SAS Programming (BIOL-4V190)

Chapter 16

Summarizing Your Data

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.1 Introduction

PROC MEANS can be used to generate a basic statistics report as well as to create data sets containing the values of these statistics.

PROC SUMMARY can be used to generate the same data sets if printed output is not needed.
PROC SUMMARY is identical to PROC MEANS NOPRINT.

16.2 PROC MEANS—Starting from the Beginning

By default, PROC MEANS generates a table of basic statistics (n, mean, std dev, min, max) on all numeric variables in the data set.

Adding the VAR statement controls the variables which will appear in the output.

Statistics that appear in the output can be controlled using options on the PROC MEANS statement.

Some of the statistics available are shown on page 321.

A complete list of the statistics available can be found in the SAS online documentation.

Syntax:

```
proc means options;
  var variablename1 variablename2...variablename'n';
run;
```

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

```
*Program 16-2 Adding a VAR statement and requesting specific statistics with PROC MEANS - page 322;
title "Selected Statistics Using PROC MEANS";
proc means data=learn.blood n nmiss mean median
                            min max maxdec=1;
   var RBC WBC;
run;
```

In this example, the requested statistics appear in the output in the same order in which they are listed on the PROC MEANS statement.

The MAXDEC= option is used to control the number of decimal places that are printed.

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.3 Adding a BY Statement to PROC MEANS

Adding a BY statement to PROC MEANS causes the statistics to be calculated within subsets of the data.

The subsets are created based on the different values in each BY group.

As with all other procedures, using a BY statement in PROC MEANS requires the data to first be sorted.

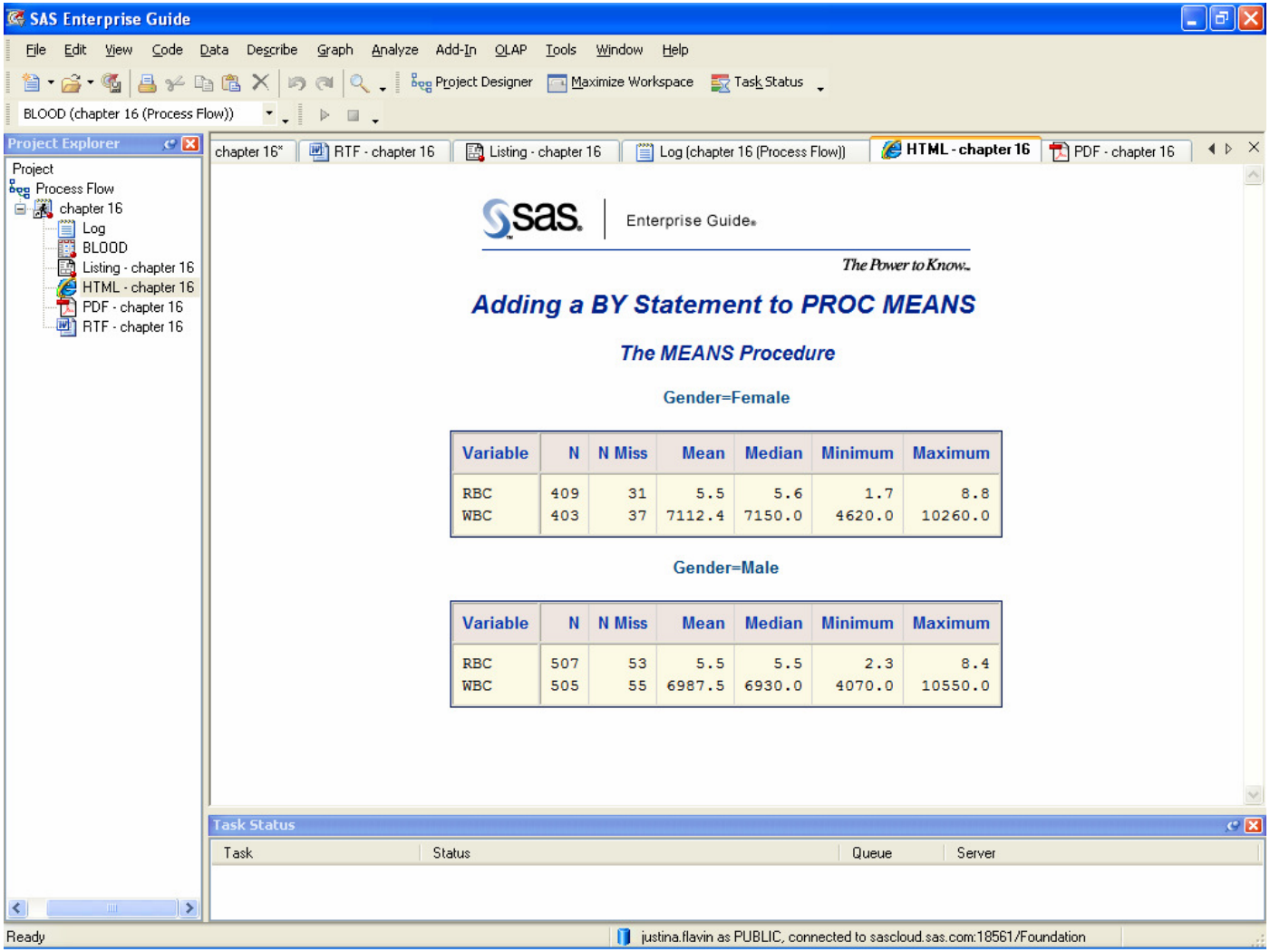
Syntax:

```
proc sort;
  by variablename1 variablename2...variablename'n';
run;

proc means options;
  by variablename1 variablename2...variablename'n';
  var variablename1 variablename2...variablename'n';
run;
```

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

Adding the BY Gender statement to Program 16-3 causes the statistics to be calculated separately for each BY group (Females, Males).



Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.4 Using a CLASS Statement with PROC MEANS

The CLASS statement can be used in place of the BY statement.

The main advantage to doing this is that the data do not need to be sorted prior to running PROC MEANS.

