# Steps to Success with PROC MEANS

**Andrew H. Karp**
Sierra Information Services
19229 Sonoma Hwy #264
Sonoma, CA 95476
707 996 7380
andrew@sierrainformation.com
www.SierraInformation.com




## Copyright and Trademark Information

- SAS is a registered trademark of SAS Institute, Inc. in the USA and other countries. ® indicates USA registration.
- 

## *Steps to Success with PROC MEANS*

- Very powerful BASE SAS Procedure
- Analyzes *numeric variables*
  - *Calculates univariate statistics*
- Analyses (Output) stored in
  - Output Window (Default)
  - SAS Data Sets (Optional)
- Why Use PROC MEANS?




## *Why Use PROC MEANS?*

- Calculate statistics on values of numeric variables
- Prepare “data marts” or analysis data sets for subsequent analyses
- Generate reports
- Create SAS data sets for use in other tasks
- Explore data prior to applying other SAS capabilities

## *PROC MEANS: Basic Steps*

- PROC MEANS is a BASE SAS Procedure
- Identical to PROC SUMMARY as of Version 6.0 (released in 1991)
- Many features/enhancements added in SAS 8 and a few more in SAS 9 Software
- **Defaults**:
  - Analyzes all numeric variables in a SAS data set
  - Presents results in the Output Window




## *PROC MEANS: Key Terms and Concepts*

- **Input Data Set**:
  - SAS data set PROC MEANS will analyze
- **Analysis Variables**:
  - Numeric Variables whose values will be analyzed by PROC MEANS
- **Statistics Keywords**:
  - The statistical analyses PROC MEANS will generate
- **Output Data Set**:
  - SAS data set created by PROC MEANS containing the analyses (optional)
- **Classification Variables**:
  - Numeric or character variables whose values will be used to "classify" or "subgroup" the analyses

## *Step 1: Getting Started*

- Example Data Set
  - 16,400 rows/observations from an electrical utility
    - One observation = one year's data from one customer on electrical usage, rate schedule, billing, etc.
    - Twelve Monthly Variables
      - KWH = number of kilowatts (KWH) used that month
      - REV = revenue billed that month




## *Step 1: Getting Started*

- Task: Analyze KWH and REV for January

```
2  options nodate nonumber nocenter orientation=landscape;
3
4  libname andrew  "C:\Documents and Settings\Owner\My Documents\PROC MEANS";
5
6  title 'Steps to Success with PROC MEANS';
7
8  * Ex. 1: Defaults;
9  proc means data=andrew.electricity;
10 var kwh1 rev1;
11 title2 'Example 1: Default PROC MEANS Output';
12 run;
```

Input Data Set

Analysis Variables

## *More on Analysis Variables*

- Placed in the **VAR Statement**
  - Must be stored as numeric variables in the input data set
    - Otherwise, PROC MEANS will not execute and errors will appear in your SASLOG
  - If you omit the VAR Statement from your PROC MEANS task then ALL numeric variables in the input data set will be analyzed and have statistics computed on them
    - Almost always unnecessary and wasteful of computing resources




## *Step 1: Results in the Output Window*

**Five default statistics: N, Mean, Standard Deviation, Minimum, Maximum**

Steps to Success with PROC MEANS
Example 1: Default PROC MEANS Output

The MEANS Procedure

| Variable | N | Mean | Std Dev | Minimum | Maximum |
|---|---|---|---|---|---|
| KWH1 | 16238 | 538.4494396 | 1036.51 | 0 | 65557.00 |
| REV1 | 16243 | 63.7236414 | 86.4849402 | 0 | 2798.92 |

**N is the number of observations in the Input Data Set with a non-missing value of the Analysis Variable**

## *Step 2: Take Control!*

- PROC MEANS has many options and statements you can use to control the analytic processes it carries out
  - Select the observations from the Input Data Set to be analyzed
  - Choose the statistics to be calculated
  - Control the display of the results in the Output Window
  - Store results in a new SAS data set




## *Step 2: Take Control!*

- Tasks:
  - Calculate Default Statistics for EASTERN Region Customers
  - Add Variable Labels to the Output.
  - Round displayed results to two decimal places

# Step 2: Take Control!

MAXDEC Option specifies maximum number of decimal places

```
14 * Ex 2: MAXDEC and Labels, WHERE Statement ;
15 proc means data=andrew.electricity maxdec=2;
16 where region = 'EASTERN';
17 var kwh1 rev1;
18 label kwh1 = 'January KWH'
19       rev1 = 'January Revenue';
20 title2 'Example 2: MAXDEC Option and Labels';
21 title3 'Analysis of Eastern Region Customers Only';
22 run;
```

WHERE Statement selects rows/observations for analysis

LABEL Statement Provides descriptive information for the Analysis Variable names. Can be up to 256 characters long.




# Step 2: Results in the Output Window

Effect of specifying MAXDEC=2 in the PROC MEANS Statement. Displayed results of statistics are rounded to two decimal places.

Steps to Success with PROC MEANS
Example 2: MAXDEC Option and Labels
Analysis of Eastern Region Customers Only

The MEANS Procedure

| Variable | Label | N | Mean | Std Dev | Minimum | Maximum |
|---|---|---|---|---|---|---|
| KWH1 | January KWH | 5077 | 489.00 | 998.56 | 0.00 | 26040.00 |
| REV1 | January Revenue | 5077 | 57.95 | 105.23 | 0.00 | 2798.92 |

Variable labels specified in the LABEL Statement. By default, Variable Labels in the Input Data Set's Descriptor Portion are displayed in the PROC MEANS output. You can specify your own Variable Labels using the LABEL Statement in the PROC step, as shown on the previous slide.

## *A Comment about the WHERE Statement*

- The WHERE Statement selects the rows, or observations, for analysis by PROC MEANS from the Input Data Set
  - Very similar to the WHERE Clause Data Set Option, which will be demonstrated later in this presentation.
- Both are very powerful and both can be used in ANY SAS Procedure that reads a SAS data set
  - Both avoid creating a “subset” data set from a larger file before conducting the analyses you need from the subset
- Remember that testing values of character variables is CaSe-SenSItiVe !




## *Step 3:*
## *Select the Statistics You Want*

- Effective with the release of SAS 9.2 Phase 1 Software, PROC MEANS can calculate 32 statistics
  - Categories:
    - Descriptive
    - Quantile
    - Hypothesis Testing

## *Step 3:*
## *Select the Statistics You Want*

- Statistics KEYWORDS identify the analyses you want PROC MEANS to peform on the Analysis Variables specified in the VAR Statement
  - Pick as many, or as few, as you need/want




## *Descriptive Statistics Keywords*

| | |
|---|---|
| **CLM** | **RANGE** |
| **CSS** | **SKEWNESS \|SKEW** |
| **CV** | **STDDEV\| STD** |
| **KURTOSIS\|KURT** | **STDERR** |
| **LCLM** | **SUM** |
| **MAX** | **SUMWGT** |
| **MEAN** | **UCLM** |
| **MIN** | **USS** |
| **N** | **VAR** |
| **NMISS** | **MODE** |

## Quantile Statistics Keywords

| | |
|---|---|
| MEDIAN\|P50 | Q3\|P75 |
| P1 | P90 |
| P5 | P95 |
| P10 | P99 |
| Q1\|P25 | QRANGE |

**Quantile Statistics were added to PROC MEANS with the release of SAS Version 8 Software. PROCs MEANS, SUMMARY, REPORT, TABULATE and UNIVARIATE now compute a common "suite" of statistics.**




## Hypothesis Testing Keywords

| | |
|---|---|
| PROBT | T |

**The T Keyword computes the "single sample" or "paired difference" test and the PROBT Keyword computes the probability value (p-value) associated with the computed value of the T Statistic.**

**A "two-sample" T-test is computed using PROC TTEST in the BASE SAS Module.**

# Step 3: Select the Statistics You Want PROC MEANS to Compute

The WHERE Clause Data Set Option

Selecting Statistics to be Calculated

List-Addressing of Analysis Variables with a Common Prefix

```
24 * Ex 3: Select Statistics;
25 proc means data=andrew.electricity(where=(region = 'EASTERN'))
26     maxdec=0
27     mean median max min n nmiss;
28 var kwh1-kwh6;
29 label kwh1 = 'Jan 2008 KWH'
30       kwh2 = 'Feb 2008 KWH'
31       kwh3 = 'Mar 2008 KWH'
32       kwh4 = 'Apr 2008 KWH'
33       kwh5 = 'May 2008 KWH'
34       kwh6 = 'Jun 2008 KWH';
35 title2 'Example 3: Using the WHERE Clause Data Set Option, Selecting Statistics';
36 title3 'Keywords and List-Addressing of Analysis Variables with a Common Prefix';
37 title4 'in the VAR Statement';
38 title5 'Analysis of KWH1-KWH6 in the Eastern Region';
39 run;
```




# Step 3: Select the Statistics You Want PROC MEANS to Compute

```
Steps to Success with PROC MEANS
Example 3: Using the WHERE Clause Data Set Option, Selecting Statistics
Keywords and List-Addressing of Analysis Variables with a Common Prefix
in the VAR Statement
Analysis of KWH1-KWH6 in the Eastern Region

The MEANS Procedure
```

| Variable | Label | Mean | Median | Maximum | Minimum | N | N Miss |
|---|---|---|---|---|---|---|---|
| KWH1 | Jan 2008 KWH | 489 | 350 | 26040 | 0 | 5077 | 47 |
| KWH2 | Feb 2008 KWH | 463 | 348 | 26880 | 0 | 5089 | 35 |
| KWH3 | Mar 2008 KWH | 462 | 351 | 27240 | 0 | 5090 | 34 |
| KWH4 | Apr 2008 KWH | 539 | 404 | 34320 | 0 | 5098 | 26 |
| KWH5 | May 2008 KWH | 598 | 451 | 35040 | 0 | 5097 | 27 |
| KWH6 | Jun 2008 KWH | 562 | 411 | 36240 | 0 | 5102 | 22 |

## *Step 4: Be Classy with the CLASS Statement*

- CLASS Statement
  - Requests that PROC MEANS "classify" or "group" the analyses it carries out "by" the values of one or more Classification Variables
  - Input Data Set DOES NOT have to be sorted by the values of the variables placed in the CLASS statement
    - NOBS computed/displayed by default




## *Step 4: Be Classy with the CLASS Statement*

```
51 * Ex 4: Class Statement, Selecting Statistics;
52 proc means data=andrew.electricity maxdec=2
53           n nmiss mean median;
54  class region;
55 var kwh1 rev1;
56 label region = 'Region Serving Customer'
57       kwh1 = 'January KWH'
58       rev1 = 'January Revenue';
59 title2 'Example 4: Using the CLASS Statement and Selecting Statistics';
60 run;
```

Classification Variable in the CLASS Statement

## *Step 4:*
## *Be Classy with the CLASS Statement*

Steps to Success with PROC MEANS
Example 4: Using the CLASS Statement and Selecting Statistics

The MEANS Procedure

| Region Serving Customer | N Obs | Variable | Label | N | N Miss | Mean | Median |
|---|---|---|---|---|---|---|---|
| EASTERN | 5124 | KWH1 | January KWH | 5077 | 47 | 489.00 | 350.00 |
| | | REV1 | January Revenue | 5077 | 47 | 57.95 | 40.64 |
| NORTHERN | 5462 | KWH1 | January KWH | 5423 | 39 | 634.83 | 491.00 |
| | | REV1 | January Revenue | 5428 | 34 | 74.03 | 57.49 |
| SOUTHERN | 720 | KWH1 | January KWH | 717 | 3 | 768.82 | 622.00 |
| | | REV1 | January Revenue | 717 | 3 | 91.92 | 70.49 |
| WESTERN | 5075 | KWH1 | January KWH | 5021 | 54 | 451.46 | 361.00 |
| | | REV1 | January Revenue | 5021 | 54 | 54.40 | 42.59 |

**The NOBS (number of observations) column is computed by PROC MEANS by default when you use a CLASS statement.**




## *NOBS, N and NMISS*

- NOBS: Number of observations with the associated value of the ***Classification*** Variable
- N: Number of observations with a non-missing value of the ***Analysis*** Variable
- NMISS: Number of observations with a missing value of the ***Analysis*** Variable

## *Step 5: Be Really Classy with More than One Variable in the CLASS Statement*

- You can use as many Classification Variables as you need/want in a single CLASS statement
  - Multiple CLASS Statements are also permitted; this is covered in my “Beyond the BASICS” talk on PROC MEANS
  - Default: Statistics are computed at all possible ways to combine the NON-MISSING values of the specific Classification Variables




