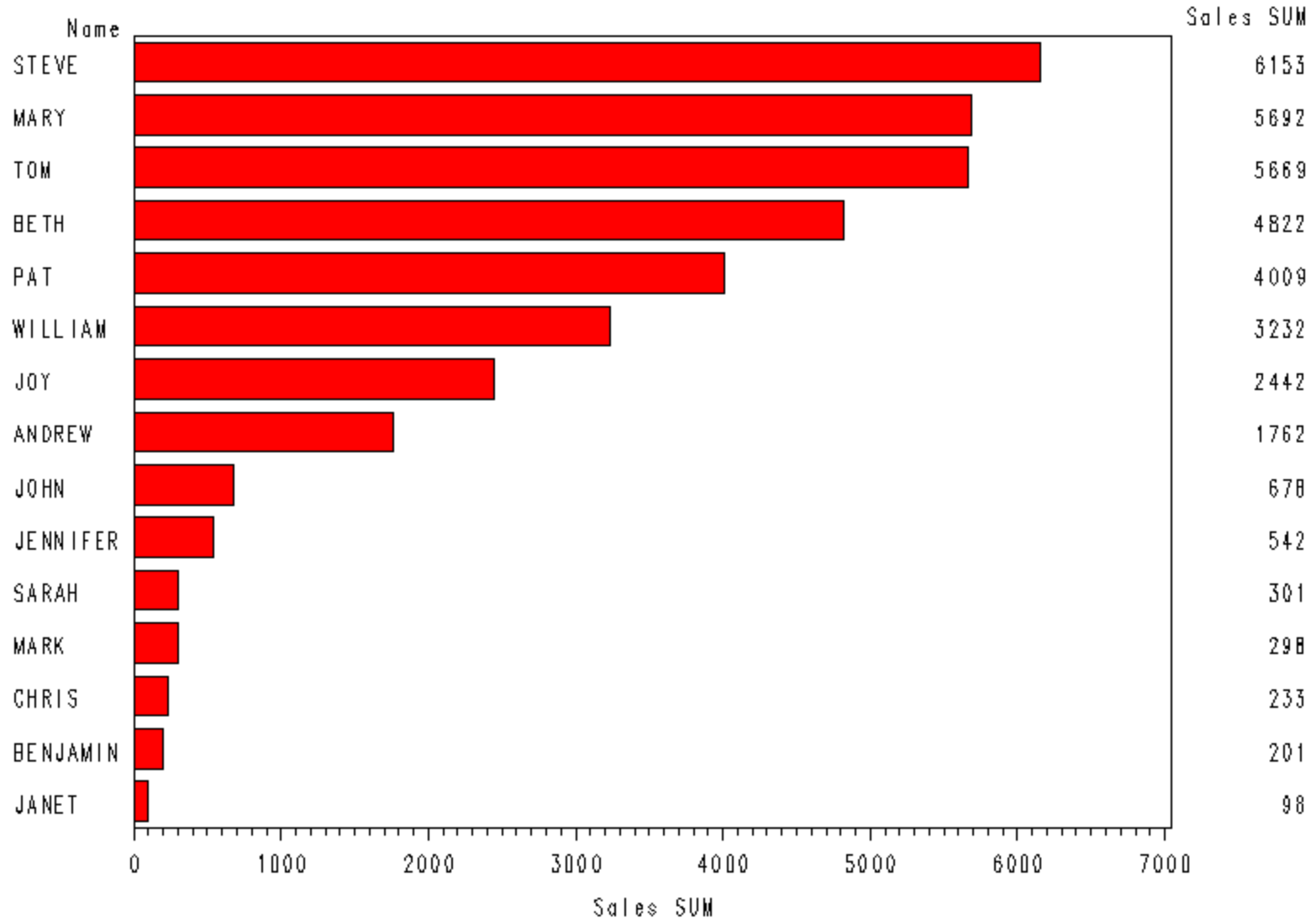
Example:

```
PROC GCHART DATA=SASUSER.SOFTSALE;  
  HBAR NAME/DESCENDING SUMVAR=SALES;  
RUN;  
QUIT;
```

Graphing: Descending Option (Jun '99)