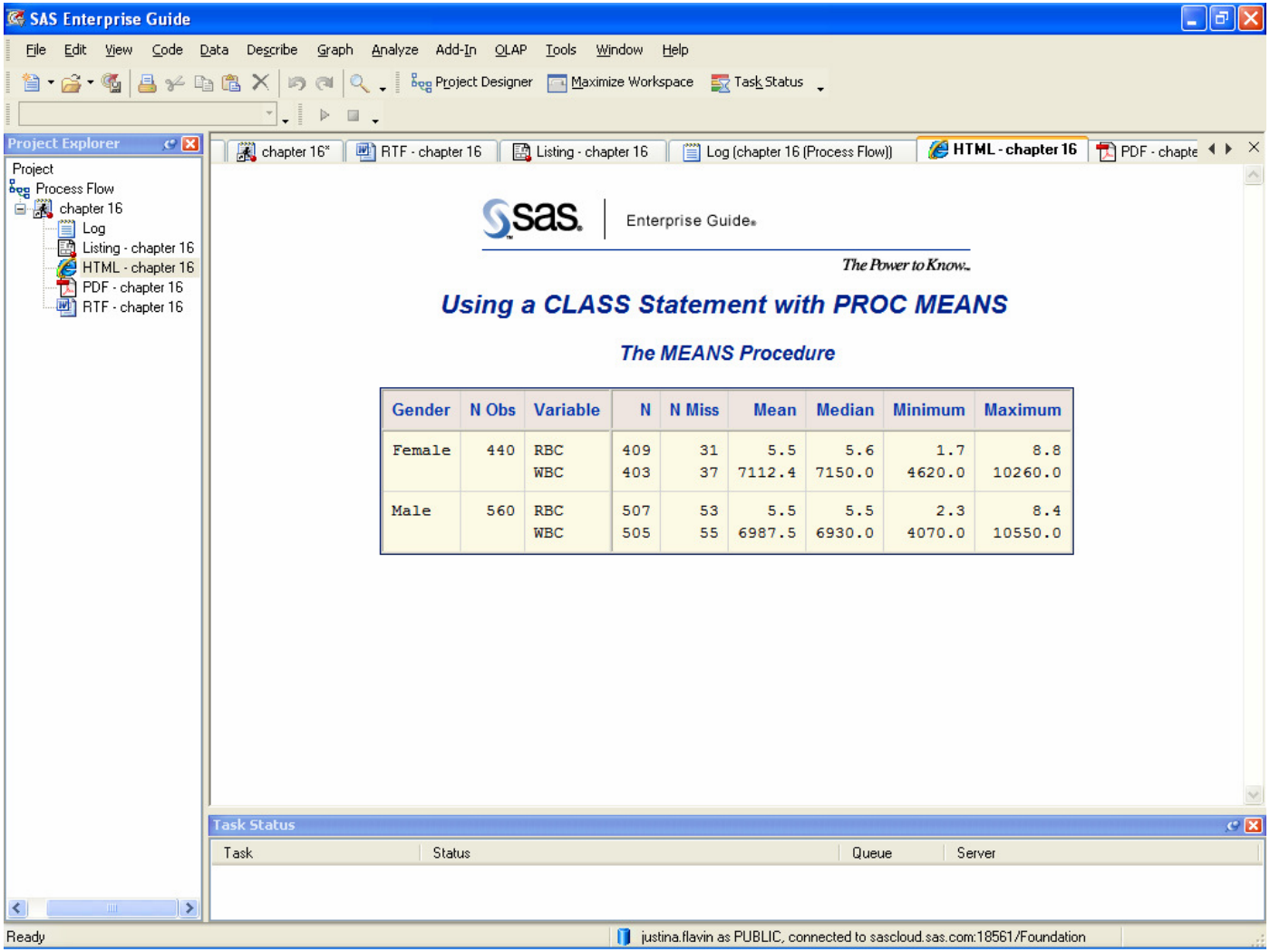
The format of the output is slightly different but contains the same information.

Syntax:

```
proc means options;
  class variablename1 variablename2...variablename'n';
  var variablename1 variablename2...variablename'n';
run;
```

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

Notice in the sample code that this output was generated on the unsorted data in LEARN.BLOOD
The layout is slightly different, but the output is identical to that produced using a BY statement



Gender	N Obs	Variable	N	N Miss	Mean	Median	Minimum	Maximum
Female	440	RBC	409	31	5.5	5.6	1.7	8.8
		WBC	403	37	7112.4	7150.0	4620.0	10260.0
Male	560	RBC	507	53	5.5	5.5	2.3	8.4
		WBC	505	55	6987.5	6930.0	4070.0	10550.0

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.5 Applying a Format to a CLASS Variable

A format can be added to a CLASS variable by using a FORMAT statement.
Adding a format to a class variable can be used to change how the CLASS variable groups the data without having to change the data in the data set.

Syntax:

```
proc means options;
  class variablename1 variablename2...variablename'n';
  var variablename1 variablename2...variablename'n';
  format variablename1 format. variablename2 $format.;
run;
```

In our example, the cholesterol variable value will be categorized into one of two groups (low and high) using a format. Then applying this format to the variable in PROC MEANS will cause the summary statistics to be generated for the two groups.