## *Step 5: Be Really Classy with More than One Variable in the CLASS Statement*

```
62 * Ex 5: Use More Than One Class Variable;
63 * Select Obs Where Transformer Model Starts with Letter K;
64 proc means data=andrew.electricity(where=(trans =: 'K'))
65             maxdec=2
66             sum n nmiss;
67 class region trans;
68 var kwh12 rev12;
69 label region = 'Region Serving Customer'
70       kwh12 = 'December KWH'
71       rev12 = 'December Revenue';
72 title2 'Example 5: Using Two Class Variables';
73 run;
```

**Two Classification Variables in the CLASS Statement**

## Step 5: Be Really Classy with More than One Variable in the CLASS Statement

Notice that statistics are calculated for every *non-missing* combination of the values of the Classification Variables. For example, Transformer K12 is only in the Western Region and K1233 is only found in the Northern Region

Steps to Success with PROC MEANS
Example 5: Using Two Class Variables

The MEANS Procedure

| Region Serving Customer | TRANS | N Obs | Variable | Label | Sum | N | N Miss |
|---|---|---|---|---|---|---|---|
| EASTERN | K1211C | 85 | KWH12 | December KWH | 33717.00 | 85 | 0 |
| | | | REV12 | December Revenue | 4786.06 | 85 | 0 |
| | K1233C | 427 | KWH12 | December KWH | 205744.00 | 424 | 3 |
| | | | REV12 | December Revenue | 26033.09 | 424 | 3 |
| NORTHERN | K1211C | 160 | KWH12 | December KWH | 109544.00 | 160 | 0 |
| | | | REV12 | December Revenue | 13618.19 | 160 | 0 |
| | K1233 | 5 | KWH12 | December KWH | . | 0 | 5 |
| | | | REV12 | December Revenue | . | 0 | 5 |
| | K1233C | 455 | KWH12 | December KWH | 267398.00 | 452 | 3 |
| | | | REV12 | December Revenue | 33498.56 | 452 | 3 |
| SOUTHERN | K1211C | 21 | KWH12 | December KWH | 20496.00 | 21 | 0 |
| | | | REV12 | December Revenue | 2611.17 | 21 | 0 |
| | K1233C | 63 | KWH12 | December KWH | 50168.00 | 63 | 0 |
| | | | REV12 | December Revenue | 6172.98 | 63 | 0 |
| WESTERN | K12 | 4 | KWH12 | December KWH | . | 0 | 4 |
| | | | REV12 | December Revenue | . | 0 | 4 |
| | K1211C | 116 | KWH12 | December KWH | 39619.00 | 108 | 8 |
| | | | REV12 | December Revenue | 4931.26 | 108 | 8 |
| | K1233C | 381 | KWH12 | December KWH | 151965.00 | 369 | 12 |
| | | | REV12 | December Revenue | 18871.38 | 369 | 12 |




## Step 6: Don't Miss the MISSINGS!

- In Step 5 we saw that only the non-missing values of the classification variables were displayed in the PROC MEANS-generated output.
- How can we account for/display missing values of the classification variables in our PROC MEANS results?
  - Answer: The MISSING option

# Step 6A: Default Results without the MISSING Option

```
75 * Ex 6: Don't Miss the MISSINGs!;
76 proc means data=andrew.electricity(where=(substr(trans,1,1) NOT IN('C','H')))
77             maxdec=0
78             sum n nmiss;
79 class trans;
80 var kwh12;
81 label region = 'Region Serving Customer'
82       kwh12 = 'December KWH'
83       rev1 = 'December Revenue';
84 title2 'Example 6A: Default Results Without the MISSING Option';
85 run;
```




# Step 6A: Default Results without the MISSING Option

By default, only the non-missing values of the Classification Variable TRANS which satisfy the conditions in the WHERE Clause SAS Data Set Option are displayed in the PROC MEANS-created output. The values of TRANS are ordered in ascending (lowest-to-highest) value.

Steps to Success with PROC MEANS
Example 6A: Default Results Without the MISSING Option

The MEANS Procedure

Analysis Variable : KWH12 December KWH

| TRANS | N Obs | Sum | N | N Miss |
|---|---|---|---|---|
| A4356C | 1674 | 843446 | 1660 | 14 |
| B2348X | 1537 | 849579 | 1525 | 12 |
| D8976V | 663 | 352330 | 656 | 7 |
| E2211U | 2724 | 1441175 | 2699 | 25 |
| J3455Y | 1077 | 544869 | 1065 | 12 |
| K12 | 4 | . | 0 | 4 |
| K1211C | 382 | 203376 | 374 | 8 |
| K1233 | 5 | . | 0 | 5 |
| K1233C | 1326 | 675275 | 1308 | 18 |
| L3333R | 733 | 430903 | 730 | 3 |
| M1211C | 556 | 274579 | 545 | 11 |
| M5671X | 458 | 354691 | 452 | 6 |
| R24 | 4 | 4128 | 4 | 0 |
| R2448Y | 1058 | 518593 | 1050 | 8 |
| XXX | 50 | 30984 | 50 | 0 |

# Step 6B: Using the MISSING Option

```
86
87 proc means data=andrew.electricity(where=(substr(trans,1,1) NOT IN('C','H')))
88              maxdec=0 sum n nmiss missing;
89 class trans;
90 var kwh12;
91 label region = 'Region Serving Customer'
92       kwh12 = 'December KWH'
93       rev1 = 'December Revenue';
94 title2 'Example 6B: With the MISSING Option';
95 run;
```




# Step 6B: Using the MISSING Option

Since missing (or 'blank') sorts 'higher' than numbers or characters, the 'blank' or 'missing' value of TRANS is shown first in the PROC MEANS output.

In this example, 79 customers had power recorded in December from transformers for which the model number is missing in the Input Data Set. Another 196 customers had an unknown value of TRANS and did NOT have power recorded in December.


Steps to Success with PROC MEANS
Example 6B: With the MISSING Option

The MEANS Procedure

Analysis Variable : KWH12 December KWH

| TRANS | N Obs | Sum | N | N Miss |
|---|---|---|---|---|
| | 275 | 55248 | 79 | 196 |
| A4356C | 1674 | 843446 | 1660 | 14 |
| B2348X | 1537 | 849579 | 1525 | 12 |
| D8976V | 663 | 352330 | 656 | 7 |
| E2211U | 2724 | 1441175 | 2699 | 25 |
| J3455Y | 1077 | 544869 | 1065 | 12 |
| K12 | 4 | . | 0 | 4 |
| K1211C | 382 | 203376 | 374 | 8 |
| K1233 | 5 | . | 0 | 5 |
| K1233C | 1326 | 675275 | 1308 | 18 |
| L3333R | 733 | 430903 | 730 | 3 |
| M1211C | 556 | 274579 | 545 | 11 |
| M5671X | 458 | 354691 | 452 | 6 |
| R24 | 4 | 4128 | 4 | 0 |
| R2448Y | 1058 | 518593 | 1050 | 8 |
| XXX | 50 | 30984 | 50 | 0 |

## Step 6C: Order Results with the ORDER=FREQ Option in the CLASS Statement

- Instead of displaying the results in “sort order” of the values of the Classification Variable(s) you specified in the CLASS Statement, order the results by frequency order using the ORDER=FREQ option in the CLASS Statement
  - Added in SAS Version 8




## Step 6C: Order Results with the ORDER=FREQ Option in the CLASS Statement

```
 97 * user ORDER=FREQ in the CLASS Statement;
 98 proc means data=andrew.electricity(where=(substr(trans,1,1) NOT IN('C','H')))
 99              maxdec=0 sum n nmiss missing;
100 class trans/order = freq;
101 var kwh12;
102 label region = 'Region Serving Customer'
103       kwh12 = 'December KWH'
104       rev1 = 'December Revenue';
105 title2 'Example 6C: With the MISSING Option and ORDER=FREQ in the CLASS Statement';
106 run;
```

## Step 6C: Order Results with the ORDER=FREQ Option in the CLASS Statement

Steps to Success with PROC MEANS
Example 6C: With the MISSING Option and ORDER=FREQ in the CLASS Statement

The MEANS Procedure

Analysis Variable : KWH12 December KWH

| TRANS | N Obs | Sum | N | N Miss |
|---|---|---|---|---|
| E2211U | 2724 | 1441175 | 2699 | 25 |
| A4356C | 1674 | 843446 | 1660 | 14 |
| B2348X | 1537 | 849579 | 1525 | 12 |
| K1233C | 1326 | 675275 | 1308 | 18 |
| J3455Y | 1077 | 544869 | 1065 | 12 |
| R2448Y | 1058 | 518593 | 1050 | 8 |
| L3333R | 733 | 430903 | 730 | 3 |
| D8976V | 663 | 352330 | 656 | 7 |
| M1211C | 556 | 274579 | 545 | 11 |
| M5671X | 458 | 354691 | 452 | 6 |
| K1211C | 382 | 203376 | 374 | 8 |
| | 275 | 55248 | 79 | 196 |
| XXX | 50 | 30984 | 50 | 0 |
| K1233 | 5 | . | 0 | 5 |
| K12 | 4 | . | 0 | 4 |
| R24 | 4 | 4128 | 4 | 0 |

## Step 7: Group Your Results with Formats

- SAS Formats: control the display of values of variables stored in a SAS data set
  - One of the most powerful aspects of the SAS System
  - Presentations: “My Friend the SAS Format,” and “Formats: Beyond the Basics”
    - Available for free download at www.SierraInformation.com

## *Step 7: Group Your Results with Formats*

- PROC MEANS will group, or classify the results of its work based on the formatted values of the Classification Variable IF you have associated a format to that variable
  - Two examples:
    - Applying an internal, or SAS-supplied Format
    - Creating and applying a Customized Format




## *Step 7A: Using a SAS-Supplied Format*

- Task: From a data set containing over 1.94 million records at an electrical utility,
  - find those customers who started service from 2000 onwards
    - Variable STARTDATE is a numeric SAS date value
  - compute statistics for January Revenue and KWH
  - group/present results by year

# Step 7A: Using a SAS-Supplied Format

```
108 * using SAS-supplied formats to group data;
109 * sasdata.bigdata3 has over 1.94 million records from 1958 to 2009;
110 * start date is SAS date value on which customer started service;
111 proc means data=sasdata.bigdata3(where=(year(startdate) GE 2000))
112     mean n nmiss maxdec=2;
113 var kwh1 rev1;
114 class startdate;
115 format startdate year.;
116 label kwh1 = 'January KWH'
117       rev1 = 'January Revenue'
118       startdate = 'Year Service Started';
119 title2 'Grouping Data by Year via the YEAR. Format';
120 run;
```




# Step 7A: Using a SAS-Supplied Format

```
225  proc means data=sasdata.bigdata3(where=(year(startdate) GE 2000))
226      mean n nmiss maxdec=2;
227  var kwh1 rev1;
228  class startdate;
229  format startdate year.;
230  label kwh1 = 'January KWH'
231        rev1 = 'January Revenue'
232        startdate = 'Year Service Started';
233  title2 'Grouping Data by Year via the YEAR. Format';
234  run;

NOTE: There were 69740 observations read from the data set SASDATA.BIGDATA3.
      WHERE YEAR(startdate)>=2000;
NOTE: PROCEDURE MEANS used (Total process time):
      real time           33.17 seconds
      cpu time            2.56 seconds
```

# Step 7A: Using a SAS-Supplied Format

Steps to Success with PROC MEANS
Grouping Data by Year via the YEAR. Format

The MEANS Procedure

| Year Service Started | N Obs | Variable | Label | Mean | N | N Miss |
|---|---|---|---|---|---|---|
| 2000 | 7516 | KWH1 | January KWH | 538.56 | 7445 | 71 |
| | | REV1 | January Revenue | 63.31 | 7447 | 69 |
| 2001 | 7409 | KWH1 | January KWH | 531.34 | 7336 | 73 |
| | | REV1 | January Revenue | 62.93 | 7340 | 69 |
| 2002 | 7436 | KWH1 | January KWH | 553.73 | 7362 | 74 |
| | | REV1 | January Revenue | 63.33 | 7365 | 71 |
| 2003 | 7310 | KWH1 | January KWH | 541.21 | 7251 | 59 |
| | | REV1 | January Revenue | 63.82 | 7253 | 57 |
| 2004 | 7478 | KWH1 | January KWH | 540.57 | 7417 | 61 |
| | | REV1 | January Revenue | 63.47 | 7418 | 60 |
| 2005 | 7371 | KWH1 | January KWH | 531.64 | 7321 | 50 |
| | | REV1 | January Revenue | 62.73 | 7322 | 49 |
| 2006 | 7296 | KWH1 | January KWH | 528.11 | 7228 | 68 |
| | | REV1 | January Revenue | 63.83 | 7232 | 64 |
| 2007 | 7379 | KWH1 | January KWH | 538.14 | 7314 | 65 |
| | | REV1 | January Revenue | 63.43 | 7314 | 65 |
| 2008 | 7356 | KWH1 | January KWH | 528.82 | 7287 | 69 |
| | | REV1 | January Revenue | 62.51 | 7288 | 68 |
| 2009 | 3189 | KWH1 | January KWH | 547.74 | 3165 | 24 |
| | | REV1 | January Revenue | 66.26 | 3166 | 23 |


