Ch4. Sorting, Printing, and Summarizing Your Data (PROC step)

1. PROC Procedure

- PROC Statement

  General form:

  > PROC proc-name

- BY Statement

  Example:

  > BY State

  run a separate analysis for each state (sorted data).

- TITLE Statement

  General form:

  > TITLE 'this is title';

- FOOTAGE Statement

  General form:

  > FOOTNOTE3 'This is the third footnote';

  Can specify up to 10 titles or footnotes by adding numbers to TITLE and FOOTNOTE.

- LABEL Statement

  General form:

  > LABEL var = 'description of this var'

2. PROC SORT

General form:

```
PROC SORT DATA = data OUT = data_out  NODUPKEY  DUPOUT = extraobs
SORTSEQ=option;
    BY DESCENDING var;
    /* ASCENDING by default */
```

If you don't specify `DATA= option`, then SAS will use the most recently created data set.

> If you don't specify `OUT= option`, then SAS will replace the original data with the newly sorted version.
>
> `NODUPKEY` option tells SAS to eliminate any duplicate observations that have the same values for the BY variables.
>
> If you specify the `DUPOUT= option`, then SAS will put the deleted observations in that data set.

Example:

```
beluga   whale 15   dwarf    shark .5    sperm    whale 60
basking  shark 30   humpback   .   50    whale    shark 40
gray     whale 50   blue     whale 100   killer   whale 30
mako     shark 12   whale    shark 40
```

```
DATA marine;
   INFILE 'c:\MyRawData\Lengths.dat';
   INPUT Name $ Family $ Length @@;
RUN;
* Sort the data;
PROC SORT DATA = marine OUT = seasort NODUPKEY;
   BY Family DESCENDING Length;
PROC PRINT DATA = seasort;
   TITLE 'Whales and Sharks';
RUN;
```

```
                        Whales and Sharks            1

                Obs   Name         Family    Length

                 1    humpback                 50.0
                 2    whale        shark       40.0
                 3    basking      shark       30.0
                 4    mako         shark       12.0
                 5    dwarf        shark        0.5
                 6    blue         whale      100.0
                 7    sperm        whale       60.0
                 8    gray         whale       50.0
                 9    killer       whale       30.0
                10    beluga       whale       15.0
```

- `SORTSEQ` changing the Sort Order For Character Variables
  - `SORTSEQ = ASCII` VS `SORTSEQ = EBCDIC`

ASCII	Blank	Numerals	Uppercase letters	Lowercase letters
EBCDIC	Blank	Uppercase letters	Lowercase letters	Numerals

- `SORTSEQ=LINGUISTIC`
  - `SORTSEQ=LINGUISTIC (strength = primary)`

Sales sorted by Customer

Obs	EmpID	Name	Region	Customer	Item	Quantity	UnitCost
1	1843	George Smith	North	Barco Corporation	144L	50	8.99
2	1843	George Smith	North	Barco Corporation	908X	1	5129.00
3	0177	Glenda Johnson	East	Barco Corporation	733	2	10000.00
4	1843	George Smith	South	Cost Cutter's	122	100	5.99
5	1843	George Smith	South	Cost Cutter's	855W	1	9109.00
6	9888	Sharon Lu	West	Cost Cutter's	122	50	5.99
7	1843	George Smith	South	Ely Corp.	122L	10	29.95
8	0177	Glenda Johnson	East	Food Unlimited	188X	100	6.99
9	1843	George Smith	North	Minimart Inc.	188S	3	5199.00
10	0177	Glenda Johnson	North	Minimart Inc.	777	5	10.50
11	1843	George Smith	North	Minimart Inc.	188S	3	5199.00

  - `SORTSEQ=LINGUISTIC (Numeric_collation = on)`

Sales sorted by item

Obs	EmpID	Name	Region	Customer	Item	Quantity	UnitCo
1	9888	Sharon Lu	West	Pet's are Us	100W	1000	1.9
2	1843	George Smith	South	Cost Cutter's	122	100	5.9
3	9888	Sharon Lu	West	Cost Cutter's	122	50	5.9
4	1843	George Smith	South	Ely Corp.	122L	10	29.9
5	0017	Jason Nguyen	East	Roger's Spirits	122L	500	39.9
6	1843	George Smith	North	Barco Corporation	144L	50	8.9
7	0177	Glenda Johnson	East	Shop and Drop	144L	100	8.9
8	1843	George Smith	North	Minimart Inc.	188S	3	5199.0
9	1843	George Smith	North	Minimart Inc.	188S	3	5199.0
10	0177	Glenda	East	Food Unlimited	188X	100	6.9