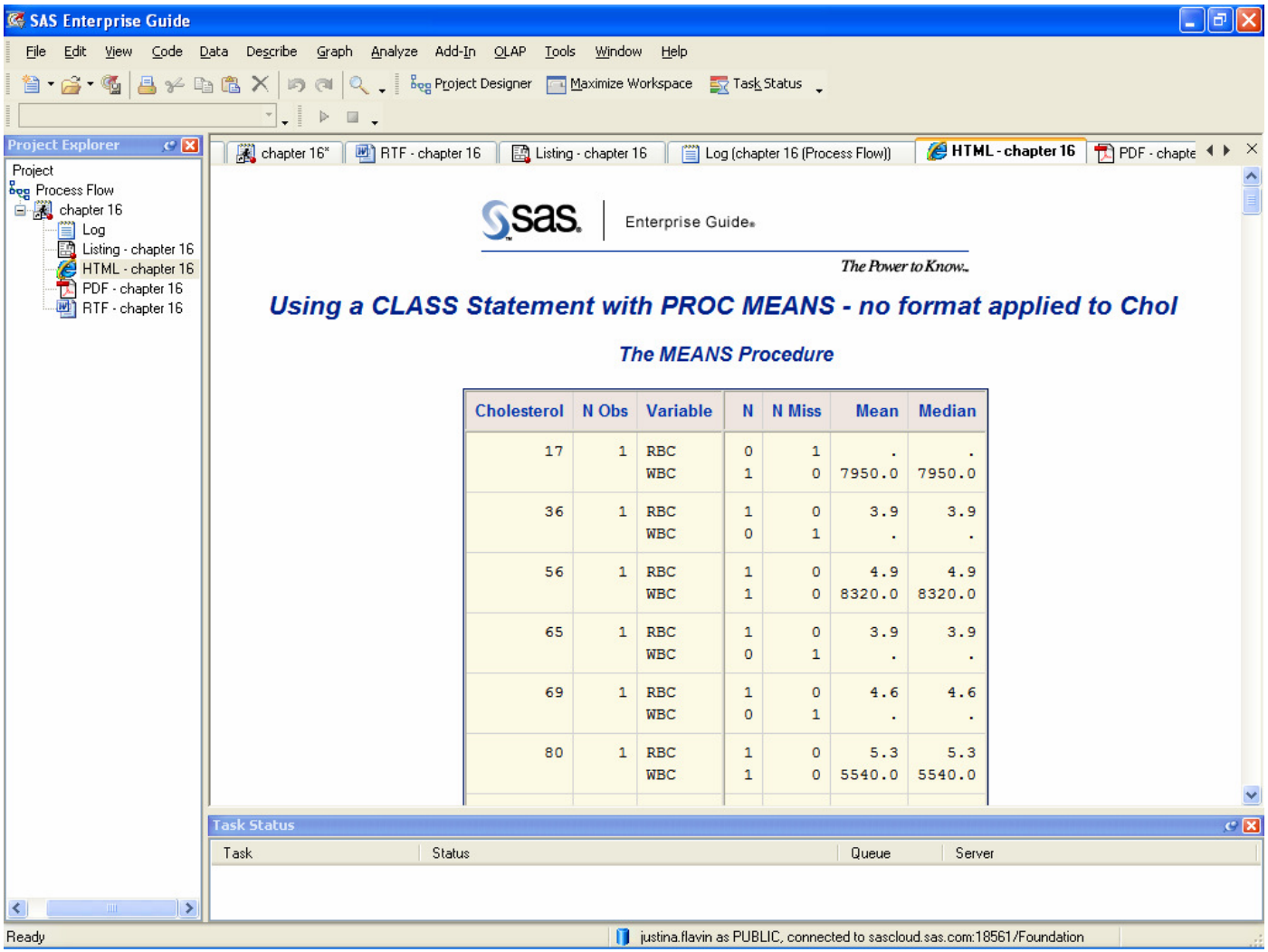
```
*Program 16-5 Demonstrating the effect of a formatted CLASS variable - page 326;
proc format;
   value chol_group
     low -< 200 = 'Low'
     200 - high = 'High';
run;
```

Note: This technique also works when using a BY statement instead of a CLASS statement.

Creating a format and applying it in PROC MEANS allows you to turn a continuous variable into a categorical variable to get meaningful results and to make meaningful comparisons.
Compare the output on page 326 to the output generated without the FORMAT statement.

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

Without a format applied to Chol, PROC MEANS generates summary statistics for every unique nonmissing value of Chol in the data set.



Cholesterol	N Obs	Variable	N	N Miss	Mean	Median
17	1	RBC	0	1	.	.
		WBC	1	0	7950.0	7950.0
36	1	RBC	1	0	3.9	3.9
		WBC	0	1	.	.
56	1	RBC	1	0	4.9	4.9
		WBC	1	0	8320.0	8320.0
65	1	RBC	1	0	3.9	3.9
		WBC	0	1	.	.
69	1	RBC	1	0	4.6	4.6
		WBC	0	1	.	.
80	1	RBC	1	0	5.3	5.3
		WBC	1	0	5540.0	5540.0

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.6 Deciding between a BY Statement and a CLASS Statement

The general rule of thumb is to use a CLASS statement since it eliminates the need for a PROC SORT.
If the data are already correctly sorted on the values of the BY variables, using a BY statement is more efficient.

16.7 Creating Summary Data Sets Using PROC MEANS

Summary statistics can be routed to data sets by using an OUTPUT statement.

Adding the NOPRINT option to the PROC MEANS statement will cause the output to be suppressed and only a data set will be generated.

Statistics are specified using the same statistics keywords that are used on the PROC MEANS line.

The statistics keywords on the PROC MEANS line can be the same or different than the statistics that will be written to the data set.

Syntax:

```
proc means noprint;
  var variablename1 variablename2...variablename'n';
  output out=datasetname
         statistics-keyword = variablename1 variablename2...variablename'n';
run;
```

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

```
proc means data=learn.blood noprint;
   var RBC WBC;
   output out = my_summary
          mean = MeanRBC MeanWBC;
run;
```

In this example, the mean values of RBC and WBC will be saved in the variables MeanRBC and MeanWBC in the output data set my_summary.

This output data set is shown on page 328.

By default, there are two automatic variables, _TYPE_ and _FREQ_, that are also written to the output data set.

These automatic variables will be discussed is a later section.

Notice that if the noprint option is removed, the procedure also generates the default output.

Specifying statistics on the PROC MEANS statement only affects the printed output.

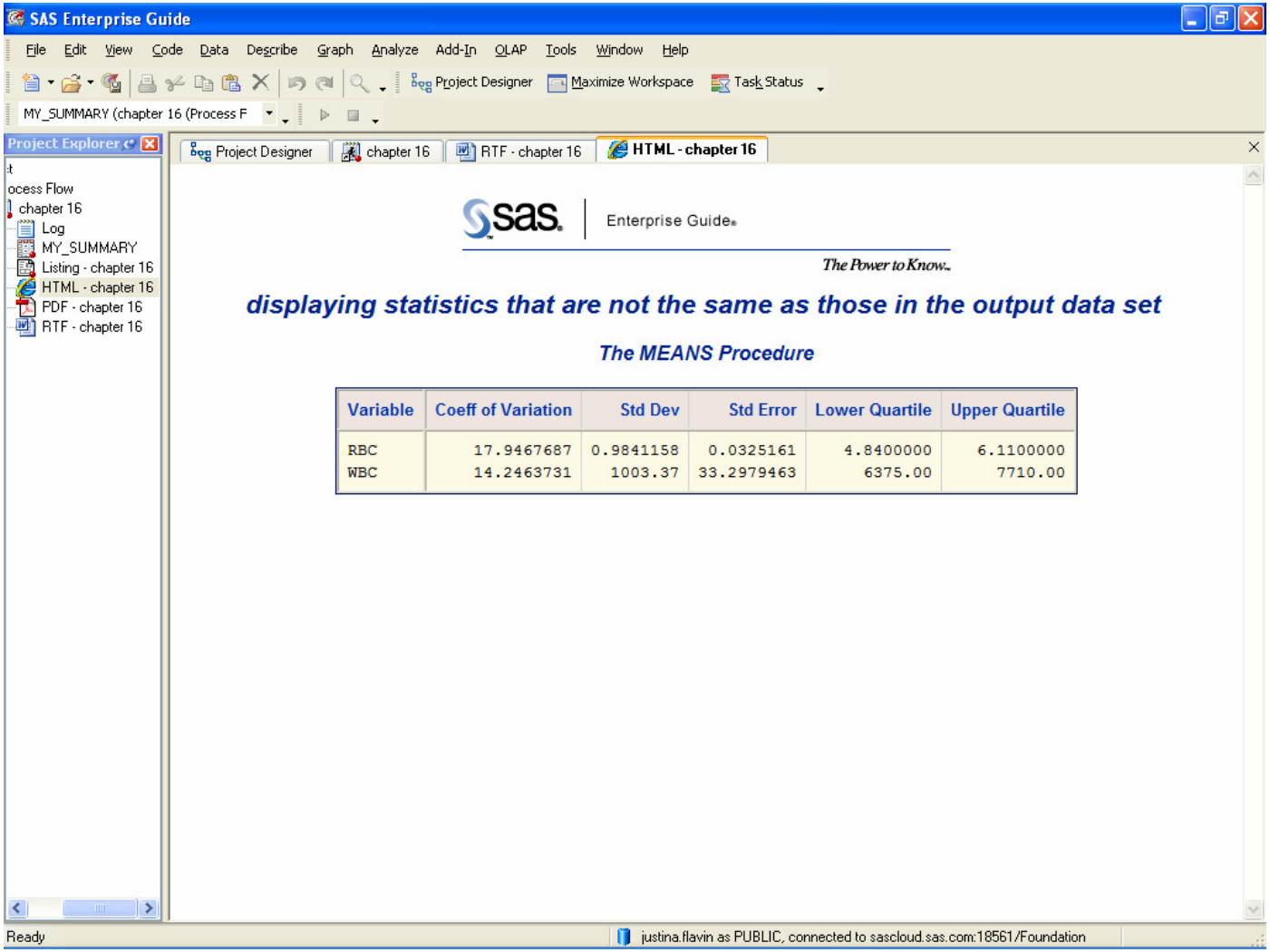
There is no effect on the output data set.