Use the descending option within the VBAR or HBAR



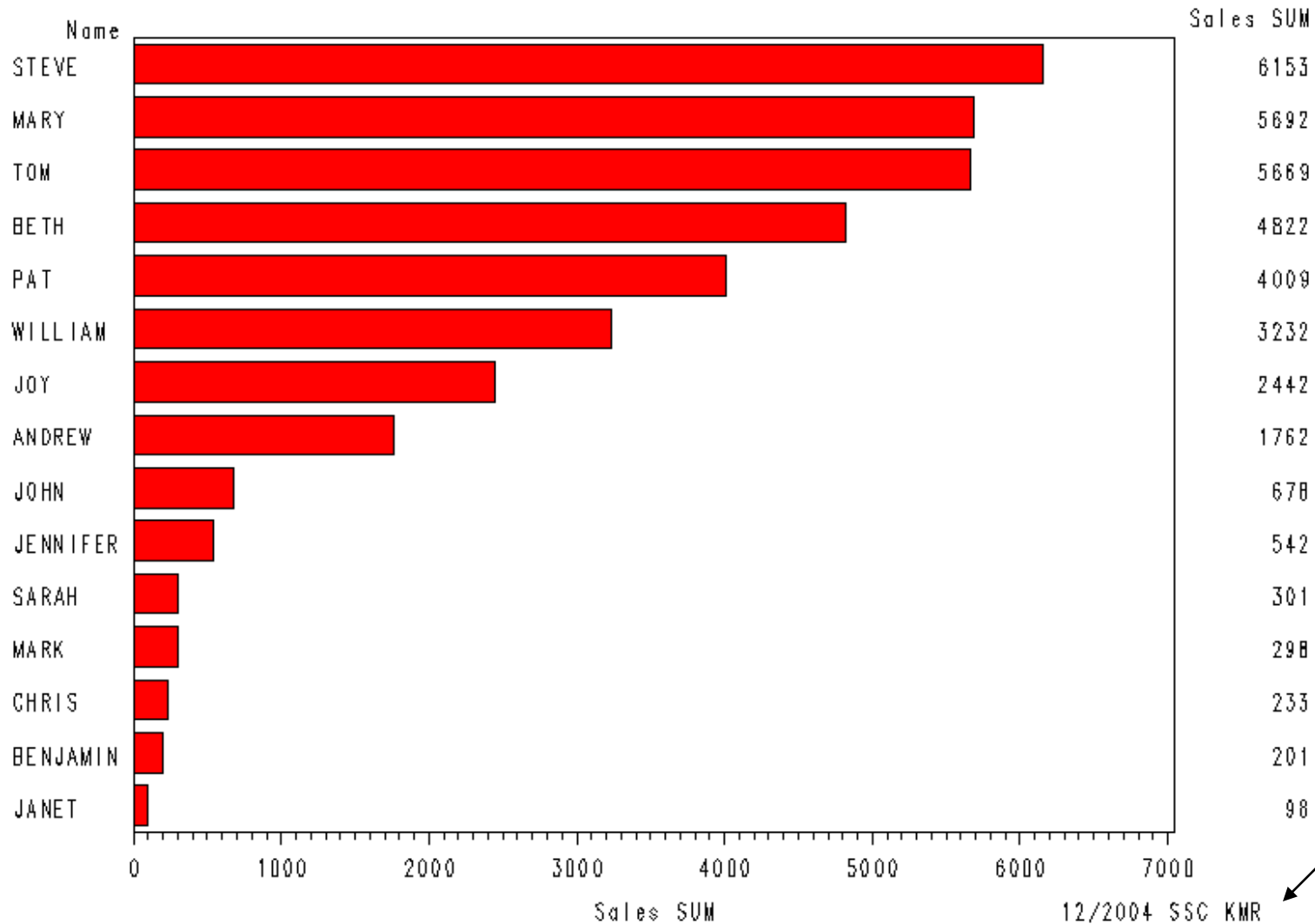
Graphing: Annotate Datasets (Jun '99)



Utilize an annotate dataset to custom initialize your graphs:

```
data MyAnnotate;  
    FUNCTION='LABEL';  
    TEXT='12/2004 SSC KMR'; /*DATE, COMPANY, INITIALS */  
    SIZE=1; X=85; Y=1;  
    OUTPUT;  
  
RUN;  
  
TITLE;  
PROC GCHART DATA=SASUSER.SOFTSALE ANNOTATE=MYANNOTATE;  
    HBAR NAME/DESCENDING SUMVAR=SALES;  
RUN;  
QUIT;
```

Graphing: Annotate Datasets (Jun '99)



PROC Summary: ID statement (Jan '00)



To take along an extra value in the output dataset of a PROC SUMMARY, use the ID statement.