# Step 7B: Create and Apply a Customized Format

- Task:
  - Calculate total revenue and total KWH by type of vendor supplying the transformer
- Steps:
  - Use Data Step to calculate TOTALREV and TOTALKWH
  - Determine vendor from first character of TRANS using the SUBSTR (substring) Function
  - Create customized format grouping each vendor in to one of four categories
  - Apply PROC MEANS to calculate statistics

# Step 7B: Create and Apply a Customized Format

```
122 * sum monthly revenue and monthly kwh;
123 * obtain company code from first letter of transformer (trans);
124 data elec(drop=rev1-rev12 kwh1-kwh12 trans);
125    set andrew.electricity(keep=trans region kwh1-kwh12 rev1-rev12);
126 length company $ 1;
127 label company = 'Transformer Supplier'
128       totalkwh = 'Total KWH'
129       totalrev = 'Total Rev ($)';
130 company = substr(trans,1,1);
131 totalkwh = sum(of kwh1-kwh12);
132 totalrev = sum(of rev1-rev12);
133 run;
134
135
136 proc freq data=elec;
137 title2 'Frequency of Variable Company';
138 tables company/nocum nopercent;
139 run;
```




# Frequency of Values of Variable COMPANY

```
136 proc freq data=elec;
137 title2 'Frequency of Variable Company';
138 tables company/nocum nopercent;
139 run;
```

```
Steps to Success with PROC MEANS
Frequency of Variable Company

The FREQ Procedure

Transformer Supplier

company    Frequency
-------------------------
A               1674
B               1537
C               1915
D                663
E               2724
H               1940
J               1077
K               1717
L                733
M               1014
R               1062
X                 50

Frequency Missing = 275
```

# Step 7B: Create and Apply a Customized Format

```
141 * assign format labels to first letter of transformer supplier name;
142 proc format;
143   value $vendorf
144     'I','J','M','R'  = 'Small Business (I,J,M,R)'
145     'G','D','N','H' = 'Preferred Vendor (G,D,N,H)'
146     'A','B','E','F','C','L','K'  = 'Regular Vendor (A,B,C,E,F,K,L)'
147     'X' = 'Unclassified Vendor (X)'
148     ' ' = 'Unknown Vendor';
149     run;
```




# Step 7B: Create and Apply a Customized Format

```
144
145 proc means data=elec sum maxdec=0;
146   class company;
147   format company $vendorf.;
148   var totalrev totalkwh;
149   title2 'Using Formats with Classification Variables';
150   title3 'Total Revenue and KWH Grouped by Vendor Type';
151   run;
```

# Step 7B: Create and Apply a Customized Format

Using Formats with Classification Variables
Total Revenue and KWH Grouped by Vendor Type

The MEANS Procedure

| Transformer Supplier | N Obs | Variable | Label | Sum |
|---|---|---|---|---|
| Regular Vendor (A,B,C,E,F,K,L) | 10300 | totalrev | Total Rev ($) | 7694806 |
| | | totalkwh | Total KWH | 63379729 |
| Preferred Vendor (G,D,N,H) | 2603 | totalrev | Total Rev ($) | 1976239 |
| | | totalkwh | Total KWH | 16303461 |
| Small Business (I,J,M,R) | 3153 | totalrev | Total Rev ($) | 2461891 |
| | | totalkwh | Total KWH | 20461006 |
| Unclassified Vendor (X) | 50 | totalrev | Total Rev ($) | 40618 |
| | | totalkwh | Total KWH | 378859 |



# Step 7C: Use MISSING Option to Control Output Display

```
153 * use missing option;
154 proc means data=elec sum maxdec=0 missing;
155 class company;
156 format company $vendorf.;
157 var totalrev totalkwh;
158 title2 'Using Formats with Classification Variables';
159 title3 'Total Revenue and KWH Grouped by Vendor Type';
160 title4 'With MISSING Option';
161 run;
```

# Step 7C: Use MISSING Option to Control Output Display

Using Formats with Classification Variables
Total Revenue and KWH Grouped by Vendor Type
With MISSING Option

The MEANS Procedure

| Transformer Supplier | N Obs | Variable | Label | Sum |
|---|---|---|---|---|
| Unknown Vendor | 275 | totalrev | Total Rev ($) | 176374 |
| | | totalkwh | Total KWH | 1327221 |
| Regular Vendor (A,B,C,E,F,K,L) | 10300 | totalrev | Total Rev ($) | 7694806 |
| | | totalkwh | Total KWH | 63379729 |
| Preferred Vendor (G,D,N,H) | 2603 | totalrev | Total Rev ($) | 1976239 |
| | | totalkwh | Total KWH | 16303461 |
| Small Business (I,J,M,R) | 3153 | totalrev | Total Rev ($) | 2461891 |
| | | totalkwh | Total KWH | 20461006 |
| Unclassified Vendor (X) | 50 | totalrev | Total Rev ($) | 40618 |
| | | totalkwh | Total KWH | 378859 |




# Step 7D: Use ORDER Options to Further Customize Output Display

```
164 * use missing option and order=formatted;
165 proc means data=elec sum maxdec=0 missing;
166 class company/order=formatted;
167 format company $vendorf.;
168 var totalrev totalkwh;
169 title2 'Using Formats with Classification Variables';
170 title3 'Total Revenue and KWH Grouped by Vendor Type';
171 title4 'With MISSING Option and Order=Formatted';
172 run;
173
174 * use missing option and order=freq;
175 proc means data=elec sum maxdec=0 missing;
176 class company/order=freq;
177 format company $vendorf.;
178 var totalrev totalkwh;
179 title2 'Using Formats with Classification Variables';
180 title3 'Total Revenue and KWH Grouped by Vendor Type';
181 title4 'With MISSING Option and Order=Freq';
182 run;
```

## Step 7D: Use ORDER Options to Further Customize Output Display

Steps to Success with PROC MEANS
Using Formats with Classification Variables
Total Revenue and KWH Grouped by Vendor Type
With MISSING Option and Order=Formatted

The MEANS Procedure

| Transformer Supplier | N Obs | Variable | Label | Sum |
|---|---|---|---|---|
| Preferred Vendor (G,D,N,H) | 2603 | totalrev | Total Rev ($) | 1976239 |
| | | totalkwh | Total KWH | 16303461 |
| Regular Vendor (A,B,C,E,F,K,L) | 10300 | totalrev | Total Rev ($) | 7694806 |
| | | totalkwh | Total KWH | 63379729 |
| Small Business (I,J,M,R) | 3153 | totalrev | Total Rev ($) | 2461891 |
| | | totalkwh | Total KWH | 20461006 |
| Unclassified Vendor (X) | 50 | totalrev | Total Rev ($) | 40618 |
| | | totalkwh | Total KWH | 378859 |
| Unknown Vendor | 275 | totalrev | Total Rev ($) | 176374 |
| | | totalkwh | Total KWH | 1327221 |



## Step 7D: Use ORDER Options to Further Customize Output Display

Steps to Success with PROC MEANS
Using Formats with Classification Variables
Total Revenue and KWH Grouped by Vendor Type
With MISSING Option and Order=Freq

The MEANS Procedure

| Transformer Supplier | N Obs | Variable | Label | Sum |
|---|---|---|---|---|
| Regular Vendor (A,B,C,E,F,K,L) | 10300 | totalrev | Total Rev ($) | 7694806 |
| | | totalkwh | Total KWH | 63379729 |
| Small Business (I,J,M,R) | 3153 | totalrev | Total Rev ($) | 2461891 |
| | | totalkwh | Total KWH | 20461006 |
| Preferred Vendor (G,D,N,H) | 2603 | totalrev | Total Rev ($) | 1976239 |
| | | totalkwh | Total KWH | 16303461 |
| Unknown Vendor | 275 | totalrev | Total Rev ($) | 176374 |
| | | totalkwh | Total KWH | 1327221 |
| Unclassified Vendor (X) | 50 | totalrev | Total Rev ($) | 40618 |
| | | totalkwh | Total KWH | 378859 |

## *Step 8: Save PROC MEANS' Output in a SAS Data Set*

- By default, results generated by PROC MEANS are displayed in your Output Window
- You can also save the output in either a permanent or temporary SAS data set by adding commands in an OUTPUT statement.




## *Step 8: The Default Output SAS Data Set Created by PROC MEANS*

```
200
201  * create default output data set, use NOPRINT option;
202  proc means NOPRINT data=elec;
203  class region;
204  var totalkwh totalrev;
205  output out=new1;
206  run;
207
208  proc print data=new1;
209  title2 'Default Data Set Created by PROC MEANS';
210  run;
```

# Step 8: The Default Output SAS Data Set Created by PROC MEANS

Steps to Success with PROC MEANS
Default Data Set Created by PROC MEANS

| Obs | REGION | _TYPE_ | _FREQ_ | _STAT_ | totalkwh | totalrev |
|---|---|---|---|---|---|---|
| 1 | | 0 | 16381 | N | 16328.00 | 16381.00 |
| 2 | | 0 | 16381 | MIN | 0.00 | 1.65 |
| 3 | | 0 | 16381 | MAX | 361920.00 | 40665.50 |
| 4 | | 0 | 16381 | MEAN | 6237.77 | 753.92 |
| 5 | | 0 | 16381 | STD | 8963.94 | 1046.70 |
| 6 | EASTERN | 1 | 5124 | N | 5109.00 | 5124.00 |
| 7 | EASTERN | 1 | 5124 | MIN | 0.00 | 6.98 |
| 8 | EASTERN | 1 | 5124 | MAX | 361920.00 | 40665.50 |
| 9 | EASTERN | 1 | 5124 | MEAN | 6030.37 | 727.94 |
| 10 | EASTERN | 1 | 5124 | STD | 12829.55 | 1443.28 |
| 11 | NORTHERN | 1 | 5462 | N | 5448.00 | 5462.00 |
| 12 | NORTHERN | 1 | 5462 | MIN | 0.00 | 13.00 |
| 13 | NORTHERN | 1 | 5462 | MAX | 114774.00 | 14565.83 |
| 14 | NORTHERN | 1 | 5462 | MEAN | 6951.14 | 834.46 |
| 15 | NORTHERN | 1 | 5462 | STD | 5830.40 | 710.39 |
| 16 | SOUTHERN | 1 | 720 | N | 720.00 | 720.00 |
| 17 | SOUTHERN | 1 | 720 | MIN | 76.00 | 60.00 |
| 18 | SOUTHERN | 1 | 720 | MAX | 22771.00 | 2917.74 |
| 19 | SOUTHERN | 1 | 720 | MEAN | 6787.57 | 806.24 |
| 20 | SOUTHERN | 1 | 720 | STD | 4108.75 | 507.99 |
| 21 | WESTERN | 1 | 5075 | N | 5051.00 | 5075.00 |
| 22 | WESTERN | 1 | 5075 | MIN | 0.00 | 1.65 |
| 23 | WESTERN | 1 | 5075 | MAX | 259360.00 | 32667.08 |
| 24 | WESTERN | 1 | 5075 | MEAN | 5599.73 | 686.04 |
| 25 | WESTERN | 1 | 5075 | STD | 7292.77 | 917.18 |



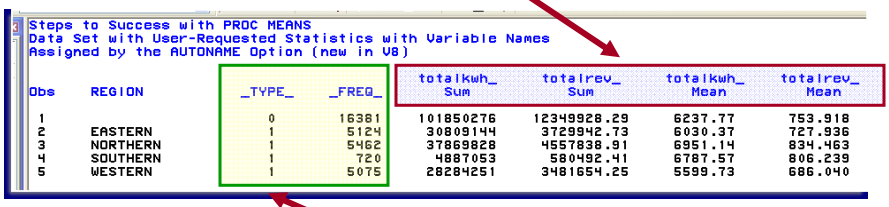
# Step 9: Control Creation of the Output SAS Data Set

```
212 proc means data=elec noprint;
213  class region;
214  var totalkwh totalrev;
215  * autoname option automatically assigns names to
216    variables in output data set;
217  output out=new2 sum= mean= / autoname;
218  run;
219
220 proc print data=new2;
221  title2 'Data Set with User-Requested Statistics with Variable Names';
222  title3 'Assigned by the AUTONAME Option (new in V8)';
223  run;
```

The AUTONAME Option automatically assigns unique variable names in the Output Data Set "holding" the statistics requested in the OUTPUT statement. The variable name is formed by adding an underscore to the last character of the name of the Analysis Variable and appending to that the name of the Statistics Keyword.