Thus is it possible to generate output that contains different statistics than those that are included in the output data set

```
proc means data=learn.blood cv std stderr q1 q3;
   title displaying statistics that are not the same as those in the output data set;
   var RBC WBC;
   output out = my_summary
          mean = MeanRBC MeanWBC;
run;
```

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

Here is the output. The statistics are those that were specified on the PROC MEANS statement.

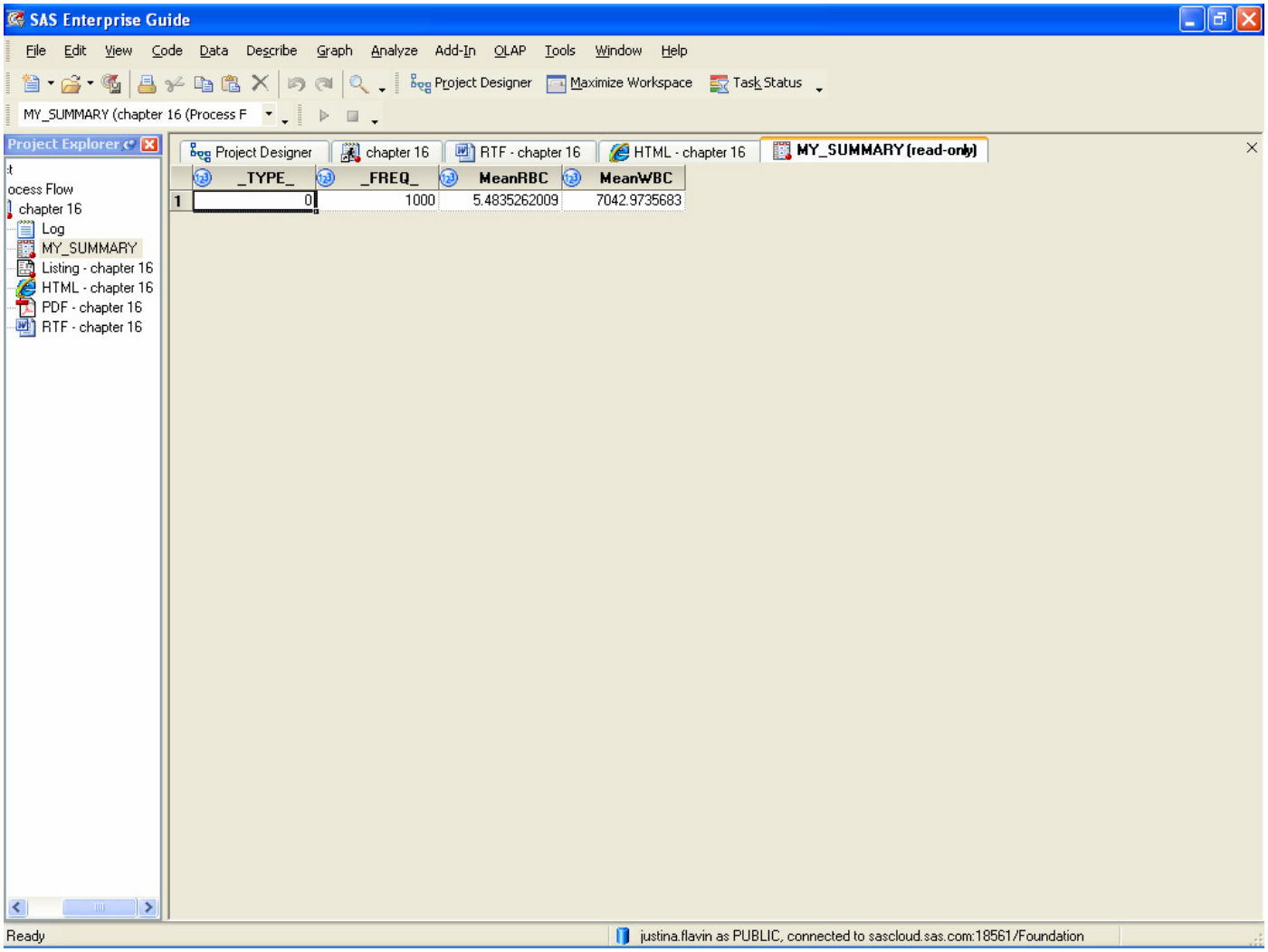


Variable	Coeff of Variation	Std Dev	Std Error	Lower Quartile	Upper Quartile
RBC	17.9467687	0.9841158	0.0325161	4.8400000	6.1100000
WBC	14.2463731	1003.37	33.2979463	6375.00	7710.00

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

Here is the data set MY_SUMMARY.

It has the variables MeanRBC and MeanWBC which were specified on the output statement.



Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.8 Outputting Other Descriptive Statistics with PROC MEANS

More than one statistic can be written to the output data set.