- Default is the maximum value
- Option for minimum value.

Example:

```
PROC SUMMARY DATA=SALES NWAY IDMIN;  
  VAR SALEAMT;  
  ID SALEDATE;  
  CLASS CUSTNUMB;  
  OUTPUT OUT=SALESUM  
  N(SALEAMT)=NUMSALES  
  SUM(SALEAMT)=TOTSALES;  
RUN;
```

Report Formatting: Remove Page Breaks (Jun '98)



To stop SAS from issuing page breaks between pages of a reports, set the FORMDLIM option equal to a quoted blank.

```
OPTIONS FORMDLIM=' ';
```

```
PROC PRINT DATA=SOFTSALE;  
RUN;
```

```
PROC FREQ DATA=SOFTSALE;  
RUN;
```

Report Formatting: Remove Page Breaks (Jun '98)



FORMDLIM=' ' removes all page breaks.

Obs	Name	Division	Years	Sales	Expense	Stat
1	CHRIS	H	2	233.11	94.12	WI
2	MARK	H	5	298.12	52.65	WI
3	SARAH	S	6	301.21	65.17	MN
4	PAT	H	4	4009.21	322.12	IL
5	JOHN	H	7	678.43	150.11	WI
6	WILLIAM	H	11	3231.75	644.55	MN
7	ANDREW	S	24	1762.11	476.13	MN

...

The FREQ Procedure

Name	Frequency	Percent	Cumulative Frequency	Cumulative Percent
ANDREW	1	6.67	1	6.67
BENJAMIN	1	6.67	2	13.33
BETH	1	6.67	3	20.00

Report Formatting: PAGENO= (Feb '99)



If your job contains many PROCs, and you want the first page of each PROC to be number 1, code the following line before each PROC:

```
OPTIONS PAGENO=1;
```

Report Formatting: SKIP= (Jan '01)



By default, all printed reports do not skip any lines at the top of the page.

Change this with the SKIP=nn global option.

```
OPTIONS SKIP=5;
```

```
TITLE "This Title will start on line 5 of the page, not line 1";  
PROC PRINT DATA=TEST;  
RUN;
```

Report Formatting: Macro Variables (Oct '99)



To include text on a report and avoid modifying each PROC, create a macro variable at the top of the program and refer to it in the PROC.

Simply modify the macro the next time you run the same job.

Example:

```
%LET MONTH=August 1999;  
TITLE1 "Sales Summary as of &MONTH ";
```



To put the number of observations in a title, use the SET statement and the NOOBS option to query the compiler. SYMPUT can then create a macro variable usable in a title.

Example:

```
DATA _NULL_;  
    CALL SYMPUT('TOTOBS', TRIM(LEFT(PUT(NMEMB, 8.))));  
    SET INDATA NOBS=NMEMB;  
    STOP;
```

```
RUN;
```

```
TITLE "&TOTOBS OBSERVATIONS IN THE DATASET";  
PROC MEANS DATA=INDATA;  
RUN;
```

Report Formatting: WIDTH=MIN (Jul '00)



To condense the size of your proc print output, use the width=min option.

```
TITLE;  
PROC PRINT DATA=SOFTSALE WIDTH=MIN;  
RUN;
```



The MISSING= option can be used to print another character in the place of missing values.

Example:

```
OPTIONS MISSING='*';
```

General Processing: CANCEL option (Oct '00)



You can use the RUN CANCEL; statement to have a step compiled but not executed.