# Step 9: Control Creation of the Output SAS Data Set

**Variables created (and automatically named) by PROC MEANS**



Steps to Success with PROC MEANS
Data Set with User-Requested Statistics with Variable Names
Assigned by the AUTONAME Option (new in V8)

| Obs | REGION | _TYPE_ | _FREQ_ | totalkwh_ Sum | totalrev_ Sum | totalkwh_ Mean | totalrev_ Mean |
|---|---|---|---|---|---|---|---|
| 1 | | 0 | 16381 | 101850276 | 12349928.29 | 6237.77 | 753.918 |
| 2 | EASTERN | 1 | 5124 | 30809144 | 3729942.73 | 6030.37 | 727.936 |
| 3 | NORTHERN | 1 | 5462 | 37869828 | 4557838.91 | 6951.14 | 834.463 |
| 4 | SOUTHERN | 1 | 720 | 4887053 | 580492.41 | 6787.57 | 806.239 |
| 5 | WESTERN | 1 | 5075 | 28284251 | 3481654.25 | 5599.73 | 686.040 |

**Variables automatically created by PROC MEANS when an OUTPUT Statement. _TYPE_ shows how the values of the Classification Variables were combined from the Input Data Set to create the row/observation in the Output Data Set. _FREQ_ shows the number of observations from the Input Data Set whose values were used to calculate statistics on that row /observation in the Output Data Set.**



# Step 9: Control Creation of the Output SAS Data Set

- By default, observations in the Output Data Set are ordered by ascending (lowest to highest) value of _TYPE_
  - And, within each value of _TYPE_, by ascending value of the classification variable(s) specified in the CLASS Statement
- DESCENDTYPES Option
  - Orders rows/observations in the Output Data Set by descending value of _TYPE_

# Step 9A: Using the DESCENDTYPES Option

Specifying the DESCENDTYPES Option

```
225 proc means data=elec noprint DESCENDTYPES;
226 class region;
227 var totalkwh totalrev;
228 * autoname option automatically assigns names to
229   variables in output data set;
230 output out=new2B sum= mean= / autoname;
231 run;
232
233 proc print data=new2B;
234 title2 'Data Set with User-Requested Statistics with Variable Names';
235 title3 'Assigned by the AUTONAME Option (new in V8)';
236 title4 'Applying the DESCENDTYPES Option';
237 run;
```




# Step 9A: Using the DESCENDTYPES Option

Steps to Success with PROC MEANS
Data Set with User-Requested Statistics with Variable Names
Assigned by the AUTONAME Option (new in V8)
Applying the DESCENDTYPES Option

| Obs | REGION | _TYPE_ | _FREQ_ | totalkwh_ Sum | totalrev_ Sum | totalkwh_ Mean | totalrev_ Mean |
|---|---|---|---|---|---|---|---|
| 1 | EASTERN | 1 | 5124 | 30809144 | 3729942.73 | 6030.37 | 727.936 |
| 2 | NORTHERN | 1 | 5462 | 37869828 | 4557838.91 | 6951.14 | 834.463 |
| 3 | SOUTHERN | 1 | 720 | 4887053 | 580492.41 | 6787.57 | 806.239 |
| 4 | WESTERN | 1 | 5075 | 28284251 | 3481654.25 | 5599.73 | 686.040 |
| 5 | | 0 | 16381 | 101850276 | 12349928.29 | 6237.77 | 753.918 |

## Step 10: Understand the _TYPE_ Variable

- Automatically added to data sets created by an OUTPUT Statement in PROC MEANS
  - Is a numeric variable by default
  - Number of values in the Output Data Set:
    - $2^N$, where N = number of Classification Variables
      - Values range from zero (0) to $2^N$ - 1




## Step 10: Understand the _TYPE_ Variable

- First, a new data set to analyze!

```
240 *  understand _type_;
241 * make a new data set;
242 data elec_new(drop=trans kwh1-kwh12 rev1-rev12);
243 set andrew.electricity(where=(region in('EASTERN','WESTERN') and
244                               cesched =: 'E1' and
245                               substr(trans,1,1) in('A','B','C'))
246                keep=trans cesched region kwh1-kwh12 rev1-rev12);
247 length company $ 1;
248 label company = 'Transformer Supplier'
249       totalkwh = 'Total KWH'
250       totalrev = 'Total Rev ($)'
251       cesched = 'Electric Rate Schedule'
252       region = 'Region Serving Customer';
253 company = substr(trans,1,1);
254 totalkwh = sum(of kwh1-kwh12);
255 totalrev = sum(of rev1-rev12);
256 run;
```

# Step 10: Understand the _TYPE_ Variable

```
258 proc means noprint data=elec_new;
259  class region company; * < two classification variables;
260  var totalkwh;
261  output out=new3 min= max= mean=/autoname;
262  run;
263
264 proc print data=new3;
265  title2 'Understanding _TYPE_: Two Class Variables';
266  run;
```




# Step 10: Understand the _TYPE_ Variable

Steps to Success with PROC MEANS
Understanding _TYPE_: Two Class Variables

| Obs | REGION | company | _TYPE_ | _FREQ_ | totalkwh_ Min | totalkwh_ Max | totalkwh_ Mean |
|---|---|---|---|---|---|---|---|
| 1 | | | 0 | 3175 | 0 | 144240 | 5327.63 |
| 2 | | A | 1 | 974 | 0 | 23339 | 5245.48 |
| 3 | | B | 1 | 943 | 0 | 144240 | 5876.83 |
| 4 | | C | 1 | 1258 | 1 | 14958 | 4979.56 |
| 5 | EASTERN | | 2 | 1551 | 0 | 144240 | 5424.41 |
| 6 | WESTERN | | 2 | 1624 | 24 | 22739 | 5235.20 |
| 7 | EASTERN | A | 3 | 481 | 0 | 23339 | 4968.57 |
| 8 | EASTERN | B | 3 | 433 | 0 | 144240 | 6662.15 |
| 9 | EASTERN | C | 3 | 637 | 1 | 14958 | 4927.26 |
| 10 | WESTERN | A | 3 | 493 | 24 | 18687 | 5515.64 |
| 11 | WESTERN | B | 3 | 510 | 30 | 22739 | 5210.08 |
| 12 | WESTERN | C | 3 | 621 | 794 | 14068 | 5033.20 |

# Step 10: Understand the _TYPE_ Variable

```
268 proc means data=elec_new noprint;
269 class region company cesched; * three classification variables;
270 var totalkwh;
271 output out=new4 min= max= mean=/autoname;
272 run;
273
274 proc print data=new4;
275 title2 'Understanding _TYPE_: Three Class Variables';
276 run;
```




# Step 10: Understand the _TYPE_ Variable

Steps to Success with PROC MEANS
Understanding _TYPE_: Three Class Variables

| Obs | REGION | company | CESCHED | _TYPE_ | _FREQ_ | totalkwh_ Min | totalkwh_ Max | totalkwh_ Mean |
|---|---|---|---|---|---|---|---|---|
| 1 | | | | 0 | 3175 | 0 | 144240 | 5327.63 |
| 2 | | | E1 | 1 | 2937 | 0 | 23339 | 5148.06 |
| 3 | | | E1L | 1 | 197 | 1073 | 10430 | 4637.03 |
| 4 | | | E1M | 1 | 40 | 2124 | 122400 | 18441.28 |
| 5 | | | E1T | 1 | 1 | 144240 | 144240 | 144240.00 |
| 6 | | A | | 2 | 974 | 0 | 23339 | 5245.48 |
| 7 | | B | | 2 | 943 | 0 | 144240 | 5876.83 |
| 8 | | C | | 2 | 258 | 1 | 14958 | 4979.56 |
| 9 | | A | E1 | 3 | 888 | 0 | 23339 | 5386.13 |
| 10 | | A | E1L | 3 | 58 | 1128 | 6072 | 3220.45 |
| 11 | | A | E1M | 3 | 28 | 2124 | 8600 | 4979.46 |
| 12 | | B | E1 | 3 | 900 | 0 | 16423 | 5180.54 |
| 13 | | B | E1L | 3 | 34 | 1910 | 6966 | 4546.18 |
| 14 | | B | E1M | 3 | 8 | 22739 | 122400 | 72569.50 |
| 15 | | B | E1T | 3 | 1 | 144240 | 144240 | 144240.00 |
| 16 | | C | E1 | 3 | 149 | 1 | 14958 | 4938.62 |
| 17 | | C | E1L | 3 | 105 | 1073 | 10430 | 5448.94 |
| 18 | | C | E1M | 3 | 4 | 3527 | 7089 | 4417.50 |
| 19 | EASTERN | | | 4 | 551 | 0 | 144240 | 5424.41 |
| 20 | WESTERN | | | 4 | 624 | 24 | 22739 | 5235.20 |

# Step 10: Understand the _TYPE_ Variable

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 21 | EASTERN | | E1 | 5 | 1440 | 0 | 23339 | 5041.19 |
| 22 | EASTERN | | E1L | 5 | 91 | 1128 | 9661 | 4979.32 |
| 23 | EASTERN | | E1M | 5 | 19 | 2124 | 122400 | 29294.16 |
| 24 | EASTERN | | E1T | 5 | 1 | 144240 | 144240 | 144240.00 |
| 25 | WESTERN | | E1 | 5 | 1497 | 24 | 18687 | 5250.86 |
| 26 | WESTERN | | E1L | 5 | 106 | 1073 | 10430 | 4343.18 |
| 27 | WESTERN | | E1M | 5 | 21 | 3130 | 22739 | 8622.00 |
| 28 | EASTERN | A | | 6 | 481 | 0 | 23339 | 4968.57 |
| 29 | EASTERN | B | | 6 | 433 | 0 | 144240 | 6662.15 |
| 30 | EASTERN | C | | 6 | 637 | 1 | 14958 | 4927.26 |
| 31 | WESTERN | A | | 6 | 493 | 24 | 18687 | 5515.64 |
| 32 | WESTERN | B | | 6 | 510 | 30 | 22739 | 5210.08 |
| 33 | WESTERN | C | | 6 | 621 | 794 | 14068 | 5033.20 |
| 34 | EASTERN | A | E1 | 7 | 441 | 0 | 23339 | 5098.02 |
| 35 | EASTERN | A | E1L | 7 | 26 | 1128 | 5350 | 3144.42 |
| 36 | EASTERN | A | E1M | 7 | 14 | 2124 | 5510 | 4278.57 |




# Step 10: Understand the _TYPE_ Variable

Steps to Success with PROC MEANS
Understanding _TYPE_: Three Class Variables

| Obs | REGION | company | CESCHED | _TYPE_ | _FREQ_ | totalkwh_ Min | totalkwh_ Max | totalkwh_ Mean |
|---|---|---|---|---|---|---|---|---|
| 37 | EASTERN | B | E1 | 7 | 417 | 0 | 16423 | 5302.56 |
| 38 | EASTERN | B | E1L | 7 | 11 | 1910 | 5295 | 3609.18 |
| 39 | EASTERN | B | E1M | 7 | 4 | 122400 | 122400 | 122400.00 |
| 40 | EASTERN | B | E1T | 7 | 1 | 144240 | 144240 | 144240.00 |
| 41 | EASTERN | C | E1 | 7 | 582 | 1 | 14958 | 4810.85 |
| 42 | EASTERN | C | E1L | 7 | 54 | 3295 | 9661 | 6141.89 |
| 43 | EASTERN | C | E1M | 7 | 1 | 7089 | 7089 | 7089.00 |
| 44 | WESTERN | A | E1 | 7 | 447 | 24 | 18687 | 5670.37 |
| 45 | WESTERN | A | E1L | 7 | 32 | 1799 | 6072 | 3282.22 |
| 46 | WESTERN | A | E1M | 7 | 14 | 3130 | 8600 | 5680.36 |
| 47 | WESTERN | B | E1 | 7 | 483 | 30 | 12969 | 5075.19 |
| 48 | WESTERN | B | E1L | 7 | 23 | 2800 | 6966 | 4994.30 |
| 49 | WESTERN | B | E1M | 7 | 4 | 22739 | 22739 | 22739.00 |
| 50 | WESTERN | C | E1 | 7 | 567 | 794 | 14068 | 5069.77 |
| 51 | WESTERN | C | E1L | 7 | 51 | 1073 | 10430 | 4715.24 |
| 52 | WESTERN | C | E1M | 7 | 3 | 3527 | 3527 | 3527.00 |

# Step 10B: Understand the CHARTYPE Option

- The CHARTYPE Option converts the default numeric values of _TYPE_ to a character string composed of zeros and ones
  - The order of the zeros and ones corresponds to the arrangement, from left to right, of the CLASS Statement variables
  - Using the CHARTYPE Option drastically simplifies the creation of multiple output SAS data sets in a single use of PROC MEANS, the final “step for success” in this presentation.