Different statistics can be selected for each variable on the VAR statement.

Syntax:

```
proc means;
  var variablename1 variablename2...variablename'n';
  output out=datasetname
         statistics-keyword1 = variablename1 variablename2...variablename'n'
         statistics-keyword2 = variablename1 variablename2...variablename'n'
         statistics-keyword'n' = variablename1 variablename2...variablename'n';
run;
```

This is illustrated in Program 16-7 on page 329.

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

The previous example illustrated how to obtain the same statistics for every variable on the VAR statement.

There is also a way to specify different statistics for each variable, by modifying the output statement as shown below.

```
output out=datasetname
    statistics-keyword1(variablename1 ) = newvariablename1
    statistics-keyword2(variablename1 variablename2) = newvariablename1 newvariablename2
```

Placing the variable name or names inside parentheses following the statistics keyword causes SAS to calculate the requested statistic for only the variables listed rather than all variables on the VAR statement.

```
/* Creating a data set that has different statistics for each variable */

proc means data=blood;
   var RBC WBC chol;
   output out = diff_stats
          mean(wbc chol)= dog cat
          stddev(chol) =
          n        = N_RBC N_WBC
          nmiss    = Miss_RBC Miss_WBC
          median   = Med_RBC Med_WBC;
run;
```

In this example, the mean is only calculated for WBC and chol. The mean of wbc is saved in DOG and the mean of chol is saved in CAT.

An output variable name is not specified for the std dev of chol, so by default, the output variable will have the same name as the input variable (chol).

Only two output variable names are provided for n, nmiss, and median, so SAS calculates these values for the first two variables listed on the var statement.

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.9 Asking SAS to Name the Variables in the Output Data Set

Using the AUTONAME option on the OUTPUT statement causes SAS to generate the variable names in the output data set.
If variable names are specified after *statistics-keyword,* then those variable names are used.
However, if not enough of these are specified, AUTONAME will automatically generate the remaining variable names.

Syntax:

```
proc means;
  var variablename1 variablename2...variablename'n';
  output out=datasetname
         statistics-keyword1 =
         statistics-keyword2 =
         statistics-keyword'n' = / autoname;
run;
```

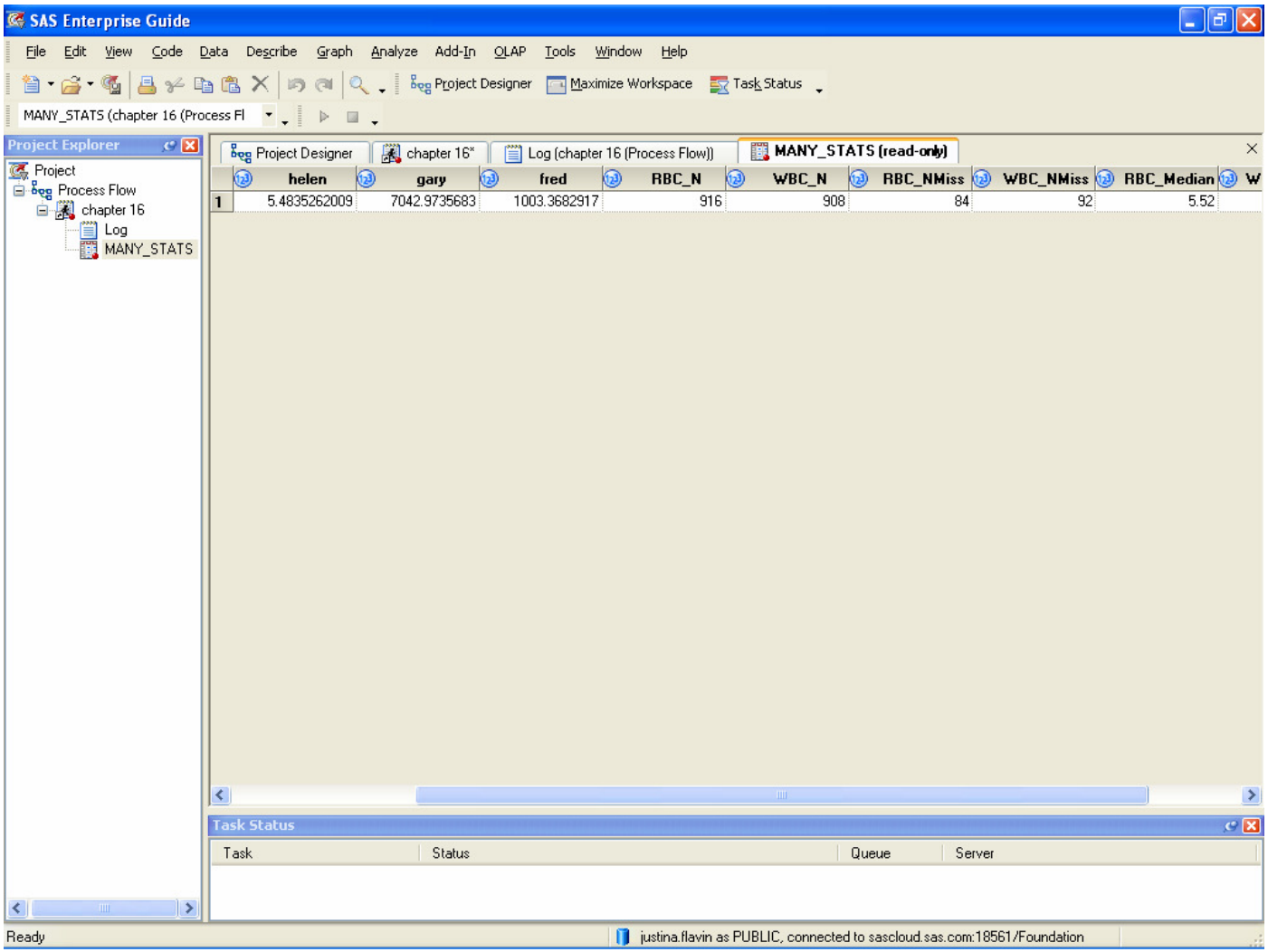
The output data set with the auto-generated variable names for Program 16-8 is shown on page 330.

Here is a different example:

```
proc means data=learn.blood noprint;
   var RBC WBC;
   output out = many_stats
          mean     = helen gary
          stddev(wbc) = fred
          n        =
          nmiss    =
          median   = / autoname;
run;
```

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

In this example, some variable names are provided. SAS uses those variable names in the output data set and auto-generates the remainder of the variable names.



Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.10 Outputting a Summary Data Set: Including a BY Statement

Adding a BY statement will cause the statistics to be calculated and output by subsetting the data based on the different values in each BY group.

