

## Chapter 4: Sorting, Printing, Summarizing

- PROC statements have required statements and optional statements.

**Example:** PROC ... DATA = ...;

- This DATA statement is optional, specifies which SAS data set to use. Default is to use most recently created data set.
- BY = ... is a common option.
- Tells SAS to do some analysis separately for each value of the specified variable.
- In PROC SORT, the BY statement is required.
- You must sort the data before using the BY option in other procedures.

- `TITLE` (and `FOOTNOTE`) print text at the top (and bottom, respectively) of the output pages.

**Example:** `TITLE 'Some Statistical Output';`  
`TITLE2 "This is the Stat Dept's Output";`  
`FOOTNOTE 'See page 3 of SAS manual';`

- The null statement `TITLE;` cancels all previous titles (same with footnotes)
- `LABEL` statement allows you to add information about variables (In `DATA` step, label is part of data set. In `PROC` step, it is only in effect for that procedure.)

## Subsetting with WHERE statement

- We've seen how to extract subsets of data in the data step using `IF` statements.
- You can have procedures analyze only a subset of the data using `WHERE`:

```
WHERE ...;
```

where the `...` represents some specific logical condition.

- Note: This method of subsetting *does not* create a new SAS data set.
- This works with any SAS procedure, not just `PROC PRINT`.

## Sorting Data

- `PROC SORT` creates a sorted data set that puts observations in order, according to the values of one or more variables.
- The sorted data set is stored with a name specified in the `OUT` option.
- If `OUT` is not specified, the original data set is overwritten with the sorted data set.
- The required statement `BY = ...;` tells SAS which variable(s) values to use in sorting the data.

- With one `BY` variable, the data are sorted by the values of that variable.
- With more than one `BY` variable, observations sorted by the first variable; then the second variable within the first; etc.  
(Can be useful for categorical data)
- By default, this sorts in *ascending* order (small to large, A to Z). To sort in descending order, put keyword `DESCENDING` *before* the appropriate `BY` variable. (Missing values are considered “low”.)
- Can eliminate observations duplicate in the `BY` variables with `NODUPKEY` option.

## Printing data with PROC PRINT

- PROC PRINT prints data sets to output window. We've already seen PROC PRINT many times, but it has some nice options:
- NOOBS — suppresses printing of observation numbers
- LABEL — prints labels (if defined) instead of variable names

(These appear on the same line as PROC PRINT)

Various optional statements can follow the initial line:

- `BY varname` (prints data separately for each value of a specified variable – data need to be sorted by that variable first)
- `ID var1 var2` (puts `var1`, `var2`, etc. at left of page instead of observation numbers)
- `SUM varname` (prints sums of these variables(s) – group sums also, if `BY` option is specified)
- `VAR var1 var2 var3` (tells SAS which variables to print and in what order — by default, all variables are printed)

## FORMAT statement

- You can specify the format in which your data are printed with a `FORMAT` statement in `PROC PRINT`.
- We've seen several formats for dates. Monetary values can be written with dollar signs:

`DOLLAR8 .`

`DOLLAR8 .2`

- Scientific notation: `E8 .`
- Other formats given on pg. 112-113.
- Used in `DATA` step, `FORMAT` sets the format permanently.
- Used in `PROC` step, `FORMAT` only works for that procedure.



- `PUT` statement is similar, but used for writing data to a raw data file.
- With `PROC FORMAT`, you can create your own formats — good for printing coded data in a way that's easy to read.
- Specify the name of your created format after the keyword `VALUE`.

## Simple Custom Reports with FILE and PUT

- FILE is the reverse of INFILE: it prints to an external file.
- PUT (opposite of INPUT) specifies exactly what is printed to that file.

```
FILE 'file specification' PRINT;
```

```
PUT ...
```

```
...;
```

- PUT `_PAGE_;` — skips to the next page after each report.

- Spacing options for `PUT` are same as those for `INPUT`:

`@n` → moves to position `n`

`+n` → skips `n` spaces

`/` → skips to next line

`#n` → skips to `n`-th line

`@` → holds current line

- `PUT` can put variable values and text in specified locations. The format is specified the same way as with `FORMAT` statement.
- If no `FILE` statement given, the report is printed to the Log window.