3. PROC PRINT

```
PROC PRINT DATA = data-set NOOBS LABEL;
```

> `NOOBS` : no observation number
>
> `LABEL` : print the labels instead of the variable names

PROC PRINT options	Explanation
`BY variable-list;`	output a new section for each value of the presorted BY variables.
`ID variable-list;`	the variables in the ID variable list appear on the left-hand side of the page.
`SUM variable-list;`	prints sums for the variables in the list.
`VAR variable-list;`	which variables to print and the order (default as data set).

Example:

```
data selthree;
    set lecture.Sales;
proc sort data=selthree;
    by Name;
proc print data=selthree;
    by Name;
    sum Quantity;
    var Region Quantity UnitCost;
    title 'Sales by EmplID';
run;
```

Sales by EmpID

Name=George Smith

Obs	Region	Quantity	UnitCost
1	North	50	8.99
2	South	100	5.99
3	North	3	5199.00
4	North	1	5129.00
5	South	10	29.95
6	South	1	9109.00
7	North	3	5199.00
Name		168	

Name=Glenda Johnson

Obs	Region	Quantity	UnitCost
8	East	100	6.99
9	East	100	8.99
10	North	5	10.50
11	East	2	10000.00
Name		207	

Name=Jason Nguyen

Obs	Region	Quantity	UnitCost
12	East	500	39.99
13	South	100	19.95
Name		600	

- **Change the Appearance of Printed Values**
  - **PROC FORMAT**

    General form:

Character	Numeric	Date
$format.	formatw.d	formatw.

Create format:

```
PROC FORMAT;
    VALUE name range-1 = 'formatted text 1'
               range-2= 'formatted text 2'
               /* ..................................... */
               range-n = 'formatted text n' ;
```

Example:

```
DATA carsurvey;
       INPUT Age Sex Income Color $;
       datalines;
       19 1 14000 Y
       45 1 65000 G
```

```
        72 2 35000 B
        31 1 44000 Y
        58 2 83000 W
        ;
PROC FORMAT;
        VALUE gender 1 = 'Male'
                     2 = 'Female';
        VALUE agegroup 13 -< 20 = 'Teen'
                       20 -< 65 = 'Adult'
                       65 - HIGH = 'Senior';
        VALUE $col  'W' = 'Moon White'
                    'B' = 'Sky Blue'
                    'Y' = 'Sunburst Yellow'
                    'G' = 'Rain Cloud Gray';
* Print data using user-defined and standard (DOLLAR8.) formats;
PROC PRINT DATA = carsurvey;
        FORMAT Sex gender. Age agegroup. Color $col. Income DOLLAR8.;
        TITLE 'Survey Results Printed with User-Defined Formats';
RUN;
```

```
          Survey Results Printed with User-Defined Formats         1

        Obs     Age       Sex        Income     Color

         1      Teen      Male       $14,000    Sunburst Yellow
         2      Adult     Male       $65,000    Rain Cloud Gray
         3      Senior    Female     $35,000    Sky Blue
         4      Adult     Male       $44,000    Sunburst Yellow
         5      Adult     Female     $83,000    Moon White
```

- FORMAT Statement

  > Format A B DOLLAR8.2 C MMDDYY8.

  Variable A and B use DOLLAR8.2 format, and variable C uses MMDDYY8.

- PUT Statement

  > Put A DOLLAR8.2 B DOLLAR8.2 C MMDDYY8.

  Place a format after each variable name

4. REPORT

(1) Simple Report

In data step: Instead of using an INFILE statement, you use a FILE statement; instead of INPUT statements, you use PUT statements.

- **FILE Statement**

  `FILE 'file – specification' PRINT;`

  Specifies the current output file for PUT statements.