This is great for checking SYNTAX!

```
PROC PRINT DATA=SOFTSALE;  
  VAR NAME DIVISION;  
RUN CANCEL; /* step compiles but does not execute */
```



Many dataset options can be used in PROC steps as well as DATA steps. The most common data step options are RENAME, WHERE, DROP and KEEP.

Example:

```
proc summary data=customer(rename=(id=custid));  
  class custid;  
  var sales;  
  output out=nosales(where =(totalsales <= 0 )  
                    rename=( _freq_=numbobs)  
                    drop  =_type_)  
  sum(sales)=totalsales;  
run;
```

Data Set Processing: Label Option (Jun '98)



Data sets have labels too! The data set label is a good place to store critical information about the dataset.

```
DATA EMPLOYEES(LABEL="Employees of Systems Seminar Consultants");  
...  
RUN;
```

The CONTENTS Procedure

Data Set Name:	WORK.EMPLOYEES	Observations:	0
Member Type:	DATA	Variables:	6
Engine:	V8	Indexes:	0
Created:	15:10 Tuesday, November	Observation Length:	40
Last Modified:	15:10 Tuesday, November	Deleted Observations:	0
Protection:		Compressed:	NO
Data Set Type:		Sorted:	NO
Label:	Employees of Systems Seminar Consultants		

Variable Processing: Colon Wildcard (Jun '98)



The colon can be used as a wildcard in a variable list. For example, BAL: refers to all variables beginning with BAL.

```
PROC PRINT DATA=EMPLOYEES;  
  VAR NAME BAL: ;  
RUN;
```

Obs	NAME	BAL1	BALAMT	BAL20
1	Katie	400	331	45

GROUPING: NOTSORTED option (Feb '99)



When data is grouped by logical sections, but remains unsorted alphabetically or numerically (i.e. JAN, FEB, MAR, etc.), a BY statement can still be used with the NOTSORTED option.

Example:

```
PROC PRINT DATA=MYDATA;  
  BY MONTH NOTSORTED;  
  VAR MONTH AMOUNT;  
RUN;
```

GROUPING: NOTSORTED option (Feb '99)



Sample Output:

----- MONTH=JANUARY -----

Obs	MONTH	AMOUNT
1	JANUARY	300

----- MONTH=FEBRUARY -----

Obs	MONTH	AMOUNT
2	FEBRUARY	300



To change the colors in your SAS environment while working with display manager, submit the following statement:

```
DM 'COLOR area color';
```

Example: `DM 'COLOR background yellow';`

The areas you can color are: background, banner, border, command, foreground, and message

PC SAS: Open Quote Problem (Jul '00)



Submitting code with missing quote marks (single ' or double ") in Display Manager or Windows/SAS can be frustrating to fix.

Try submitting the following to close off the mis-quoted string:

```
* ' ; * " ; run ;
```

Date and Time Processing: Date Format (Apr '00)



To format a SAS date with a four-digit year, two-digit month, and two-digit day without slashes (e.g., 20041120), use the YYMMDDN8. format.



Use the TIMEAMPMw. format to display times with AM or PM.

Example:

<u>Value</u>	<u>Format</u>	<u>Result</u>
'18:15'T	TIME5.	18:15
	TIMEAMPM8.	6:15 PM

Date & Time Processing:DATEPART Function(Jul '00)



To change a datetime variable to a date only, use the datepart function.

—
— Example: `mydate=DATEPART(mydate) ;`

Space Issues: COMPRESS Option (Oct '99)



If you are running out of space (disk space) when creating new data sets try the data set `OPTION COMPRESS=YES;`

This reduces the data file storage size by using a compression technique.

Space Issues: Drop Unneeded Datasets (Jan '00)



To clear up space in the middle of your programs, delete any unneeded data sets as soon as they are no longer needed.

Example:

```
PROC DATASETS;  
  DELETE TEST;  
QUIT;
```

or

```
PROC SQL;  
  DROP TABLE TEST;  
QUIT;
```



SYSTEMS SEMINAR CONSULTANTS, INC.

SAS® Training, Consulting, & Help Desk Services
2997 Yarmouth Greenway Drive • Madison, WI 53711
(608) 278-9964 • Fax (608) 278-0065
www.sys-seminar.com



Katie Ronk

Director of Operations
kronk@sys-seminar.com