# Step 10B: Understand the CHARTYPE Option

```
278 * understand CHARTYPE;
279 proc means noprint data=elec_new CHARTYPE;
280 class region company; * < two classification variables;
281 var totalkwh;
282 output out=new5 min= max= mean=/autoname;
283 run;
284
285 proc print data=new5;
286 title2 'Understanding _TYPE_: Two Class Variables';
287 title3 'The CHARTYPE Option';
288 run;
```

# Step 10B: Understand the CHARTYPE Option

Steps to Success with PROC MEANS
Understanding _TYPE_: Two Class Variables
The CHARTYPE Option

| Obs | REGION | company | _TYPE_ | _FREQ_ | totalkwh_ Min | totalkwh_ Max | totalkwh_ Mean |
|---|---|---|---|---|---|---|---|
| 1 | | | 00 | 3175 | 0 | 144240 | 5327.63 |
| 2 | | A | 01 | 974 | 0 | 23339 | 5245.48 |
| 3 | | B | 01 | 943 | 0 | 144240 | 5876.83 |
| 4 | | C | 01 | 1258 | 1 | 14958 | 4979.56 |
| 5 | EASTERN | | 10 | 1551 | 0 | 144240 | 5424.41 |
| 6 | WESTERN | | 10 | 1624 | 24 | 22739 | 5235.20 |
| 7 | EASTERN | A | 11 | 481 | 0 | 23339 | 4968.57 |
| 8 | EASTERN | B | 11 | 433 | 0 | 144240 | 6662.15 |
| 9 | EASTERN | C | 11 | 637 | 1 | 14958 | 4927.26 |
| 10 | WESTERN | A | 11 | 493 | 24 | 18687 | 5515.64 |
| 11 | WESTERN | B | 11 | 510 | 30 | 22739 | 5210.08 |
| 12 | WESTERN | C | 11 | 621 | 794 | 14068 | 5033.20 |




# Step 11: Create Multiple Output Data Sets in a Single PROC MEANS Step

- One of the most powerful, and often overlooked, capabilities of PROC MEANS
  - Avoids unnecessary multiple “interrogations” of the Input Data Set to calculate statistics at various combinations of the Classification Variables
    - CHARTYPE Option simplifies this process greatly
    - Unlimited number of OUTPUT Statements allowed in a single PROC MEANS step, each of which creates a separate SAS data set according to the instructions specified in that Statement.

# Step 11: Create Multiple Output Data Sets in a Single PROC MEANS Step

```
289
290 proc means noprint data=elec_new CHARTYPE DESCENDTYPES;
291 class region company;
292 var totalkwh;
293 output out=new6(where=(_type_ in('00','11'))) sum= /autoname;
294 output out=new7(where=(_type_ in('10','00'))) sum= mean=/autoname;
295 output out=new8(where=(_type_ in('01','00'))) sum = mean= max= min=/autoname;
296 run;
```




# Step 11: Create Multiple Output Data Sets in a Single PROC MEANS Step

```
298 proc print data=new6;
299 title2 'Creating Multiple SAS Data Sets in a Single PROC MEANS Step';
300 title3 'Data Set New6: Analysis Grouped by Region and Company';
301 title4 'With Overall Results at the Bottom of the Output Data Set';
302 run;
303
304 proc print data=new7;
305 title2 'Creating Multiple SAS Data Sets in a Single PROC MEANS Step';
306 title3 'Data Set New7: Analysis Grouped by Region';
307 title4 'With Overall Results at the Bottom of the Output Data Set';
308 run;
309
310 proc print data=new8;
311 title2 'Creating Multiple SAS Data Sets in a Single PROC MEANS Step';
312 title3 'Data Set New8: Analysis Grouped by Company';
313 title4 'With Overall Results at the Bottom of the Output Data Set';
314 run;
```

# Step 11: Create Multiple Output Data Sets in a Single PROC MEANS Step

Steps to Success with PROC MEANS
Creating Multiple SAS Data Sets in a Single PROC MEANS Step
Data Set New6: Analysis Grouped by Region and Company
With Overall Results at the Bottom of the Output Data Set

| Obs | REGION | company | _TYPE_ | _FREQ_ | totalkwh_ Sum |
|---|---|---|---|---|---|
| 1 | EASTERN | A | 11 | 481 | 2389881 |
| 2 | EASTERN | B | 11 | 433 | 2884710 |
| 3 | EASTERN | C | 11 | 637 | 3138663 |
| 4 | WESTERN | A | 11 | 493 | 2719212 |
| 5 | WESTERN | B | 11 | 510 | 2657140 |
| 6 | WESTERN | C | 11 | 621 | 3125620 |
| 7 | | | 00 | 3175 | 16915226 |




# Step 11: Create Multiple Output Data Sets in a Single PROC MEANS Step

Steps to Success with PROC MEANS
Creating Multiple SAS Data Sets in a Single PROC MEANS Step
Data Set New7: Analysis Grouped by Region
With Overall Results at the Bottom of the Output Data Set

| Obs | REGION | company | _TYPE_ | _FREQ_ | totalkwh_ Sum | totalkwh_ Mean |
|---|---|---|---|---|---|---|
| 1 | EASTERN | | 10 | 1551 | 8413254 | 5424.41 |
| 2 | WESTERN | | 10 | 1624 | 8501972 | 5235.20 |
| 3 | | | 00 | 3175 | 16915226 | 5327.63 |

## Step 11: Create Multiple Output Data Sets in a Single PROC MEANS Step

Steps to Success with PROC MEANS
Creating Multiple SAS Data Sets in a Single PROC MEANS Step
Data Set New8: Analysis Grouped by Company
With Overall Results at the Bottom of the Output Data Set

| Obs | REGION | company | _TYPE_ | _FREQ_ | totalkwh_ Sum | totalkwh_ Mean | totalkwh_ Max | totalkwh_ Min |
|---|---|---|---|---|---|---|---|---|
| 1 | | A | 01 | 974 | 5109093 | 5245.48 | 23339 | 0 |
| 2 | | B | 01 | 943 | 5541850 | 5876.83 | 144240 | 0 |
| 3 | | C | 01 | 1258 | 6264283 | 4979.56 | 14958 | 1 |
| 4 | | | 00 | 3175 | 16915226 | 5327.63 | 144240 | 0 |


*Sept 2010*

# PROC MEANS: Beyond the Basics

**Andrew H. Karp**
Sierra Information Services
19229 Sonoma Hwy #264
Sonoma, CA 95476
707 996 7380
andrew@sierrainformation.com
www.SierraInformation.com


*Sept 2010*

80

## Step 12: Use the PRELOADFMT and COMPLETETYPES Options in PROC MEANS

- By default, PROC MEANS will only display formatted values of classification variables with non-missing values of the specified analysis variables
- But, in many analytic situations we want to show all formatted values of the classification variable, including those which have no observations in the input data set




## Step 12: Use the PRELOADFMT and COMPLETETYPES Options in PROC MEANS

```
5  proc format;
6     value color_new_fmt
7        1 = 'White'
8        2,3,4,5 = 'Natural'
9        6,7 = 'Black'
10       9 = 'Pink'
11       10 = 'Orange'
12       11 = 'Purple';
13       run;
14
15 proc means data=sasclass.tables_sales_final sum mean maxdec=2;
16 class color;
17 var price;
18 format color color_new_fmt.;
19 title1 'Intro to Data Summarization and Reporting w/PROCs FREQ and MEANS';
20 title2 'Without PRELOADFMT and COMPLETETYPES';
21 run;
```

# Step 12: Use the PRELOADFMT and COMPLETETYPES Options in PROC ME

```
Intro to Data Summarization and Reporting w/PROCs FREQ and MEANS
Without PRELOADFMT and COMPLETETYPES

The MEANS Procedure

            Analysis Variable : price

color       N Obs             Sum             Mean
------------------------------------------------------
White        6185       3211399.00           519.22

Natural     24247      12442922.00           513.17

Black       12411       6304752.00           508.00
------------------------------------------------------
```

*By default, PROC MEANS only shows formatted values of COLOR that exist in the data set whose observations it analyzed. Since there were no observations in the data set with values of 9 (PINK) , 10 (ORANGE) or 11 (PURPLE), there are no rows for those values displayed in the PROC MEANS-generated output.*




# Step 12: Use the PRELOADFMT and COMPLETETYPES Options in PROC MEANS

```
23 proc means data=sasclass.tables_sales_final sum mean maxdec=2 completetypes;
24 class color/preloadfmt;
25 var price;
26 format color color_new_fmt.;
27 title1 'Intro to Data Summarization and Reporting w/PROCs FREQ and MEANS';
28 title2 '*With* PRELOADFMT and COMPLETETYPES';
29 run;
```

## Step 12: Use the PRELOADFMT and COMPLETETYPES Options in PROC MEANS

Intro to Data Summarization and Reporting w/PROCs FREQ and MEANS
*With* PRELOADFMT and COMPLETETYPES

The MEANS Procedure

Analysis Variable : price

| color | N Obs | Sum | Mean |
|---|---|---|---|
| White | 6185 | 3211399.00 | 519.22 |
| Natural | 24247 | 12442922.00 | 513.17 |
| Black | 12411 | 6304752.00 | 508.00 |
| Pink | 0 | . | . |
| Orange | 0 | . | . |
| Purple | 0 | . | . |




## Step 13: Use Multilabel Formats

- Default: One value of a variable can be assigned to one Format label
  - Overlapping value ranges not allowed
- Multilabel Formats permit assignment of the same variable value to multiple format labels
  - Added to SAS Version 8, updated/enhanced in SAS 9
  - Great for “subtotals” and other groupings
  - Can be used **ONLY** in PROCs MEANS, SUMMARY and TABULATE

# Step 13: Use Multilabel Formats

```
proc format;
  value $region_mlf_fmt (multilabel)
'01' = 'District 01'
'02' = 'District 02' /* Primary Format Label */
'03' = 'District 03' '04' = 'District 04' '05' = 'District 05'
'06' = 'District 06' '07' = 'District 07' '08' = 'District 08'
'09' = 'District 09'
'01' - '09' = 'Region A'  /* Secondary Format Label */
'10' = 'District 10' '11' = 'District 11'   '12' = 'District 12' '13' = 'District 13'
'14' = 'District 14' '15' = 'District 15'  '16' = 'District 16' '17' = 'District 17'
'18' = 'District 18' '19' = 'District 19'  '20' = 'District 20'
'10' - '20' = 'Region B'
'21' = 'District 21' '22' = 'District 22' '23' = 'District 23' '24' = 'District 24'
'25' = 'District 25' '26' = 'District 26' '27' = 'District 27' '28' = 'District 28'
'29' = 'District 29' '30' = 'District 30' '21' - '30' = 'Region C'
'31' = 'District 31'    '32' = 'District 32' '33' = 'District 33'   '34' = 'District 34'
'35' = 'District 35'    '36' = 'District 36' '37' = 'District 37'    '40' = 'District 40'
'41' = 'District 41'    '42' = 'District 42' '43' = 'District 43'    '44' = 'District 44'
'45' = 'District 45'    '46' = 'District 46' '47' = 'District 47'    '48' = 'District 48' '49' = 'District 49'
'31' - '49' = 'Region D'
low - high = 'Statewide';
run;
```



# Step 13: Use Multilabel Formats

```
proc means data=sasclass.payments_sample sum mean n maxdec=0;
class legislative/mlf preloadfmt; /* BOTH options needed */
var total_dollars;
format legislative $region_mlf_fmt.;
title1 'Summarizing and Reporting Data w/PROCs FREQ and MEANS';
title2 'Using MULTILABEL Formats';
run;
```

*The MLF Option instructs PROC MEANS to use both the PRIMARY and SECONDARY Format labels when grouping/analyzing the data.*

*PRELOADFMT instructs PROC MEANS to 'preload' the Format in to memory so that all groupings/classifications formed by it are 'available' to the PROC as it reads data from the input data set and "buckets" observations in to classes (or groups) for analysis. By default, the Format labels are applied AFTER the grouping/classification of the input data set has taken place.*

# Step 13: Use Multilabel Formats

Summarizing and Reporting Data w/PROCs FREQ and MEANS
Using MULTILABEL Formats

The MEANS Procedure

Analysis Variable : total_dollars

| legislative | N Obs | Sum | Mean | N |
|---|---|---|---|---|
| District 01 | 1054 | 3698207 | 3509 | 1054 |
| District 02 | 1066 | 3775850 | 3542 | 1066 |
| District 03 | 1049 | 4905324 | 4676 | 1049 |
| District 04 | 1059 | 4634930 | 4377 | 1059 |
| District 05 | 1072 | 4453337 | 4154 | 1072 |
| District 06 | 1053 | 5629480 | 5346 | 1053 |
| District 07 | 1060 | 4919414 | 4641 | 1060 |
| District 08 | 1042 | 3226456 | 3096 | 1042 |