  `PRINT` directs the output to the same file as the output that is produced by SAS procedures.

- **PUT Statement**

  Writes lines to the SAS log, to the SAS output window, or to an external location that is specified in the most recent FILE statement.

  Control spacing:

@n	move to column n
+n	move n columns
/	skip to the next line
#n	skip to line n
@	hold the current line
" " or ' '	enclose a text

  Example:

```
data _NULL_;
    set Lec9.Sales;
    file 'J:\CLASSES\STAT46\Customer.txt' PRINT;
    title;
    /* The null TITLE statement tells SAS to eliminate all automatic
titles. */
    PUT @5 'Sales report for ' Name 'from region ' Region 'customer of '
customer           // @5 'your total sales are ' TotalSales;
    Put _Page_;
    /* inserts a page break after each student's report */
run;
```

  the external file *Customers.txt*:

```
Sales report for George Smith from region North customer of Barco Corporation
your total sales are 449.5
Sales report for George Smith from region South customer of Cost Cutter's
your total sales are 599
Sales report for George Smith from region North customer of Minimart Inc.
your total sales are 15597
```

(2) PROC REPORT

General form:

```
PROC REPORT NOWINDOWS;
COLUMN variable-list;
```

> COLUMN (similar to VAR): variables to include and in what order.
>
> > NOWINDOWS: If leave it out, SAS will open the interactive Report window.

Defaults:

Data with all Numeric variable	Data with one Character variable
PROC REPORT will sum the variables (1 row)	one row per observation

- DEFINE Statements

  General form:

  > DEFINE variable / options 'column-header';

Changing column headers	Keeping Missing data
`DEFINE Age/ORDER 'Age at/Admission' WIDTH = 9;`	`PROC REPORT NOWINDOWS MISSING;`

  you specify the variable name followed by a slash and any options for that particular variable.

  - Usage Options

Options	Explanation
ACROSS	creates a column for each unique value of the variable
ANALYSIS	calculates statistics for the variable (default: sum)
DISPLAY	creates one row for each variable in the data set.
GROUP	creates one row for each unique value of the variable.
ORDER	Creates one row for each observation with rows arranged according to the values of the order variable.

Example:

```
Proc report data = medic NOWIMDOWS;
    column Clinic VisitDate Weight HR;
    define Clinic / group;
    define weight / analysis;
    title 'Medical data arranged by Clinic group';
run;

Proc report data = selthree NOWIMDOWS;
    column Region Quantity TotalSales;
    define Region / group;
    define TotalSales / analysis "Total Sales"; /* change header name */
    title 'Total Sales by Region';
run;
```

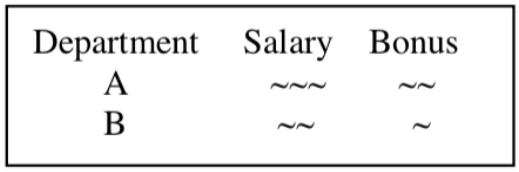
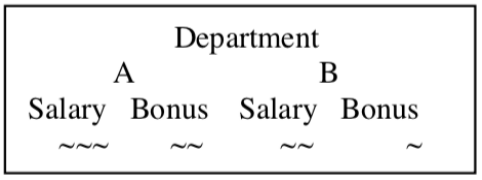
Medical data arranged by Clinic group

Clinic	Visit Date	Weight	Heart Rate
HMC	01/15/2100	653	206
Mayo Clinic	11/19/2099	510	200

Total Sales by Region

Region	Quantity	Total Sales
East	702	41593
North	62	36825
South	211	12002.5
West	1050	2289.5

- Create Summary Report

Group Variable	Across Variable
`DEFINE group-var / GROUP;`	`DEFINE across-var/ ACROSS;`
	