Syntax:

```
proc means;
  by variablename1 variablename2...variablename'n';
  var variablename1 variablename2...variablename'n';
  output out=datasetname
         statistics-keyword1 = variablename1 variablename2...variablename'n'
         statistics-keyword2 = variablename1 variablename2...variablename'n'
         statistics-keyword'n' = variablename1 variablename2...variablename'n';
run;
```

This is illustrated in Program 16-9 on page 331.

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.11 Outputting a Summary Data Set: Including a CLASS Statement

A CLASS statement can also be used in place of the BY statement when creating output data sets.

The resulting data set is slightly different unless the NWAY option is used on the PROC MEANS statement.

Syntax:

```
proc means < noprint nway >;
  class variablename1 variablename2...variablename'n';
  var variablename1 variablename2...variablename'n';
  output out=datasetname
         statistics-keyword1 = variablename1 variablename2...variablename'n'
         statistics-keyword2 = variablename1 variablename2...variablename'n'
         statistics-keyword'n' = variablename1 variablename2...variablename'n';
run;
```

Programs 16-10 and 16-11 illustrate the use of a CLASS statement with and without the NWAY option.

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.12 Using Two CLASS Variables with PROC MEANS

When there are two or more variables on the CLASS statement, the automatic variable _TYPE_ in the output data set can be used to identify which rows correspond to which combinations of the different levels of the variables specified on the CLASS statement.

This can be useful for subsetting the data appropriately if multiple output data sets are desired.

The values of _TYPE_ can be converted to a binary representation by adding the CHARTYPE option on the PROC MEANS statement.

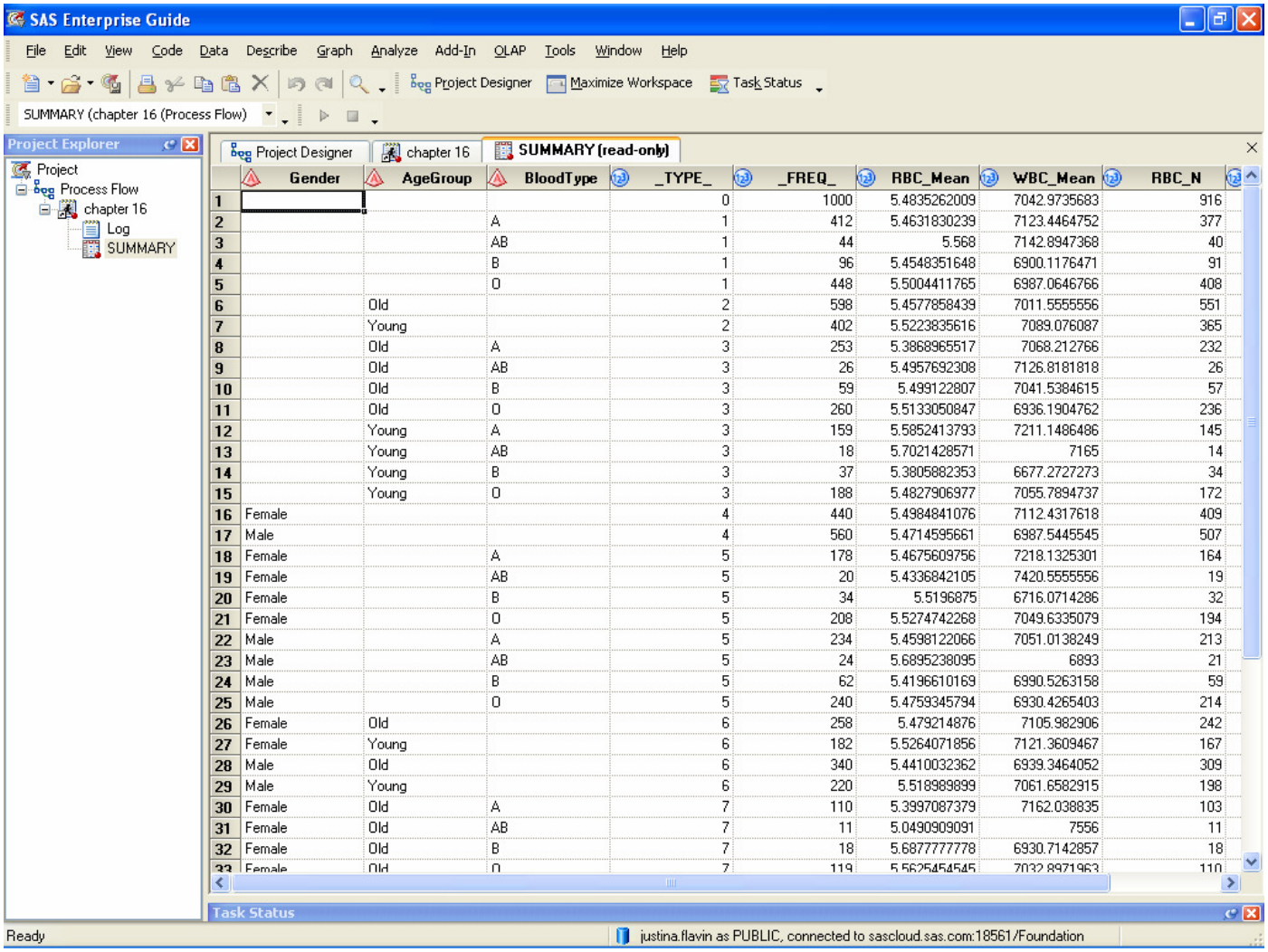
The examples in this section have been modified to better illustrate the concepts.

```
*Modified Program 16-12 Using 3 CLASS variables with PROC MEANS;
proc means data=learn.blood noprint;
   class Gender AgeGroup bloodtype;
   var RBC WBC;
   output out = summary
          mean =
          n = / autoname;
run;
```

In this example, there are three class variables and we will look at the values of _TYPE_ in the output data set.

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

Here is the output data set SUMMARY.