# Step 13: Use Multilabel Formats

| legislative | N Obs | Sum | Mean | N |
|---|---|---|---|---|
| District 49 | 1067 | 4351421 | 4078 | 1067 |
| Region A | 9501 | 38556042 | 4058 | 9501 |
| Region B | 11667 | 40850692 | 3501 | 11667 |
| Region C | 10595 | 48371810 | 4566 | 10595 |
| Region D | 20184 | 92742845 | 4595 | 20184 |
| Statewide | 51947 | 220521389 | 4245 | 51947 |

## Step 13: Use Multilabel Formats

- Although the results on the previous two slides are correct, the ordering of the PROC MEANS-generated output is not quite what we wanted
  - By default, PROC FORMAT sorts the labels in to “sort order” when creating the Format
  - What we need is a way to get PROC FORMAT to “respect,” or “honor” the ordering of the labels as coded in the Value Statement
    - NOTSORTED Option




## Step 13: Use Multilabel Formats

```
40  proc format;
41    value $region_mlf_fmt (multilabel notsorted)   /* add the NOTSORTED option to preserve label order */
42  '01' = 'District 01'
43  '02' = 'District 02' /* Primary Format Label */
44  '03' = 'District 03' '04' = 'District 04' '05' = 'District 05'
45  '06' = 'District 06' '07' = 'District 07' '08' = 'District 08'
46  '09' = 'District 09'
47  '01' - '09' = 'Region A'  /* Secondary Format Label */
48  '10' = 'District 10' '11' = 'District 11'   '12' = 'District 12' '13' = 'District 13'
49  '14' = 'District 14' '15' = 'District 15'  '16' = 'District 16' '17' = 'District 17'
50  '18' = 'District 18' '19' = 'District 19'  '20' = 'District 20'
51  '10' - '20' = 'Region B'
52  '21' = 'District 21' '22' = 'District 22' '23' = 'District 23' '24' = 'District 24'
53  '25' = 'District 25' '26' = 'District 26' '27' = 'District 27' '28' = 'District 28'
54  '29' = 'District 29' '30' = 'District 30' '21' - '30' = 'Region C'
55  '31' = 'District 31'    '32' = 'District 32' '33' = 'District 33'   '34' = 'District 34'
56  '35' = 'District 35'    '36' = 'District 36' '37' = 'District 37'    '40' = 'District 40'
57  '41' = 'District 41'    '42' = 'District 42' '43' = 'District 43'    '44' = 'District 44'
58  '45' = 'District 45'    '46' = 'District 46' '47' = 'District 47'    '48' = 'District 48' '49' = 'District 49'
59  '31' - '49' = 'Region D'
60  low - high = 'Statewide';
61  run;
```

# Step 13: Use Multilabel Formats

```
63 proc means data=sasclass.payments_sample sum mean n maxdec=0;
64 class legislative/mlf preloadfmt order=data; /* ALL options needed */
65 var total_dollars;
66 format legislative $region_mlf_fmt.;
67 title1 'Summarizing and Reporting Data w/PROCs FREQ and MEANS';
68 title2 'Using MULTILABEL Formats with ORDER=DATA Option';
69 run;
```

***The ORDER=DATA option instructs PROC MEANS to display the output ordered by the***




# ORDER=DATA in PROC MEANS

ORDER=DATA | FORMATTED | FREQ | UNFORMATTED

specifies the order to group the levels of the class variables in the output, where

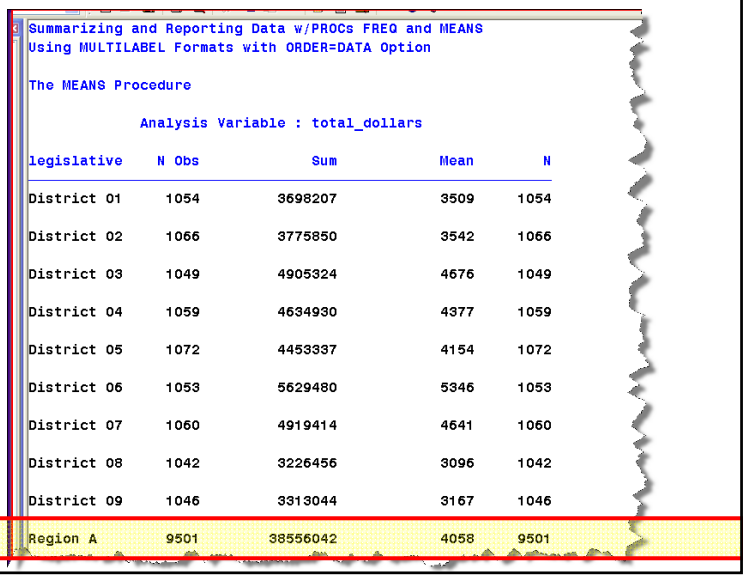
DATA

orders values according to their order in the input data set.

| | |
|---|---|
| **Interaction:** | If you use PRELOADFMT, then the order of the values of each class variable matches the order that PROC FORMAT uses to store the values of the associated user-defined format. If you use the CLASSDATA= option in the PROC statement, then PROC MEANS uses the order of the unique values of each class variable in the CLASSDATA= data set to order the output levels. If you use both options, then PROC MEANS first uses the user-defined formats to order the output. If you omit EXCLUSIVE in the PROC statement, then PROC MEANS appends after the user-defined format and the CLASSDATA= values the unique values of the class variables in the input data set based on the order in which they are encountered. |
| **Tip:** | By default, PROC FORMAT stores a format definition in sorted order. Use the NOTSORTED option to store the values or ranges of a user defined format in the order that you define them. |
| **Featured in:** | Computing Output Statistics with Missing Class Variable Values |

**Excerpt from PROC MEANS Documentation**

# Step 13: Use Multilabel Formats



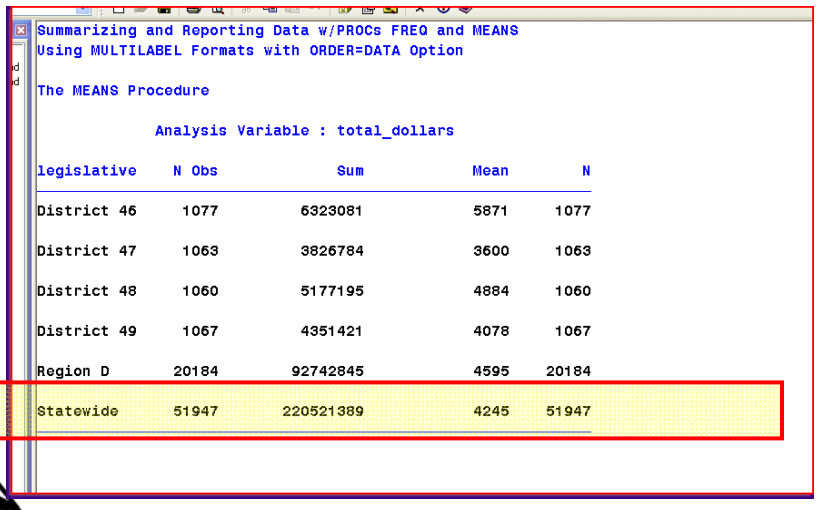
Summarizing and Reporting Data w/PROCs FREQ and MEANS
Using MULTILABEL Formats with ORDER=DATA Option

The MEANS Procedure

Analysis Variable : total_dollars

| legislative | N Obs | Sum | Mean | N |
|---|---|---|---|---|
| District 01 | 1054 | 3698207 | 3509 | 1054 |
| District 02 | 1066 | 3775850 | 3542 | 1066 |
| District 03 | 1049 | 4905324 | 4676 | 1049 |
| District 04 | 1059 | 4634930 | 4377 | 1059 |
| District 05 | 1072 | 4453337 | 4154 | 1072 |
| District 06 | 1053 | 5629480 | 5346 | 1053 |
| District 07 | 1060 | 4919414 | 4641 | 1060 |
| District 08 | 1042 | 3226456 | 3096 | 1042 |
| District 09 | 1046 | 3313044 | 3167 | 1046 |
| Region A | 9501 | 38556042 | 4058 | 9501 |

# Step 13: Use Multilabel Formats



Summarizing and Reporting Data w/PROCs FREQ and MEANS
Using MULTILABEL Formats with ORDER=DATA Option

The MEANS Procedure

Analysis Variable : total_dollars

| legislative | N Obs | Sum | Mean | N |
|---|---|---|---|---|
| District 46 | 1077 | 6323081 | 5871 | 1077 |
| District 47 | 1063 | 3826784 | 3600 | 1063 |
| District 48 | 1060 | 5177195 | 4884 | 1060 |
| District 49 | 1067 | 4351421 | 4078 | 1067 |
| Region D | 20184 | 92742845 | 4595 | 20184 |
| Statewide | 51947 | 220521389 | 4245 | 51947 |

## *Step 14: Use the Same Variable in both the CLASS and the VAR Statement*




## *Step 15: Use the NWAY Option*

- PROC MEANS is able to:
  - analyze numeric 'analysis' variables
  - at all potential "crossings" or "interactions" of the classification variables
    - is one of the most powerful aspects of the procedure
  - but, there are times when we want the output SAS data set to contain ONLY the highest "crossing" of the classification variables.
    - Using the NWAY Option instructs PROC MEANS to output only observations with the highest value of _TYPE_ to the new data set it is creating

# Step 15: Use the NWAY Option

```
97 proc means data=new noprint NWAY;
98 class region trans;
99 var sumrev sumkwh;
100 output out=new3 mean(sumrev) = mean_rev mean(sumkwh) = mean_kwh
101                 sum(sumrev) = sum_rev   sum(sumkwh) = sum_kwh
102                 max(sumkwh) = max_kwh  min(sumkwh) = min_kwh ;
103 run;
104
105 proc print data=new3;
106 title2 'Data Set Created by PROC MEANS';
107 title3 'Using the NWAY Option';
108 run;
```




# Step 15: Use the NWAY Option

Summarizing and Reporting Data Using the SAS System
Data Set Created by PROC MEANS
Using the NWAY Option

| Obs | REGION | TRANS | _TYPE_ | _FREQ_ | mean_rev | mean_kwh | sum_rev | sum_kwh | max_kwh | min_kwh |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | EASTERN | K1211C | 3 | 85 | 663.78 | 5359.62 | 56421.03 | 455568 | 34409 | 390 |
| 2 | EASTERN | K1233C | 3 | 427 | 703.07 | 5758.56 | 300211.68 | 2458906 | 25064 | 812 |
| 3 | NORTHERN | K1211C | 3 | 160 | 901.73 | 7652.28 | 144277.03 | 1224364 | 44301 | 1117 |
| 4 | NORTHERN | K1233 | 3 | 5 | 677.14 | 5457.00 | 3385.70 | 27285 | 5457 | 5457 |
| 5 | NORTHERN | K1233C | 3 | 455 | 788.14 | 6464.52 | 358602.69 | 2941355 | 17347 | 160 |
| 6 | SOUTHERN | K1211C | 3 | 21 | 1054.32 | 9477.14 | 22140.63 | 199020 | 17533 | 4584 |
| 7 | SOUTHERN | K1233C | 3 | 63 | 922.49 | 7622.11 | 58116.92 | 480193 | 14963 | 1386 |
| 8 | WESTERN | K12 | 3 | 4 | 1698.06 | 12900.00 | 6792.24 | 51600 | 12900 | 12900 |
| 9 | WESTERN | K1211C | 3 | 116 | 564.78 | 4636.49 | 65514.18 | 537833 | 10264 | 760 |
| 10 | WESTERN | K1233C | 3 | 381 | 649.94 | 5344.88 | 247626.88 | 2036400 | 13012 | 279 |

## Step 16: Use the NONOBS Option

- By default, PROC MEANS calculates and displays the NOBS column when you
  - Display results in the Output Window
  - Use a CLASS Statement
- The NoNobs Option suppresses calculation and display of this column




## Step 16: Use the NONOBS Option

```
 4
 5 * nonobs option in PROC MEANS;
 6
 7 proc means data=mydata.electricity maxdec=2;
 8 class region;
 9 var rev1 rev2 rev3;
10 title 'Step 16: Use NoNobs';
11 title2 'Results Without NoNobs (Default)';
12 run;
13
14 proc means data=mydata.electricity maxdec=2 NoNobs;
15 class region;
16 var rev1 rev2 rev3;
17 title 'Step 16: Use NoNobs';
18 title2 'Results With NoNobs';
19 run;
```

# Step 16: Use the NONOBS Option

- Default Output

Step 16: Use NoNobs
Results Without NoNobs (Default)