Example:

```
DATA natparks;
       INFILE 'c:\MyRawData\Parks.dat';
       INPUT Name $ 1-21 Type $ Region $ Museums Camping;
RUN;
* Region and Type as GROUP variables;
PROC REPORT DATA = natparks NOWINDOWS HEADLINE;
    COLUMN Region Type Museums Camping;
    DEFINE Region / GROUP;
    DEFINE Type / GROUP;
    TITLE 'Summary Report with Two Group Variables';
RUN;
* Region as GROUP and Type as ACROSS with sums;
PROC REPORT DATA = natparks NOWINDOWS HEADLINE;
    COLUMN Region Type,(Museums Camping);
    DEFINE Region / GROUP;
    DEFINE Type / ACROSS;
    TITLE 'Summary Report with a Group and an Across Variable';
RUN;
```

```
Summary Report with Two Group Variables                    1

Region    Type        Museums    Camping
──────────────────────────────────────────
East      NM                2          0
          NP                8         12
West      NM                3          7
          NP               18         29


 Summary Report with a Group and an Across Variable        2

                              Type
                    NM                        NP
Region       Museums     Camping     Museums     Camping
──────────────────────────────────────────────────────────
East               2           0           8          12
West               3           7          18          29
```

- Adding Summary Breaks

  General form:

  `BREAK location variable / options;`

  BREAK statement: adds a break for each unique value of the specified variable.

  `RBREAK location / options;`

  RBREAK statement: does the same as BREAK for the entire report.

  `location`: two possible values *BEFORE* or *AFTER* depending on whether you want the break to precede or follow that particular section of the report.

  `options`: possible kind of break to insert

OL	draws a line over the break
PAGE	starts a new page
SKIP	inserts a blank line
SUMMARIZE	inserts sums of numeric variables
UL	draws a line under the break

Example:

```
Dinosaur                NM West 2  6
Ellis Island            NM East 1  0
Everglades              NP East 5  2
Grand Canyon            NP West 5  3
Great Smoky Mountains   NP East 3 10
Hawaii Volcanoes        NP West 2  2
Lava Beds               NM West 1  1
Statue of Liberty       NM East 1  0
Theodore Roosevelt      NP .    2  2
Yellowstone             NP West 9 11
Yosemite                NP West 2 13
```

```
DATA natparks;
       INFILE 'c:\MyRawData\Parks.dat';
       INPUT Name $ 1-21 Type $ Region $ Museums Camping;
RUN;
    * PROC REPORT with breaks;
    PROC REPORT DATA = natparks NOWINDOWS HEADLINE;
       COLUMN Name Region Museums Camping;
       DEFINE Region / ORDER;
       BREAK AFTER Region / SUMMARIZE OL SKIP;
       RBREAK AFTER / SUMMARIZE OL SKIP;
       TITLE 'Detail Report with Summary Breaks';
RUN;
```

Detail Report with Summary Breaks 1

Name	Region	Museums	Camping
Ellis Island	East	1	0
Everglades		5	2
Great Smoky Mountains		3	10
Statue of Liberty		1	0
	East	10	12
Dinosaur	West	2	6
Grand Canyon		5	3
Hawaii Volcanoes		2	2
Lava Beds		1	1
Yellowstone		9	11
Yosemite		2	13
	West	21	36
		31	48

5. PROC MEANS

General form:

> PROC MEANS options;

Summary Options:

MAX	the maximum value
MIN	the minimum value
MEAN	the mean
MEDIAN	the median
MODE	the mode
N	number of non-missing values
NMISS	number of missing values
RANGE	the range
STDDEV	the standard deviation
SUM	the sum

Control Options:

MAXDEC = n	specifies the number of decimal places to be displayed
MISSING	treats missing values as valid summary groups

Optional Statements:

BY variable-list;	performs separate analyses for each level of listed variables. (sorted data)
CLASS variable-list;	Similar but more compact output than BY statement. (no matter sorted or not)
VAR variable-list;	specifies which numeric variables to use in the analysis (default all).

Example:

```
proc means data = Learn.Blood;
/* PROC MEANS with no other statements will get statistics for all
observations and all numeric variables in the data set. */
run;
```

PROC MEANS With All the Defaults

The MEANS Procedure

Variable	Label	N	Mean	Std Dev	Minimum	Maximum
Subject		1000	500.5000000	288.8194361	1.0000000	1000.00
WBC		908	7042.97	1003.37	4070.00	10550.00
RBC		916	5.4835262	0.9841158	1.7100000	8.7500000
Chol	Cholesterol	795	201.4352201	49.8867157	17.0000000	331.0000000

- OUTPUT Statement

  General form:

  > OUTPUT OUT = data-set output-statistic-list;

  `data-set`: name of the SAS data set that will contain the results (either temporary or permanent)

  `output-statistic-list`: defines which statistics you want and the associated variable names.

