

SCL Mini Manual 2  
SAS in the SCL

September 27, 1999

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Files used in SAS</b>	<b>5</b>
<b>3</b>	<b>Starting a SAS Session</b>	<b>7</b>
<b>4</b>	<b>Data Entry</b>	<b>9</b>
4.1	The Data Step . . . . .	9
4.1.1	Reading Data from an External File . . . . .	10
4.1.2	Reading Instream Data . . . . .	10
4.1.3	Reading Data from Existing SAS Data Sets . . . . .	11
4.1.4	Generating Data from Programming Statements . . . . .	11
4.2	A Sample DATA Step . . . . .	11
<b>5</b>	<b>Basic Statistical Analysis</b>	<b>13</b>
5.1	UNIVARIATE Procedures . . . . .	13
5.2	MEANS procedure . . . . .	16
5.3	SUMMARY procedure . . . . .	17
5.4	RANK procedure . . . . .	17
5.5	SORT . . . . .	19



# Chapter 1

## Introduction

SAS, Statistical Analysis System, is an extremely powerful statistical package. Available on SCL, SAS offers a wide range of data management, statistical, and graphical procedures. SAS basics are very easy to learn for the beginning user, regardless of programming experience. This document is intended to get you started using SAS.



# Chapter 2

## Files used in SAS

Using SAS requires some knowledge of the UNIX operating systems and a text editor such as vi. It also requires some knowledge of files. Depending on the complexity of the analysis, some or all the file types below could be used.

- **SAS Program File** - The file which contains your SAS statements. This file must have a filetype extension of sas.
- **Data File** - Contains the data you wish SAS to process. This file are usually named with a filetype of dat.
- **Listing File** - The file created by SAS which contains the results from the SAS procedures you have requested. This file will always have the file types of lst.
- **Log File** - A file created by SAS which contains diagnostic information about your SAS job, including any errors which SAS may encounter. This file will always have the file type of log.



# Chapter 3

## Starting a SAS Session

To start SAS session at the `~>` prompt simply type

```
~> sas
```

The system will then open 3 windows in screen. They are LOG, PROGRAM EDITOR and OUTPUT windows. This display mode is called *Display Manager Mode*. In *Display Manager Mode*, you execute SAS programs in a windowing environment. You can edit and execute programming statements, display the SAS log, display procedure output, display online help, set function keys, and more. We can also use *Noninteractive Mode*. In *Noninteractive Mode*, SAS program statement are stored in an external file. That file type is *.sas*. The statements in the file execute immediately when you issue a SAS command referencing the file. For example, suppose you have written a sas program file, *myfile.sas*, then simply type

```
~> sas myfile.sas
```

After a few seconds, the UNIX prompt will displayed again. It means that your job is finished. Then you type `ls myfile.*`, you should see two new files listed: `myfile.lst` and `myfile.log`. *myfile.lst* is the output file created by SAS which contains the results of your analysis. If you SAS program contains errors, this file may not be created.

Any errors which the program might have contained, as well as other information about your SAS job, are contained in the log file - *myfile.log*.



# Chapter 4

## Data Entry

Before you can analyze your data and produce a report with SAS software, the data must be in a form the SAS System can recognize. This form is called a *SAS data set*, and it consists of the following:

- descriptor information
- data values.

The *descriptor information* describes the contents of the SAS data set to the SAS System. The *data values* are the data that have been collected or calculated. They are organized into a rectangular structure containing rows called *observations* and columns called *variables*. An *observation* is a collection of data values that usually relate to a single object. A *variable* is the set of data values that describe a given characteristic.

To create a SAS data set from raw data with base SAS software, you had to use a group of SAS language statements called a *DATA step*. Using a DATA step, you can create a SAS data set in which data values are physically stored.

### 4.1 The Data Step

A DATA step consists of a group of statements in the SAS language that read raw data or existing SAS data sets to create a SAS data set. The kind of SAS data set created in a DATA step is called a *SAS data file*. With a DATA step, you can read your raw data or other existing SAS data sets and perform the calculations or manipulation necessary so that you can analyze your data and create reports with *SAS procedures*.