The MEANS Procedure

| REGION | N Obs | Variable | N | Mean | Std Dev | Minimum | Maximum |
|---|---|---|---|---|---|---|---|
| EASTERN | 5124 | REV1 | 5077 | 57.95 | 105.23 | 0.00 | 2798.92 |
| | | REV2 | 5089 | 55.02 | 106.84 | 0.00 | 2897.73 |
| | | REV3 | 5090 | 54.99 | 105.84 | 2.29 | 2958.40 |
| NORTHERN | 5462 | REV1 | 5428 | 74.03 | 74.07 | 3.29 | 1485.44 |
| | | REV2 | 5431 | 63.39 | 60.52 | 0.00 | 1489.19 |
| | | REV3 | 5422 | 60.05 | 59.00 | 0.00 | 1573.30 |
| SOUTHERN | 720 | REV1 | 717 | 91.92 | 72.60 | 5.00 | 499.86 |
| | | REV2 | 717 | 68.38 | 48.04 | 5.00 | 257.48 |
| | | REV3 | 720 | 56.18 | 39.51 | 5.00 | 237.51 |
| WESTERN | 5075 | REV1 | 5021 | 54.40 | 77.23 | 0.00 | 2661.67 |
| | | REV2 | 5038 | 52.07 | 70.22 | 0.00 | 2414.25 |
| | | REV3 | 5045 | 52.43 | 72.45 | 0.00 | 2548.36 |



# Step 16: Use the NONOBS Option

- With NoNobs Option

Step 16: Use NoNobs
Results With NoNobs

The MEANS Procedure

| REGION | Variable | N | Mean | Std Dev | Minimum | Maximum |
|---|---|---|---|---|---|---|
| EASTERN | REV1 | 5077 | 57.95 | 105.23 | 0.00 | 2798.92 |
| | REV2 | 5089 | 55.02 | 106.84 | 0.00 | 2897.73 |
| | REV3 | 5090 | 54.99 | 105.84 | 2.29 | 2958.40 |
| NORTHERN | REV1 | 5428 | 74.03 | 74.07 | 3.29 | 1485.44 |
| | REV2 | 5431 | 63.39 | 60.52 | 0.00 | 1489.19 |
| | REV3 | 5422 | 60.05 | 59.00 | 0.00 | 1573.30 |
| SOUTHERN | REV1 | 717 | 91.92 | 72.60 | 5.00 | 499.86 |
| | REV2 | 717 | 68.38 | 48.04 | 5.00 | 257.48 |
| | REV3 | 720 | 56.18 | 39.51 | 5.00 | 237.51 |
| WESTERN | REV1 | 5021 | 54.40 | 77.28 | 0.00 | 2661.67 |
| | REV2 | 5038 | 52.07 | 70.22 | 0.00 | 2414.25 |
| | REV3 | 5045 | 52.43 | 72.45 | 0.00 | 2548.36 |

## Step 17: Take Advantage of AUTONAME and AUTOLABEL

- We saw the AUTONAME Option in Step 9
  - Automatically assigns a unique name to variables created by PROC MEANS when you use an OUTPUT statement
    - Takes the existing analysis variable name, attaches an underscore at the end, and then attaches the statistics keyword at the end of the underscore
      - **Warning:** If the length of the analysis variable, plus the underscore and the statistics keyword exceed 32 characters, PROC MEANS will remove characters in the analysis variable name, from left to right, until the combined length of the new variable name is 32 characters.




## Step 17: Take Advantage of AUTONAME and AUTOLABEL

- AUTOLABEL
  - Automatically assigns a customized variable label to each variable created by the OUTPUT statement

# Step 17: Take Advantage of AUTONAME and AUTOLABEL

```
26 data example;
27   length kwh1-kwh3 6;
28     set sasclass.electricity(keep=cesched kwh1-kwh3 rev1-rev3 where=(cesched =: 'E'));
29 label kwh1 = 'January Kwh'
30       kwh2 = 'February Kwh'
31       kwh3 = 'March Kwh'
32       rev1 = 'January Revenue'
33       rev2 = 'February Revenue'
34       rev3 = 'March Revenue'
35       cesched = 'Customer Rate Schedule';
36       format rev1-rev3 dollar12.2 kwh1-kwh3 comma10.;
37         run;
38
39 ods select variables;
40 proc contents data=example;
41 title  'Step 17: AUTONAME and AUTOLABEL';
42 title2 'Descriptor Portion of Example Data Set';
43 run;
```




# Step 17: Take Advantage of AUTONAME and AUTOLABEL

```
Step 17: AUTONAME and AUTOLABEL
Descriptor Portion of Example Data Set

The CONTENTS Procedure

                  Alphabetic List of Variables and Attributes

#    Variable    Type    Len    Format        Informat    Label

7    CESCHED     Char      4    $4.           $4.         Customer Rate Schedule
4    REV1        Num       8    DOLLAR12.2    BEST32.     January Revenue
5    REV2        Num       8    DOLLAR12.2    BEST32.     February Revenue
6    REV3        Num       8    DOLLAR12.2    BEST32.     March Revenue
1    kwh1        Num       6    COMMA10.      BEST32.     January Kwh
2    kwh2        Num       6    COMMA10.      BEST32.     February Kwh
3    kwh3        Num       6    COMMA10.      BEST32.     March Kwh
```

# Step 17: Take Advantage of AUTONAME and AUTOLABEL

```
45 * without AUTONAME;
46 proc means data=example noprint descendtypes;
47 class cesched;
48 var kwh1-kwh3 rev1-rev3;
49 output out=ex17_a(rename=(_freq_ = count)) max= min= mean=;
50 run;
```




## Step 17: Take Advantage of AUTONAME and AUTOLABEL

```
343  * without AUTONAME;
344  proc means data=example noprint descendtypes;
345  class cesched;
346  var kwh1-kwh3 rev1-rev3;
347  output out=ex17_a(rename=(_freq_ = count)) max= min= mean=;
348  run;

WARNING: Variable kwh1 already exists on file WORK.EX17_A.
WARNING: Variable kwh2 already exists on file WORK.EX17_A.
WARNING: Variable kwh3 already exists on file WORK.EX17_A.
WARNING: Variable REV1 already exists on file WORK.EX17_A.
WARNING: Variable REV2 already exists on file WORK.EX17_A.
WARNING: Variable REV3 already exists on file WORK.EX17_A.
WARNING: Variable kwh1 already exists on file WORK.EX17_A.
WARNING: Variable kwh2 already exists on file WORK.EX17_A.
WARNING: Variable kwh3 already exists on file WORK.EX17_A.
WARNING: Variable REV1 already exists on file WORK.EX17_A.
WARNING: Variable REV2 already exists on file WORK.EX17_A.
WARNING: Variable REV3 already exists on file WORK.EX17_A.
WARNING: The duplicate variables will not be included in the output data set of the output
         statement number 1.
NOTE: There were 16366 observations read from the data set WORK.EXAMPLE.
NOTE: The data set WORK.EX17_A has 11 observations and 9 variables.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           0.12 seconds
      cpu time            0.10 seconds
```

# Step 17: Take Advantage of AUTONAME and AUTOLABEL

```
52 * with AUTONAME;
53 proc means data=example noprint descendtypes;
54 class cesched;
55 var kwh1-kwh3 rev1-rev3;
56 output out=ex17_b(rename=(_freq_ = count)) max= min= mean= / AUTONAME;
57 run;
58
59 ods select variables;
60 proc contents data=ex17_b;
61 title2 'Descriptor Portion of Data Set Created by PROC MEANS';
62 run;
```




## Step 17: Take Advantage of AUTONAME and AUTOLABEL

Step 17: AUTONAME and AUTOLABEL
Descriptor Portion of Data Set Created by PROC MEANS

The CONTENTS Procedure

Alphabetic List of Variables and Attributes

| # | Variable | Type | Len | Format | Informat | Label |
|---|---|---|---|---|---|---|
| 1 | CESCHED | Char | 4 | $4. | $4. | Customer Rate Schedule |
| 7 | REV1_Max | Num | 8 | DOLLAR12.2 | BEST32. | January Revenue |
| 19 | REV1_Mean | Num | 8 | DOLLAR12.2 | BEST32. | January Revenue |
| 13 | REV1_Min | Num | 8 | DOLLAR12.2 | BEST32. | January Revenue |
| 8 | REV2_Max | Num | 8 | DOLLAR12.2 | BEST32. | February Revenue |
| 20 | REV2_Mean | Num | 8 | DOLLAR12.2 | BEST32. | February Revenue |
| 14 | REV2_Min | Num | 8 | DOLLAR12.2 | BEST32. | February Revenue |
| 9 | REV3_Max | Num | 8 | DOLLAR12.2 | BEST32. | March Revenue |
| 21 | REV3_Mean | Num | 8 | DOLLAR12.2 | BEST32. | March Revenue |
| 15 | REV3_Min | Num | 8 | DOLLAR12.2 | BEST32. | March Revenue |
| 2 | _TYPE_ | Num | 8 | | | |
| 3 | count | Num | 8 | | | |
| 4 | kwh1_Max | Num | 8 | COMMA10. | BEST32. | January Kwh |
| 16 | kwh1_Mean | Num | 8 | COMMA10. | BEST32. | January Kwh |
| 10 | kwh1_Min | Num | 8 | COMMA10. | BEST32. | January Kwh |
| 5 | kwh2_Max | Num | 8 | COMMA10. | BEST32. | February Kwh |
| 17 | kwh2_Mean | Num | 8 | COMMA10. | BEST32. | February Kwh |
| 11 | kwh2_Min | Num | 8 | COMMA10. | BEST32. | February Kwh |
| 6 | kwh3_Max | Num | 8 | COMMA10. | BEST32. | March Kwh |
| 18 | kwh3_Mean | Num | 8 | COMMA10. | BEST32. | March Kwh |

# Step 17: Take Advantage of AUTONAME and AUTOLABEL

```
63
64 * with AUTONAME and AUTOLABEL;
65 proc means data=example noprint descendtypes;
66 class cesched;
67 var kwh1-kwh3 rev1-rev3;
68 output out=ex17_c(rename=(_freq_ = count)) max= min= mean= / AUTONAME AUTOLABEL;
69 run;
70
71 ods select variables;
72 proc contents data=ex17_c;
73 title2 'Descriptor Portion of Data Set Created by PROC MEANS';
74 title3 'Using the AUTOLABEL Option';
75 run;
```




## Step 17: Take Advantage of AUTONAME and AUTOLABEL

Descriptor Portion of Data Set Created by PROC MEANS
Using the AUTOLABEL Option

The CONTENTS Procedure

Alphabetic List of Variables and Attributes

| # | Variable | Type | Len | Format | Informat | Label |
|---|---|---|---|---|---|---|
| 1 | CESCHED | Char | 4 | $4. | $4. | Customer Rate Schedule |
| 7 | REV1_Max | Num | 8 | DOLLAR12.2 | BEST32. | January Revenue_Max |
| 19 | REV1_Mean | Num | 8 | DOLLAR12.2 | BEST32. | January Revenue_Mean |
| 13 | REV1_Min | Num | 8 | DOLLAR12.2 | BEST32. | January Revenue_Min |
| 8 | REV2_Max | Num | 8 | DOLLAR12.2 | BEST32. | February Revenue_Max |
| 20 | REV2_Mean | Num | 8 | DOLLAR12.2 | BEST32. | February Revenue_Mean |
| 14 | REV2_Min | Num | 8 | DOLLAR12.2 | BEST32. | February Revenue_Min |
| 9 | REV3_Max | Num | 8 | DOLLAR12.2 | BEST32. | March Revenue_Max |
| 21 | REV3_Mean | Num | 8 | DOLLAR12.2 | BEST32. | March Revenue_Mean |
| 15 | REV3_Min | Num | 8 | DOLLAR12.2 | BEST32. | March Revenue_Min |
| 2 | _TYPE_ | Num | 8 | | | |
| 3 | count | Num | 8 | | | |
| 4 | kwh1_Max | Num | 8 | COMMA10. | BEST32. | January Kwh_Max |
| 16 | kwh1_Mean | Num | 8 | COMMA10. | BEST32. | January Kwh_Mean |
| 10 | kwh1_Min | Num | 8 | COMMA10. | BEST32. | January Kwh_Min |
| 5 | kwh2_Max | Num | 8 | COMMA10. | BEST32. | February Kwh_Max |
| 17 | kwh2_Mean | Num | 8 | COMMA10. | BEST32. | February Kwh_Mean |
| 11 | kwh2_Min | Num | 8 | COMMA10. | BEST32. | February Kwh_Min |
| 6 | kwh3_Max | Num | 8 | COMMA10. | BEST32. | March Kwh_Max |
| 18 | kwh3_Mean | Num | 8 | COMMA10. | BEST32. | March Kwh_Mean |