  - Multiple OUTPUT statements:

    ```
    OUTPUT OUT = data-set
    statistics (variable – list) = name-list;
    ```

    `variable – list`: defines which of the variables in the VAR list you want to output.

    Example:

    ```
    PROC MEANS DATA = zoo NOPRINT;
        VAR Lions Tigers Bears;
        /* no BY or CLASS statement, then the data set will have just one
    observation.*/
        OUTPUT OUT = zoosum /* two extra variables _TYPE_ and _FREQ_ */
        MEAN = LionWeight BearWeight TigerWeight;
        /* MEAN (Lions Tigers Bears) = LionWeight BearWeight TigerWeight;
    */
    RUN;
    ```

    > BY statement: the data set will have one observation for each level of the BY group.

> CLASS statements: one observation for each level of interaction of the class variables.

Detailed example:

```
DATA sales;
       INPUT CustomerID $ @9 SaleDate MMDDYY10. Petunia SnapDragon
Marigold;
       DATALINES;
       756-01  05/04/2008 120  80 110
       834-01  05/12/2008  90 160  60
       901-02  05/18/2008  50 100  75
       834-01  06/01/2008  80  60 100
       756-01  06/11/2008 100 160  75
       901-02  06/19/2008  60  60  60
       756-01  06/25/2008  85 110 100
       ;
PROC SORT DATA = sales;
       BY CustomerID;
* Calculate means by CustomerID, output sum and mean to new data set;
PROC MEANS NOPRINT DATA = sales;
       BY CustomerID;
       VAR Petunia SnapDragon Marigold;
       OUTPUT OUT = totals
       MEAN(Petunia SnapDragon Marigold) = MeanPetunia MeanSnapDragon
MeanMarigold
       SUM(Petunia SnapDragon Marigold) = Petunia SnapDragon
Marigold;
PROC PRINT DATA = totals;
       TITLE 'Sum of Flower Data over Customer ID';
       FORMAT MeanPetunia MeanSnapDragon MeanMarigold 3.;
RUN;
```

Sum of Flower Data over Customer ID 1

Obs	Customer ID	_TYPE_	_FREQ_	Mean Petunia	Mean Snap Dragon	Mean Marigold	Snap Petunia	Snap Dragon	Snap Marigold
1	756-01	0	3	102	117	95	305	350	285
2	834-01	0	2	85	110	80	170	220	160
3	901-02	0	2	55	80	68	110	160	135

6. PROC FREQ

[1] Create tables showing the distribution of categorical data values .

[2] Reveal irregularities in your data. (data entry errors)

General form:

```
PROC FREQ;
    TABLES variable-combinations;
```

One-way frequency table: `TABLES YearsEducation;`

Cross-tabulation: `TABLES Sex * YearsEducation;` (list the variables separated by an asterisk.)

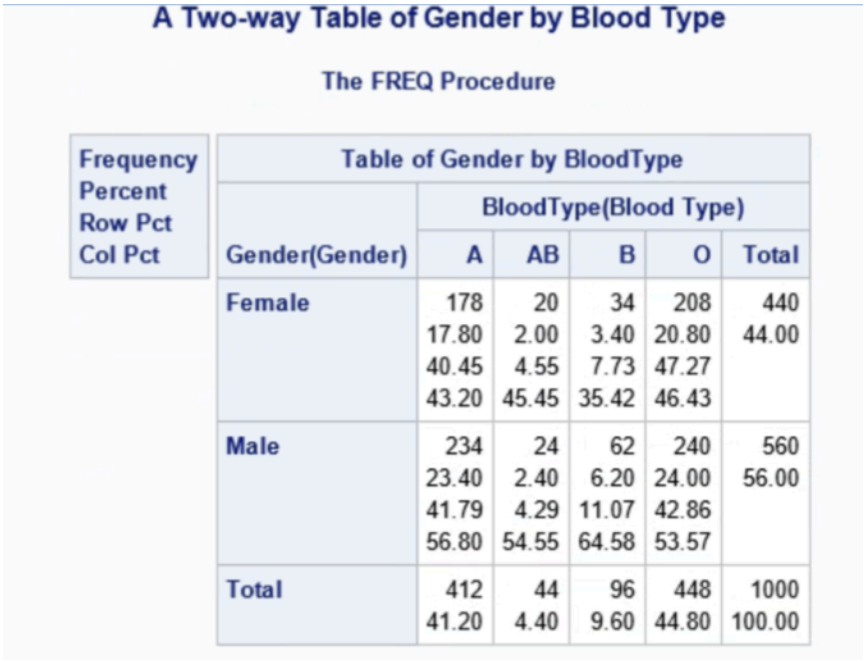
PROC FREQ options:

options appear after a slash `/` in the TABLES statement.

LIST	Prints cross tabulations in list format rather than grid
MISSPRINT	Includes missing values in frequencies but not in percentages
MISSING	Includes missing values in frequencies and in percentages
NOCOL	Suppresses printing of column percentages in cross-tabulations
NOPERCENT	Suppresses printing of percentages
NOROW	Suppresses printing of row percentages in cross-tabulations
OUT = *data - set*	Writes a data set containing frequencies

Example:

```
title "A Two-way Table of Gender by Blood Type";
proc freq data = Learn.Blood;
    tables Gender * BloodType;
run;
```



A Two-way Table of Gender by Blood Type

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Gender by BloodType				
	BloodType(Blood Type)				
Gender(Gender)	A	AB	B	O	Total
Female	178 17.80 40.45 43.20	20 2.00 4.55 45.45	34 3.40 7.73 35.42	208 20.80 47.27 46.43	440 44.00
Male	234 23.40 41.79 56.80	24 2.40 4.29 54.55	62 6.20 11.07 64.58	240 24.00 42.86 53.57	560 56.00
Total	412 41.20	44 4.40	96 9.60	448 44.80	1000 100.00

7. PROC TABULATE

Every summary statistic the TABULATE procedure computes can also be produced by other procedures such as PRINT, MEANS, and FREQ, but PROC TABULATE is popular because its reports are pretty.

General form:

```
PROC TABULATE;
    CLASS classification-variable-list;
    TABLE page-dimension, row-dimension, column-dimension;
```

`CLASS` tells which variables contain categorical data to be used for dividing observations into groups `TABLE` tells how to organize your table and what numbers to compute.

Dimensions: If you specify only one dimension, then that becomes, by default, the column dimension. If you specify two dimensions, then you get rows and columns, but no page dimension. If you specify three dimensions, then you get pages, rows, and columns.

Missing data: by default, observations are excluded from tables if they have missing values for variables listed in a CLASS statement. If you want, use `PROC TABULATE MISSING;` to include those observations.

Example:

```
title "The Effect of Missing Values on Class variables";
proc tabulate data = Learn.Missing format = 4.missing;
    class A B;
    table A ALL B ALL;
run;
```

The Effect of Missing Values on CLASS variables

	B			All
	X	Y	Z	
	N	N	N	N
A				
X	1	2	.	4
Y	.	.	1	1
Z	.	.	1	1
All	1	2	2	6

(1) Add statistics to output

```
PROC TABULATE;
    VAR analysis-variable-list;
    CLASS classification-variable-list;
    TABLE page-dimension, row-dimension, column-dimension;
```

`VAR` tells SAS which variables contain continuous data.

Keywords:

ALL	adds a row, column, or page showing the total
MAX	highest value
MIN	lowest value
MEAN	the arithmetic mean
MEDIAN	the median
MODE	the mode
N	number of non-missing values
NMISS	number of missing values
PCTN	the percentage of observations for that group
PCTSUM	the percentage of a total sum represented by that group
STDDEV	the standard deviation
SUM	the sum

Concatenating, crossing, and grouping:

Concatenating	`TABLE Locomotion Type ALL;`
Crossing	`TABLE MEAN * Price;`
Crossing, grouping, and concatenating	`TABLE PCTN *(Locomotion Type);`