	Gender	AgeGroup	BloodType	_TYPE_	_FREQ_	RBC_Mean	WBC_Mean	RBC_N
1				0	1000	5.4835262009	7042.9735683	916
2			A	1	412	5.4631830239	7123.4464752	377
3			AB	1	44	5.568	7142.8947368	40
4			B	1	96	5.4548351648	6900.1176471	91
5			O	1	448	5.5004411765	6987.0646766	408
6		Old		2	598	5.4577858439	7011.5555556	551
7		Young		2	402	5.5223835616	7089.076087	365
8		Old	A	3	253	5.3868965517	7068.212766	232
9		Old	AB	3	26	5.4957692308	7126.8181818	26
10		Old	B	3	59	5.499122807	7041.5384615	57
11		Old	O	3	260	5.5133050847	6936.1904762	236
12		Young	A	3	159	5.5852413793	7211.1486486	145
13		Young	AB	3	18	5.7021428571	7165	14
14		Young	B	3	37	5.3805882353	6677.2727273	34
15		Young	O	3	188	5.4827906977	7055.7894737	172
16	Female			4	440	5.4984841076	7112.4317618	409
17	Male			4	560	5.4714595661	6987.5445545	507
18	Female		A	5	178	5.4675609756	7218.1325301	164
19	Female		AB	5	20	5.4336842105	7420.5555556	19
20	Female		B	5	34	5.5196875	6716.0714286	32
21	Female		O	5	208	5.5274742268	7049.6335079	194
22	Male		A	5	234	5.4598122066	7051.0138249	213
23	Male		AB	5	24	5.6895238095	6893	21
24	Male		B	5	62	5.4196610169	6990.5263158	59
25	Male		O	5	240	5.4759345794	6930.4265403	214
26	Female	Old		6	258	5.479214876	7105.982906	242
27	Female	Young		6	182	5.5264071856	7121.3609467	167
28	Male	Old		6	340	5.4410032362	6939.3464052	309
29	Male	Young		6	220	5.518989899	7061.6582915	198
30	Female	Old	A	7	110	5.3997087379	7162.038835	103
31	Female	Old	AB	7	11	5.0490909091	7556	11
32	Female	Old	B	7	18	5.6877777778	6930.7142857	18
33	Female	Old	O	7	119	5.5625454545	7032.8971963	110

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

The online documentation for PROC MEANS provides a chart under **Results: MEANS Procedure**. There are three class variables A,B,C. In our example, A=BloodType, B=AgeGroup, C=Gender. A “0” in the variable column indicates that the variable is not used in calculating the summary statistics for that row. So _TYPE_=0 provides summary statistics without using any class variables. _TYPE_=1 provides summary statistics for the levels within Variable A. _TYPE_=2 provides summary statistics for the levels within Variable B…. _TYPE_=6 provides summary statistics for the combination of levels within Variables B and C.

Documentation - Windows Internet Explorer

http://support.sas.com/91doc/docMainpage.jsp

Contents | Index | Search

The Effect of Class Variables on the OUTPUT Data Set

three CLASS variables
two CLASS variables
one CLASS variable

C	B	A	_WAY_	_TYPE_	Subgroup defined by	Number of observations of this _TYPE_ and _WAY_ in the data set	Total number of observations in the data set
0	0	0	0	0	Total	1	
0	0	1	1	1	A	a	1+a
0	1	0	1	2	B	b	
0	1	1	2	3	A*B	a*b	1+a+b+a*b
1	0	0	1	4	C	c	
1	0	1	2	5	A*C	a*c	
1	1	0	2	6	B*C	b*c	1+a+b+a*b+c
1	1	1	3	7	A*B*C	a*b*c	+a*c+b*c+a*b*c
Character binary equivalent of _TYPE_ (CHARTYPE option)					A, B, C = CLASS variables	a, b, c, = number of levels of A, B, C, respectively	

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

Adding the CHARTYPE option to the code changes the value of _TYPE_ to a character representation of the binary value of _TYPE_.

The length of the variable equals the number of class variables.

So now _TYPE_ becomes a variable that contains only the digits 0 and 1, with as many digits as there are CLASS variables.

A “0” indicates that the variable’s levels are not used to stratify the data in the calculation of the statistics on that observation, while a “1” indicates that the variable’s levels are used to stratify the data in the calculation of the statistics on that observation.

So _TYPE_=011 indicates that the levels of the 2nd (AgeGroup) and 3rd (bloodtype) variables on the class statement are used to stratify the data in the calculation of the statistics on that observation.

The actual level(s) used for that row are shown in the cell for each variable.

```
*Modified Program 16-13 Adding the CHARTYPE procedure option to PROC MEANS - page 334;
proc means data=learn.blood noprint chartype;
   class Gender AgeGroup bloodtype;
   var RBC WBC;
   output out = summary
          mean =
          n = / autoname;
run;
```

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

Here is the data set SUMMARY with the character representation of the binary value of _TYPE_.

SAS Enterprise Guide

File Edit View Code Data Describe Graph Analyze Add-In OLAP Tools Window Help

Project Designer | Maximize Workspace | Task Status

SUMMARY (chapter 16 (Process Flow)

Project Explorer

- Project
- Process Flow
  - chapter 16
    - Log
    - SUMMARY

Project Designer | chapter 16* | SUMMARY (read-only)

	Gender	AgeGroup	BloodType	_TYPE_	_FREQ_	RBC_Mean	WBC_Mean	RBC_N
1				000	1000	5.4835262009	7042.9735683	916
2			A	001	412	5.4631830239	7123.4464752	377
3			AB	001	44	5.568	7142.8947368	40
4			B	001	96	5.4548351648	6900.1176471	91
5			O	001	448	5.5004411765	6987.0646766	408
6		Old		010	598	5.4577858439	7011.5555556	551
7		Young		010	402	5.5223835616	7089.076087	365
8		Old	A	011	253	5.3868965517	7068.212766	232
9		Old	AB	011	26	5.4957692308	7126.8181818	26
10		Old	B	011	59	5.499122807	7041.5384615	57
11		Old	O	011	260	5.5133050847	6936.1904762	236
12		Young	A	011	159	5.5852413793	7211.1486486	145
13		Young	AB	011	18	5.7021428571	7165	14
14		Young	B	011	37	5.3805882353	6677.2727273	34
15		Young	O	011	188	5.4827906977	7055.7894737	172
16	Female			100	440	5.4984841076	7112.4317618	409
17	Male			100	560	5.4714595661	6987.5445545	507
18	Female		A	101	178	5.4675609756	7218.1325301	164
19	Female		AB	101	20	5.4336842105	7420.5555556	19
20	Female		B	101	34	5.5196875	6716.0714286	32
21	Female		O	101	208	5.5274742268	7049.6335079	194
22	Male		A	101	234	5.4598122066	7051.0138249	213
23	Male		AB	101	24	5.6895238095	6893	21
24	Male		B	101	62	5.4196610169	6990.5263158	59
25	Male		O	101	240	5.4759345794	6930.4265403	214
26	Female	Old		110	258	5.479214876	7105.982906	242
27	Female	Young		110	182	5.5264071856	7121.3609467	167
28	Male	Old		110	340	5.4410032362	6939.3464052	309
29	Male	Young		110	220	5.518989899	7061.6582915	198
30	Female	Old	A	111	110	5.3997087379	7162.038835	103
31	Female	Old	AB	111	11	5.0490909091	7556	11
32	Female	Old	B	111	18	5.6877777778	6930.7142857	18
33	Female	Old	O	111	119	5.5625454545	7032.8971963	110