## Step 18: Use the NOINHERIT Option

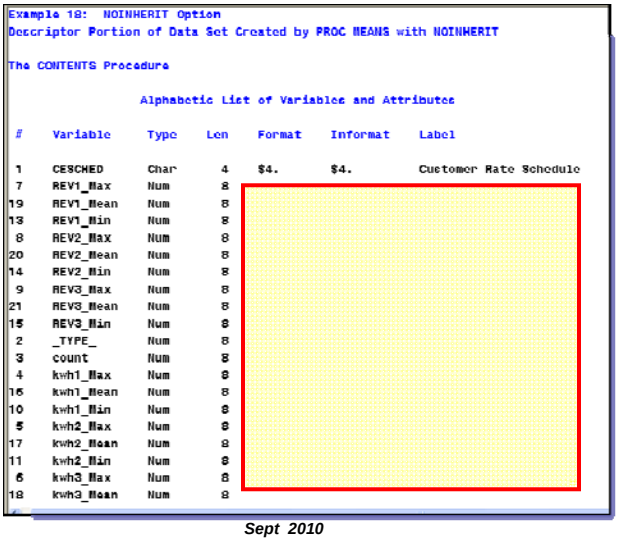
- Overrides default “inheritance” of formats and labels from the descriptor portion of the input data set to the output data set’s descriptor portion




## Step 18: Use the NOINHERIT Option

```
77 * ex 18: NOINHERIT;
78
79 proc means data=example noprint descendtypes;
80 class cesched;
81 var kwh1-kwh3 rev1-rev3;
82 output out=ex18(rename=(_freq_ = count)) max= min= mean= / AUTONAME NOINHERIT;
83 run;
84
85 ods select variables;
86 proc contents data=ex18;
87 title2 'Example 18:  NOINHERIT Option';
88 title3 'Descriptor Portion of Data Set Created by PROC MEANS with NOINHERIT';
89 run;
```

# Step 18: Use the NOINHERIT Option



Example 18: NOINHERIT Option
Descriptor Portion of Data Set Created by PROC MEANS with NOINHERIT

The CONTENTS Procedure

Alphabetic List of Variables and Attributes

| # | Variable | Type | Len | Format | Informat | Label |
|---|---|---|---|---|---|---|
| 1 | CESCHED | Char | 4 | $4. | $4. | Customer Rate Schedule |
| 7 | REV1_Max | Num | 8 | | | |
| 19 | REV1_Mean | Num | 8 | | | |
| 13 | REV1_Min | Num | 8 | | | |
| 8 | REV2_Max | Num | 8 | | | |
| 20 | REV2_Mean | Num | 8 | | | |
| 14 | REV2_Min | Num | 8 | | | |
| 9 | REV3_Max | Num | 8 | | | |
| 21 | REV3_Mean | Num | 8 | | | |
| 15 | REV3_Min | Num | 8 | | | |
| 2 | _TYPE_ | Num | 8 | | | |
| 3 | count | Num | 8 | | | |
| 4 | kwh1_Max | Num | 8 | | | |
| 16 | kwh1_Mean | Num | 8 | | | |
| 10 | kwh1_Min | Num | 8 | | | |
| 5 | kwh2_Max | Num | 8 | | | |
| 17 | kwh2_Mean | Num | 8 | | | |
| 11 | kwh2_Min | Num | 8 | | | |
| 6 | kwh3_Max | Num | 8 | | | |
| 18 | kwh3_Mean | Num | 8 | | | |



# Step 19: Use the TYPES Statement

- Very useful when you have:
  - A large analysis data set
  - Many classification variables
  - Output data sets with different combinations of the classification variables
    - Restricts the combination of class statement variable combinations to just those specified in the TYPES statement
    - Reduces CPU utilization/program completion time because SAS does not compute statistics at “unneeded” combinations of the CLASS statement variables

## Step 19: Use the TYPES Statement

- Example: 1.94 million observation customer electric utility data file
  - Six classification variables
- Requirement: Create eight output SAS data sets with sums of 12 monthly KWH variables at different combinations of the classification variables




Step 19: Use the TYPES Statement

```
91 * ex 19: TYPES Statement;
92 * bigdata3: 1.9 million record customer data set from electrical utility;
93
94 * without types;
95 proc means noprint data=mydata.bigdata3 noprint descendtypes chartype;
96 class region office trans startdate serial cesched; * 36 types;
97 format startdate year.;
98 var kwh1-kwh12 rev1-rev12;
99 * only 9 combinations of CLASS variables actually used;
100 output out=a(where=(_type_ in ('000000','000001'))) sum=/autoname autolabel;
101 output out=b(where=(_type_ in ('000000','000011'))) sum=/autoname autolabel;
102 output out=c(where=(_type_ in ('000000','000111'))) sum=/autoname autolabel;
103 output out=d(where=(_type_ in ('000000','001111'))) sum=/autoname autolabel;
104 output out=e(where=(_type_ in ('000000','011111'))) sum=/autoname autolabel;
105 output out=f(where=(_type_ in ('000000','111111'))) sum=/autoname autolabel;
106 output out=g(where=(_type_ in ('000000','100001'))) sum=/autoname autolabel;
107 output out=h(where=(_type_ in ('000000','010001'))) sum=/autoname autolabel;
108 run;
```


Sept 2010

## Step 19: Use the TYPES Statement

```
418  proc means noprint data=mydata.bigdata3 noprint descendtypes chartype;
419  class region office trans startdate serial cesched;
420  format startdate year.;
421  var kwh1-kwh12 rev1-rev12;
422  output out=a(where=(_type_ in ('000000','000001'))) sum=/autoname autolabel;
423  output out=b(where=(_type_ in ('000000','000011'))) sum=/autoname autolabel;
424  output out=c(where=(_type_ in ('000000','000111'))) sum=/autoname autolabel;
425  output out=d(where=(_type_ in ('000000','001111'))) sum=/autoname autolabel;
426  output out=e(where=(_type_ in ('000000','011111'))) sum=/autoname autolabel;
427  output out=f(where=(_type_ in ('000000','111111'))) sum=/autoname autolabel;
428  output out=g(where=(_type_ in ('000000','100001'))) sum=/autoname autolabel;
429  output out=h(where=(_type_ in ('000000','010001'))) sum=/autoname autolabel;
430  run;

NOTE: There were 1944174 observations read from the data set MYDATA.BIGDATA3.
NOTE: The data set WORK.A has 13 observations and 32 variables.
NOTE: The data set WORK.B has 112 observations and 32 variables.
NOTE: The data set WORK.C has 4790 observations and 32 variables.
NOTE: The data set WORK.D has 25946 observations and 32 variables.
NOTE: The data set WORK.E has 112107 observations and 32 variables.
NOTE: The data set WORK.F has 112107 observations and 32 variables.
NOTE: The data set WORK.G has 31 observations and 32 variables.
NOTE: The data set WORK.H has 123 observations and 32 variables.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           46.34 seconds
      cpu time            22.47 seconds
```




## Step 19: Use the TYPES Statement

```
111 proc means noprint data=mydata.bigdata3 noprint descendtypes chartype;
112 class region office trans startdate serial cesched;
113 format startdate year.;
114 var kwh1-kwh12 rev1-rev12;
115 types    ()
116          cesched
117          serial * cesched
118          startdate * serial * cesched
119          trans * startdate * serial * cesched
120          office * trans * startdate * serial * cesched
121          region *  office * trans * startdate * serial * cesched
122          region * cesched
123          office * cesched;  * 9 types;
124 output out=a(where=(_type_ in ('000000','000001'))) sum=/autoname autolabel;
125 output out=b(where=(_type_ in ('000000','000011'))) sum=/autoname autolabel;
126 output out=c(where=(_type_ in ('000000','000111'))) sum=/autoname autolabel;
127 output out=d(where=(_type_ in ('000000','001111'))) sum=/autoname autolabel;
128 output out=e(where=(_type_ in ('000000','011111'))) sum=/autoname autolabel;
129 output out=f(where=(_type_ in ('000000','111111'))) sum=/autoname autolabel;
130 output out=g(where=(_type_ in ('000000','100001'))) sum=/autoname autolabel;
131 output out=h(where=(_type_ in ('000000','010001'))) sum=/autoname autolabel;
132 run;
```

### Step 19: Use the TYPES Statement

```
516  proc means noprint data=mydata.bigdata3 noprint descendtypes chartype;
517  class region office trans startdate serial cesched;
518  format startdate year.;
519  var kwh1-kwh12 rev1-rev12;
520  types    ()
521           cesched
522           serial * cesched
523           startdate * serial * cesched
524           trans * startdate * serial * cesched
525           office * trans * startdate * serial * cesched
526           region *  office * trans * startdate * serial * cesched
527           region * cesched
528           office * cesched;  * 9 types;
529  output out=a(where=(_type_ in ('000000','000001'))) sum=/autoname autolabel;
530  output out=b(where=(_type_ in ('000000','000011'))) sum=/autoname autolabel;
531  output out=c(where=(_type_ in ('000000','000111'))) sum=/autoname autolabel;
532  output out=d(where=(_type_ in ('000000','001111'))) sum=/autoname autolabel;
533  output out=e(where=(_type_ in ('000000','011111'))) sum=/autoname autolabel;
534  output out=f(where=(_type_ in ('000000','111111'))) sum=/autoname autolabel;
535  output out=g(where=(_type_ in ('000000','100001'))) sum=/autoname autolabel;
536  output out=h(where=(_type_ in ('000000','010001'))) sum=/autoname autolabel;
537  run;
```



## Step 19: Use the TYPES Statement

```
NOTE: There were 1944174 observations read from the data set MYDATA.BIGDATA3.
NOTE: The data set WORK.A has 13 observations and 32 variables.
NOTE: The data set WORK.B has 112 observations and 32 variables.
NOTE: The data set WORK.C has 4790 observations and 32 variables.
NOTE: The data set WORK.D has 25946 observations and 32 variables.
NOTE: The data set WORK.E has 112107 observations and 32 variables.
NOTE: The data set WORK.F has 112107 observations and 32 variables.
NOTE: The data set WORK.G has 31 observations and 32 variables.
NOTE: The data set WORK.H has 123 observations and 32 variables.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           1:32.50
      cpu time            18.59 seconds
```

## Step 20: Using the IDGROUP Option

- Using the IDGROUP Option
  - This option allows output of new variables to a data set created by PROC MEANS.
    - Combines the ID and IDMIN options in the PROC MEANS statement and the MAXID and MINID options in the OUTPUT Statement
    - Allows identification of extreme values of the analysis variables
    - The OUT[N] option controls the number of extreme observations to be “captured” as variables in the output data set




## Step 20: Use the IDGROUP Option

- Example:
  - Using the Social Services Payments Data Set
    - Obtain the mean, two largest and two smallest payments for mental health services by region
    - Save results in a SAS data set
    - Display data set in output window using PROC PRINT

# Step 20: Use the IDGROUP Option

```
287 * use IDGROUP Option to Obtain two largest and two smallest
288   mental health payments by Region (formatted value of legislative district);
289 * using the descendtypes option;
290 proc means noprint
291       data=sasclass.social_services_payments(where=(mental_health_dollars > 0))
292          descendtypes;
293 class legislative;
294 format legislative $regfmt.;
295 var mental_health_dollars;
296 output out=new6(drop=_type_) mean=mean_pay
297        idgroup (max(mental_health_dollars) out[2]
298            (mental_health_dollars) = max_pay)
299        idgroup (min(mental_health_dollars) out[2]
300            (mental_health_dollars) = min_pay);
301 run;
302
303
304 proc print data=new6 noobs;
305 title1 'Intro to Data Summarziation and Reporting';
306 title2 'Using the IDGROUP Option';
307 run;
```



# Step 20: Use the IDGROUP Option

Intro to Data Summarziation and Reporting
Using the IDGROUP Option

| legislative | _FREQ_ | mean_pay | max_pay_1 | max_pay_2 | min_ pay_1 | min_ pay_2 |
|---|---|---|---|---|---|---|
| Region I | 4158 | 3465.52 | 182374.47 | 182169.83 | 3.07 | 4.76 |
| Region II | 5554 | 2105.00 | 184345.46 | 183929.70 | 3.17 | 4.09 |
| Region III | 4267 | 5630.60 | 540492.26 | 494147.55 | 5.12 | 5.12 |
| Region IV | 4812 | 4601.13 | 235114.05 | 214164.62 | 1.02 | 1.02 |
| Region V | 3204 | 3724.31 | 186704.15 | 185916.04 | 4.09 | 5.12 |
| | 21995 | 3828.14 | 540492.26 | 494147.55 | 1.02 | 1.02 |

## *Conclusions*

- PROC MEANS is a VERY powerful BASE SAS Procedure
  - The more you know about, the more you want to know about it
  - We've covered some of the basic functions/features, there are many other PROC MEANS capabilities I cover in my "Beyond the Basics" talk.




## *Thank you for attending!*

- Questions?
- Comments?
- Copies of this and other presentations?
  - "Free Downloads" link at
    - www.SierraInformation.com