Example:

```
title "Computing Percentage on a Numerical Value";
proc tabulate data = Learn.Sales;
    class Region;
    var TotalSales;
    table (Region ALL), /* rows */
          TotalSales * (n*f = 6. sum*f = dollar6. pctum*f = pctfmt7.);
          /* columns; f: format */
    keylabel ALL = "All Regions"
             n = "Number of Sales"
             sum = "Sum"
             pctsum = "Percent";
    label TotalSales = "Total Sales";
run;
```

Computing Percentages on a Numerical Value

	Total Sales		
	Number of Sales	Sum	Percent
Region			
East	4	$41,593	44.8%
North	5	$36,825	39.7%
South	4	$12,003	12.9%
West	2	$2,290	2.4%
All Regions	15	$92,710	100.0%

(2) Enhancing the Appearance of output

- `FORMAT = option`
  - to be used in the PROC statement
  - It changes the format of all the data cells in the table

    General form: `PROC TABULATE FORMAT = Comma10.0;`
- `BOX = option`
  - To be used in the TABLE statements
  - It allows to write a brief phrase in the upper left corner box of every TABULATE report

    General form: `TABLE Region, MEAN*sales / BOX = 'Mean sales by Region';`
- `MISSTEXT = option`
  - To be used in the TABLE statement.
  - It specifies a value for SAS to print in empty data cells

    General form: `TABLE region, MEAN*sales / MISSTEXT = 'No Sales';`

Example:

```
DATA boats;
    INPUT Name $ 1-12 Port $ 14-20 Locomotion $ 22-26 Type $ 28-30 Price 32-
36;
    DATALINES;
    Silent Lady  Maalea  sail  sch 75.00
    America II   Maalea  sail  yac 32.95
    Aloha Anai   Lahaina sail  cat 62.00
    Ocean Spirit Maalea  power cat 22.00
    Anuenue      Maalea  sail  sch 47.50
```

```
    Hana Lei     Maalea  power cat 28.99
    Leilani      Maalea  power yac 19.99
    Kalakaua     Maalea  power cat 29.50
    Reef Runner  Lahaina power yac 29.95
    Blue Dolphin Maalea  sail  cat 42.95
    ;
RUN;
* PROC TABULATE report with options;
PROC TABULATE DATA = boats FORMAT=DOLLAR9.2;
    CLASS Locomotion Type;
    VAR Price;
    TABLE Locomotion ALL, MEAN*Price*(Type ALL)/BOX='Full Day Excursions'
MISSTEXT='none';
    TITLE;
RUN;
```

Full Day Excursions	Mean			
	Price			
	Type			
	cat	sch	yac	All
Locomotion				
power	$26.83	none	$24.97	$26.09
sail	$52.48	$61.25	$32.95	$52.08
All	$37.09	$61.25	$27.63	$39.08

(3) Changing Headers

| CLASS variable values | 1. use `PROC FORMAT` to create a user-defined format
2. assign the format to the variable in FORMAT statement. | | ---------- | ---------- | | **BY variable values** | `TABLE Region='', MEAN=''*Sales='Mean Sales by Region';` |

Multiple Formats for Data Cells: `Variable- name * FORMAT = format.d`

```
TABLE Region, MEAN * (SALES*FORMAT = COMMA8.0
Profit*FORMAT=DOLLAR1010.2);
```

Example (same data):

```
PROC FORMAT;
```

```
    VALUE $typ  'cat' = 'catamaran'
                'sch' = 'schooner'
                'yac' = 'yacht';
RUN;
PROC TABULATE DATA = boats FORMAT=DOLLAR9.2;
    CLASS Locomotion Type;
    VAR Price;
    FORMAT Type $typ.;
    TABLE Locomotion='' ALL,
          MEAN=''*Price='Mean Price by Type of Boat'*(Type='' ALL)
          /BOX ='Full Day Excursions' MISSTEXT='none';
    TITLE;
RUN;
```

Full Day Excursions	Mean Price by Type of Boat			
	catamaran	schooner	yacht	All
power	$26.83	none	$24.97	$26.09
sail	$52.48	$61.25	$32.95	$52.08
All	$37.09	$61.25	$27.63	$39.08