## Summary Statistics with PROC MEANS

- PROC MEANS can give a variety of summary statistics for each variable.

```
PROC MEANS ...;
```

(with a list of summary statistics you want)

- By default, SAS gives `n`, `mean`, `stddev`, `min`, `max`.
- Can specify others: `MEDIAN`, `NMISS`, `RANGE`, `SUM`, `Q1`, `Q3`, `CLM`, etc. (see list on pg. 118)

Optional statements:

```
BY ...;
```

(calculates summary stats separately for each level of BY variable — data must be sorted first)

```
CLASS ...;
```

(similar to BY, but data don't need to be sorted)

```
VAR ...;
```

(tells SAS specifically which variables you want summary stats for — default is to use all numeric variables)

You can write the summary statistics to another SAS data set using an OUTPUT statement:

```
OUTPUT OUT = ... _____ _____ _____;
```

## Counting Data with PROC FREQ

- PROC FREQ provides one-way, two-way (or more) frequency tables for data with counts and percentages.
- Generally used with categorical variables.

```
PROC FREQ DATA = ...;  
TABLES var1;  
TABLES var1*var2;  
TABLES var1*var2*var3;
```

- Options specified after slash in TABLES statement (see p. 122 for options)

```
TABLES var1*var2 / MISSING;  
(tells SAS to include missing values as part of table)
```

## Summary Tables with PROC TABULATE

- PROC TABULATE similar to PROC FREQ, but tends to produce cleaner-looking tables.

```
PROC TABULATE;
```

```
CLASS _____ ;
```

```
TABLE _____ ;
```

(CLASS statement specifies your classification variables, TABLE statement specifies desired type of table)

- TABLE statements specify up to 3 dimensions:

3-D → pages, rows, columns (in that order)

2-D → rows, columns (in that order)

1-D → columns

### Some Keywords:

- `ALL`: include totals for that classification variable
- With a `VAR` statement, you can specify a continuous variable. Certain keywords (`MEAN`, `STDDEV`, `MAX`, `MIN`) will give statistics for that variable, typically by groups specified in the `TABLE` statement. (see pg. 126 for more keywords)
- `"BOX = "` allows you to write text in the normally blank upper-left box.
- `"MISSTEXT = "` allows you to specify text to go in cells with no data counts (empty cells).

Sections 4.16, 4.17 deal with adjusting header appearance and variable format in `PROC TABULATE`.



## Summary Reports with PROC REPORT

- Similar to other procedures (`PRINT`, `MEANS`, `FREQ`, `SORT`) but produces clean-looking reports as output.

- Syntax:

```
PROC REPORT DATA = ... NOWINDOWS;  
    COLUMN var1 var2 ...;
```

- `COLUMN` tells SAS which variables (and in which order) to include in the report.
- Other useful options on `PROC REPORT` line:

`HEADSKIP` → skips a line after headers

`HEADLINE` → puts horizontal line after headers

- If character variables included in `COLUMN` statement, report consists of one row for each observation.
- If only numeric variables included, reports consist of variable sums, unless `DISPLAY` option specified.

- `DEFINE` statements allow the use of several options in `PROC REPORT` (including `DISPLAY`).
- `ORDER` arranges rows in the order of the values of the specified variable.
- `GROUP` creates a separate row for each level of specified variable.
- `ACROSS` creates a separate column for each level of specified variable.
- With `GROUP`, sums of the other variables are given for each level of grouping variable.
- With `ACROSS`, need a special syntax to produce such sums.

Other statements:

- `BREAK` provides summary breaks at each level of a variable.
- `RBREAK` provides summary break for the entire report.

- While many reports contain sums of values of the numeric variables, other summary statistics can be printed as well.
- These are typically specified in the COLUMN statement:

```
        COLUMN varname, KEYWORD;  
        COLUMN (varname1 varname2), KEYWORD;  
        COLUMN varname, (KEYWORD1, KEYWORD2);
```

where KEYWORD specifies the desired summary statistic(s) MEAN, MEDIAN, MIN, MAX, STD; — see list on pg. 142.

## Adding Computed Variables to PROC REPORT Output

- COMPUTE blocks begin with a COMPUTE statement and end with a ENDCOMP statement.
- Syntax:

```
DEFINE newvar / COMPUTED;  
COMPUTE newvar / <options>;  
    . . . ;  
ENDCOMP;
```

- If character variables are computed in COMPUTE statement, use CHAR option, and it's good to include the LENGTH option (otherwise, default length is 8).
- If a numeric variable are used with an analysis in the COMPUTE block, then variable name is appended with a statistic (the default is SUM).
- You can exclude the original variables (which the computed variables are based on) from the report with the NOPRINT option in a DEFINE statement.