The type of input data used in a DATA step determines what data-reading statements you use. There are four types of simple DATA steps, categorized by the source of the input data: those that read

- data from an external file

- instream data
  
- data from existing SAS data sets
  
- no data but generate data from programming statements.

### 4.1.1 Reading Data from an External File

Structure of a DATA step:

DATA dataname;	marks the beginning of the DATA step and gives a name ( <i>dataname</i> ) to the SAS data set being created.
INFILE filename;	identifies the external file ( <i>filename</i> ) that contains the data.
INPUT statement;	describes your input by giving a name to each variable and identifying its location in the data record. It causes a data record to be read.
Other SAS statement;	enable you to modify the data, create new variables and so on.

### 4.1.2 Reading Instream Data

Structure of a DATA step:

DATA dataname;	marks the beginning of the DATA step and gives a name ( <i>dataname</i> ) to the SAS data set being created.
INPUT statement;	describes your input by giving a name to each variable and identifying its location in the data record.
Other SAS statement;	enable you to modify the data, create new variables and so on.
CARDS;	marks the end of the programming statements and the beginning of the data.
Data lines	are records of data values.
;	marks the end of the data lines.

### 4.1.3 Reading Data from Existing SAS Data Sets

Structure of a DATA step:

DATA dataname;	marks the beginning of the DATA step and gives a name ( <i>dataname</i> ) to the SAS data set being created.
SET, MERGE or UPDATE statement;	identifies the existing SAS data sets used as input in the current DATA step. No description of the data values is needed; the descriptor portion of the existing SAS data set provides the necessary information about variables.
BY statement;	specifies the identifying variables for the SET, MERGE, or UPDATE statement. This statement is optional with a SET or a MERGE statement but is required with an UPDATE statement.
Other SAS statement;	enable you to modify the data, create new variables and so on.

### 4.1.4 Generating Data from Programming Statements

It is possible to create data for a SAS data set by generating observations with programming statements, such as DO loops and assignment statements, rather than by reading data.

Structure of a DATA step:

DATA dataname;	marks the beginning of the DATA step and gives a name ( <i>dataname</i> ) to the SAS data set being created.
Programming statements;	generate data, as with DO loops and assignment statements.
Other SAS statement;	enable you to modify the data, create new variables and so on.

## 4.2 A Sample DATA Step

The following statements provide a simple example of a DATA step.

```
DATA age;
INFILE 'age.dat';
INPUT age gender;
DATA press;
INPUT low high;
diff=high-low;
CARDS;
70 110
75 108
90 110
80 112
```

```
95 130
;
DATA general;
MERGE age press;
diffage=diff*age;
FILE 'total.dat';
PUT age -- diffage;
```

In above DATA step, at first step, we create a SAS data set *age* from raw data stored in an external file *age.dat*. This SAS data set includes two variables *age* and *gender*. The second step is to produce a SAS data set *press* from data lines in the job stream. In this SAS data set, there are three variables *low*, *high*, and new created variable *diff*. At third step, we create a SAS data set *general*. This SAS data set one-to-one merging combines observations from two SAS data set *age* and *press*. In same time, we create a new variables *diffage*. So, there are total 6 variables in this SAS data set. We use FILE 'total.dat' to create external data file *total.dat*. PUT *age* -- *diffage*, this step writes variables *age* to *diffage* to the file *total.dat*.

# Chapter 5

## Basic Statistical Analysis

The SAS programming statements are consisted by several PROC steps to do some sort of analysis. In this section, we will brief introduction some basic SAS procedures.

### 5.1 UNIVARIATE Procedures

This procedure computes univariate statistics, including quantiles, and draws distributional plots. It can provides the following:

- details on the extreme values of a variable
- quantiles, such as the median
- frequency tables
- several plots to illustrate the distribution
- paired comparison tests
- tests of central location
- a test to determine whether the data are normally distributed

The UNIVARIATE procedure is controlled by the following statements:

```
PROC UNIVARIATE <DATA=data set name><FREQ><PLOT><NORMAL>;  
BY variable-list;  
VAR variable-list;
```

Option:

1. **DATA=***SAS-data-set* names the SAS data set to be used by PROC UNIVARIATE. If the **DATA=** option is omitted, the most recently created SAS data set is used.