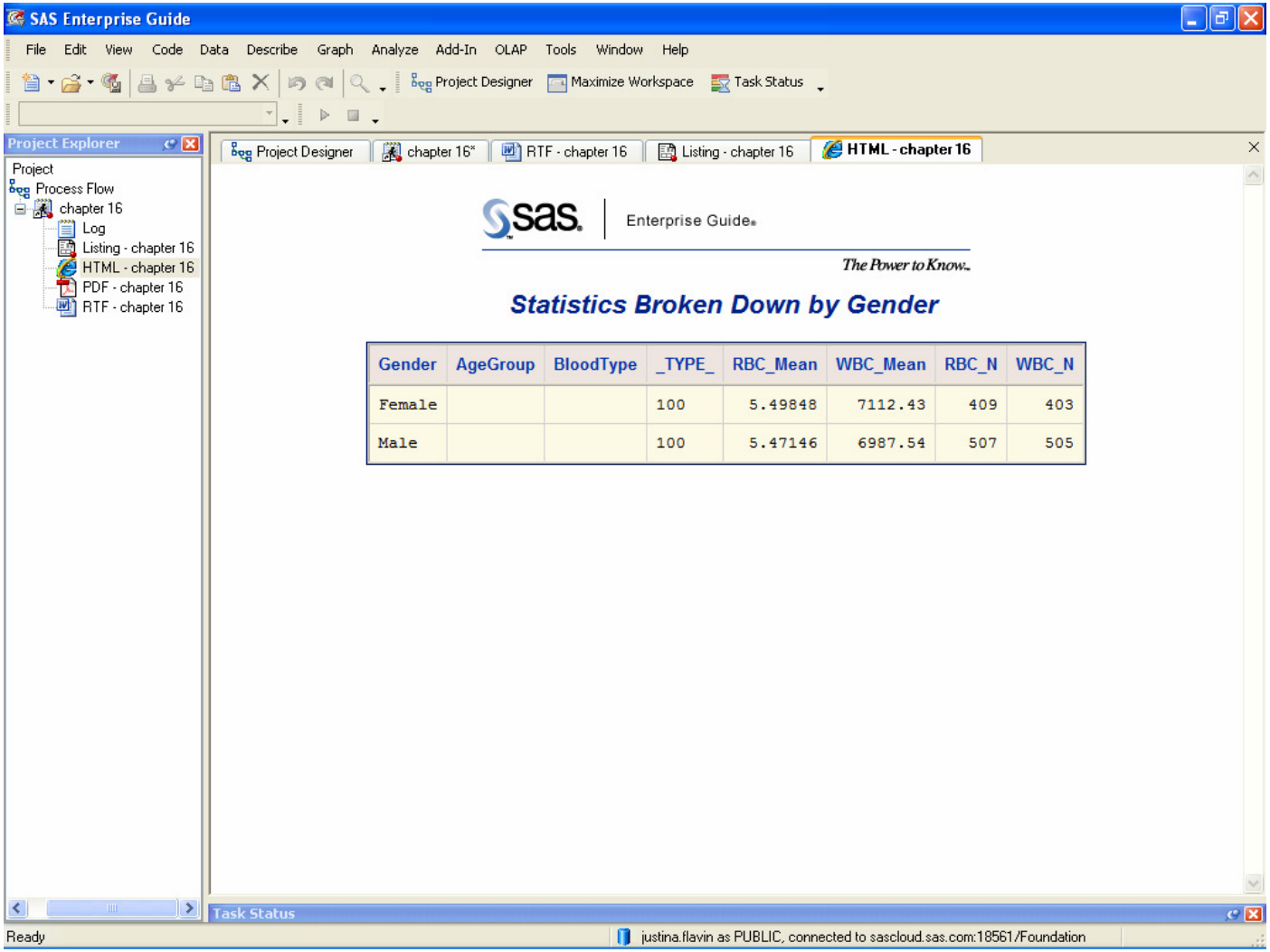
Task Status

Ready | justina.flavin as PUBLIC, connected to sascloud.sas.com:18561/Foundation

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

The _TYPE_ variable can be used to select results of interest

```
*Modified Program 16-14 Using the _TYPE_ variable to select cell means - page 336;
title "Statistics Broken Down by Gender";
proc print data=summary(drop = _freq_) noobs;
   where _TYPE_ = '100';
run;
```



Gender	AgeGroup	BloodType	_TYPE_	RBC_Mean	WBC_Mean	RBC_N	WBC_N
Female			100	5.49848	7112.43	409	403
Male			100	5.47146	6987.54	507	505

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

This example illustrates using the _TYPE_ variable to subset the data in creating new data sets.

```
*Modified Program 16-15 Using a DATA step to create separate summary data sets - page 336;
data grand(drop = Gender AgeGroup bloodtype)
     by_gender(drop = AgeGroup bloodtype)
     by_age(drop = Gender bloodtype)
     cellmeans;
   set summary;
   drop _type_;
   rename _freq_ = Number;
   if _type_ = '000' then output grand;
   else if _type_ = '010' then output by_age;
   else if _type_ = '100' then output by_gender;
   else if _type_ = '111' then output cellmeans;
run;
```

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.

16.13 Selecting Different Statistics for Each Variable

An output data set does not have to contain every requested statistic for every variable.

To select the variables, place the variable names inside a set of parentheses behind the statistic keyword.

The example in this section uses colon wildcard notation.
The colon modifier ":" used with a character string can be used as shorthand notation to select variables based on pattern matching.
The ":" acts as a wildcard character.

So *name:* will select all variables that are prefixed with the characters specified in *name*

i.e**. response:** would select variables response1, response2,…,response_tot
i.e**. :cat** would select variables acat, bcat, ccat, totcat

Syntax:

```
proc means < noprint nway >;
  class variablename1 variablename2...variablename'n';
  var variablename1 variablename2...variablename'n';
  output out=datasetname
         statistics-keyword1 (variablename1 variablename2...variablename'n') =
         statistics-keyword2 (variablename1 variablename2...variablename'n') =
         statistics-keyword'n' (variablename1 variablename2...variablename'n') =
         / autoname;
run;
```

This is illustrated in Program 16-16 on page 337.

Copyright 2009, Justina Flavin. All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the copyright holder.