2. **FREQ** requests a frequency table consisting of the variable values, frequencies, percentages, and cumulative percentages.
3. **PLOT** produces a stem-and-leaf plot (or a horizontal bar chart), a box plot, and a normal probability plot. If a **BY** statement is used, side-by-side boxplots labeled **Schematic Plots** appear for groups defined by the **BY** variables.
4. **NORMAL** computes a test statistic for the hypothesis that the input data come from a normal distribution. The **NORMAL** option also computes and prints the probability of a more extreme value of the test statistic. If the sample size is less than or equal to 2000, this is the Shapiro-Wilk statistic. Otherwise, it is the Kolmogorov statistic.
5. **BY variable-list** obtain separate analyses on observations in groups defined by the **BY** variables.

Example: SAS program:

```
option ls=80;
DATA press;
INPUT low high;
diff=high-low;
CARDS;
70 110
75 108
90 110
80 112
95 130
;
PROC UNIVARIATE FREQ PLOT NORMAL;
VAR diff low high;
run;
```

The selected SAS output (for variable **fidd**):

#### Univariate Procedure

Variable=DIFF

#### Moments

N	5	Sum Wgts	5
Mean	32	Sum	160
Std Dev	7.382412	Variance	54.5
Skewness	-1.2303	Kurtosis	2.484808
USS	5338	CSS	218

CV	23.07004	Std Mean	3.301515
T:Mean=0	9.692521	Pr> T	0.0006
Num ^= 0	5	Num > 0	5
M(Sign)	2.5	Pr>= M	0.0625
Sgn Rank	7.5	Pr>= S	0.0625
W:Normal	0.901126	Pr<W	0.4131

Quantiles(Def=5)

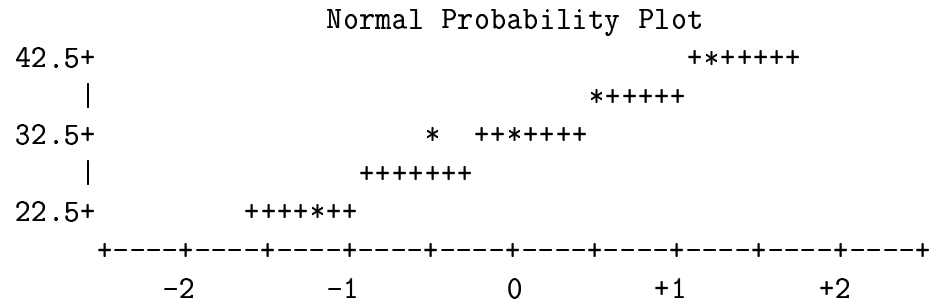
100% Max	40	99%	40
75% Q3	35	95%	40
50% Med	33	90%	40
25% Q1	32	10%	20
0% Min	20	5%	20
		1%	20
Range	20		
Q3-Q1	3		
Mode	20		

Extremes

Lowest	Obs	Highest	Obs
20(	3)	20(	3)
32(	4)	32(	4)
33(	2)	33(	2)
35(	5)	35(	5)
40(	1)	40(	1)

Stem Leaf	#	Boxplot
4 0	1	0
3 5	1	+-----+
3 23	2	*--+--*
2		
2 0	1	*
-----+-----+-----+-----+		
Multiply Stem.Leaf by 10**+1		

Variable=DIFF



Frequency Table

		Percents				Percents	
Value	Count	Cell	Cum	Value	Count	Cell	Cum
20	1	20.0	20.0	35	1	20.0	80.0
32	1	20.0	40.0	40	1	20.0	100.0
33	1	20.0	60.0				

## 5.2 MEANS procedure

The MEANS procedure produces simple univariate descriptive statistics for numeric variables. You can use the OUTPUT statement to request that MEANS output statistics to a SAS data set. It is the easiest and most direct descriptive procedure.

The MEANS procedure is controlled by the following statements:

```

PROC MEANS;
VAR variable-list;
FREQ variable;
WEIGHT variable;
BY variable-list;
OUTPUT SAS data set name;

```

Example: SAS program:

```

option ls=80;
DATA press;
INPUT low high;
diff=high-low;
CARDS;
70 110
75 108

```

```

90 110
80 112
95 130
;
PROC MEANS;
VAR diff low high;
run;

```

The SAS output is:

Variable	N	Mean	Std Dev	Minimum	Maximum
DIFF	5	32.0000000	7.3824115	20.0000000	40.0000000
LOW	5	82.0000000	10.3682207	70.0000000	95.0000000
HIGH	5	114.0000000	9.0553851	108.0000000	130.0000000

### 5.3 SUMMARY procedure

The SUMMARY procedure computes descriptive statistics on numeric variables in a SAS data set and outputs the results to a new SAS data set. PROC SUMMARY does not produce printed output except when you specify the PRINT option. The SUMMARY procedure performs tasks similar to the MEANS procedure.

The SUMMARY procedure is controlled by the following statements:

```

PROC SUMMARY;
VAR variable-list;
FREQ variable;
WEIGHT variable;
BY variable-list;
OUTPUT SAS data set name;

```

### 5.4 RANK procedure

The RANK procedure computes ranks for one or more variables across the observations of a SAS data set. The ranks are output to a new SAS data set. Alternatively, PROC RANK procedures normal scores or other rank scores.

The RANK procedure is controlled by the following statements:

```

PROC RANK option;
BY variable-list;

```

**RANKS** *new-variable-list*;  
**VAR** *variable-list*;

Option:

- **DATA=** names the SAS data set to be used by PROC RANK. If the **DATA=** option is omitted, the most recently created SAS data set is used.
- **DESCENDING** reverses the ranking to be from largest to smallest. The largest value is given a rank of 1, the next smallest a rank of 2, and so on. When the **DESCENDING** option is omitted, values are ranked from smallest to largest.
- **FRACTION** requests fractional ranks. The RANK procedure divides each rank by the number of observations having nonmissing values of the ranking variable and expresses the ranks as fractions.
- **NORMAL=BLOM|TUKEY|VW** requests normal scores to be computed from the ranks. The resulting variables appear normally distributed. The formulas are as follows:

$$\begin{array}{ll} \text{BLOM} & y_i = \Phi^{-1}(r_i - 3/8)/(n + 1/4) \\ \text{TUKEY} & y_i = \Phi^{-1}(r_i - 1/3)/(n + 1/3) \\ \text{VW} & y_i = \Phi^{-1}(r_i)/(n + 1) \end{array}$$

where  $r_i$  is the rank of the  $i$ th observation.

- **SAVAGE** requests Savage (or exponential) scores be computed from the ranks. The scores are computed by this formula:

$$y_i = \left[ \sum_{j=n-r_i+1}^n \frac{1}{j} \right] - 1$$

Example: SAS program:

```
option ls=80;
DATA press;
INPUT low high;
diff=high-low;
CARDS;
70 110
75 108
90 110
80 112
95 130
;
PROC RANK;
```

```

VAR diff low high;
RANKS rdiff rlow rhigh;

PROC PRINT;
run;

```

The SAS output is:

OBS	LOW	HIGH	DIFF	RDIFF	RLOW	RHIGH
1	70	110	40	5	1	2.5
2	75	108	33	3	2	1.0
3	90	110	20	1	4	2.5
4	80	112	32	2	3	4.0
5	95	130	35	4	5	5.0

## 5.5 SORT

The SORT procedure sorts observation in a SAS data set by one or more variables, storing the resulting sorted observations in a new SAS data set or replacing the original data set.

The SORT procedure is controlled by the following statements:

```

PROC SORT
  BY variable-list

```

Example: SAS program:

```

option ls=80;
DATA press;
INPUT low high;
diff=high-low;
CARDS;
70 110
75 108
90 110
80 112
95 130
;
PROC SORT;
BY diff;
run;

```

The SAS output is:

OBS	LOW	HIGH	DIFF
1	90	110	20
2	80	112	32
3	75	108	33
4	95	130	35
5	70	110	40