

Copyright © 2018, SAS Institute Inc., Cary, NC, USA

ISBN 978-1-63526-659-7 (Paperback)
ISBN 978-1-63526-656-6 (EPUB)
ISBN 978-1-63526-657-3 (MOBI)
ISBN 978-1-63526-658-0 (PDF)

All Rights Reserved. Produced in the United States of America.

For a hard copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

The scanning, uploading, and distribution of this book via the Internet or any other means without the permission of the publisher is illegal and punishable by law. Please purchase only authorized electronic editions and do not participate in or encourage electronic piracy of copyrighted materials. Your support of others’ rights is appreciated.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication, or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a), and DFAR 227.7202-4, and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government’s rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

July 2018

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open-source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to http://support.sas.com/thirdpartylicenses.
# Contents

List of Programs ........................................................................................................................................... xi  
Preface ........................................................................................................................................................ xix  
About This Book ......................................................................................................................................... xxi  
About the Author ...................................................................................................................................... xxiii  
Acknowledgments .................................................................................................................................... xxv  

**Part 1: Getting Started...................... 1**

**Chapter 1: What Is SAS? .............................. 3**  
1.1 Introduction ............................................................ 3  
1.2 Getting Data into SAS .................................................. 4  
1.3 A Sample SAS Program ............................................... 4  
1.4 SAS Names ........................................................................ 6  
1.5 SAS Data Sets and SAS Data Types ................................ 7  
1.6 The SAS Windowing Environment, SAS Enterprise Guide, and the SAS University Edition .... 7  
1.7 Problems ........................................................................... 8  

**Chapter 2: Writing Your First SAS Program ............... 9**  
2.1 A Simple Program to Read Raw Data and Produce a Report ..................................................... 9  
2.2 Enhancing the Program .................................................. 16  
2.3 More on Comment Statements ......................................... 18  
2.4 How SAS Works (a Look inside the “Black Box”) ............................................................... 18  
2.5 Problems ........................................................................... 21  

**Part 2: DATA Step Processing...................... 23**

**Chapter 3: Reading Raw Data from External Files .......... 25**  
3.1 Introduction .......................................................................... 25  
3.2 Reading Data Values Separated by Blanks .................................. 26  
3.3 Specifying Missing Values with List Input ................................ 27  
3.4 Reading Data Values Separated by Commas (CSV Files) ......................... 27  
3.5 Using an alternative Method to Specify an External File .................................................. 28  
3.6 Reading Data Values Separated by Delimiters Other Than Blanks or Commas ................ 29  
3.7 Placing Data Lines Directly in Your Program (the DATALINES Statement) ................. 30  
3.8 Specifying INFILE Options with the DATALINES Statement ................................. 30  
3.9 Reading Raw Data from Fixed Columns—Method 1: Column Input ......................... 31  
3.10 Reading Raw Data from Fixed Columns—Method 2: Formatted Input ....................... 32  
3.11 Using a FORMAT Statement in a DATA Step versus in a Procedure ......................... 36  
3.12 Using Informats with List Input ........................................ 36  
3.13 Supplying an INFORMAT Statement with List Input ............................................. 37  
3.14 Using List Input with Embedded Delimiters .................................................. 37  
3.15 Problems ........................................................................... 38
8.4 The Iterative DO Loop ................................................................. 106
8.5 Other Forms of an Iterative DO Loop ........................................ 110
8.6 DO WHILE and DO UNTIL Statements ................................. 112
8.7 A Caution When Using DO UNTIL Statements ...................... 115
8.8 LEAVE and CONTINUE Statements ........................................ 116
8.9 Problems ..................................................................................... 117

Chapter 9: Working with Dates ................................................. 121
9.1 Introduction ............................................................................... 121
9.2 How SAS Stores Dates ............................................................. 121
9.3 Reading Date Values from Text Data ....................................... 122
9.4 Computing the Number of Years between Two Dates ............ 124
9.5 Demonstrating a Date Constant ............................................... 125
9.6 Computing the Current Date .................................................... 126
9.7 Extracting the Day of the Week, Day of the Month, Month, and Year from a SAS Date ............................... 126
9.8 Creating a SAS Date from Month, Day, and Year Values ........ 127
9.9 Substituting the 15th of the Month when the Day Value Is Missing ..................................................... 128
9.10 Using Date Interval Functions .................................................. 129
9.11 Problems .................................................................................. 134

Chapter 10: Subsetting and Combining SAS Data Sets .............. 137
10.1 Introduction .............................................................................. 137
10.2 Subsetting a SAS Data Set ...................................................... 137
10.3 Creating More Than One Subset Data Set in One DATA Step ... 139
10.4 Adding Observations to a SAS Data Set .................................. 139
10.5 Interleaving Data Sets ............................................................. 142
10.6 Combining Detail and Summary Data ..................................... 143
10.7 Merging Two Data Sets .......................................................... 144
10.8 Omitting the BY Statement in a Merge ................................. 146
10.9 Controlling Observations in a Merged Data Set .................... 147
10.10 More Uses for IN= Variables ............................................... 149
10.11 When Does a DATA Step End? .............................................. 150
10.12 Merging Two Data Sets with Different BY Variable Names ... 151
10.13 Merging Two Data Sets with Different BY Variable Data Types ..................................................... 152
10.14 One-to-One, One-to-Many, and Many-to-Many Merges ......... 154
10.15 Updating a Master File from a Transaction File .................... 156
10.16 Problems .............................................................................. 158

Chapter 11: Working with Numeric Functions ......................... 161
11.1 Introduction ............................................................................ 161
11.2 Functions That Round and Truncate Numeric Values .......... 161
11.3 Functions That Work with Missing Values ........................... 163
11.4 Setting Character and Numeric Values to Missing .......... 163
11.5 Descriptive Statistics Functions ............................................. 164
11.6 Computing Sums within an Observation ......................... 166
11.7 Mathematical Functions ......................................................... 167
11.8 Computing Some Useful Constants ...................................... 168
11.9 Generating Random Numbers .............................................. 168
11.10 Special Functions ................................................................. 172
11.11 Functions That Return Values from Previous Observations .. 174
Chapter 12: Working with Character Functions ........................................ 181
12.1 Introduction .................................................................................. 181
12.2 Determining the Length of a Character Value .................................. 182
12.3 Changing the Case of Characters ..................................................... 182
12.4 Removing Characters from Strings ................................................. 183
12.5 Joining Two or More Strings Together ............................................ 184
12.6 Removing Leading or Trailing Blanks .............................................. 185
12.7 Using the COMPRESS Function to Remove Characters from a String ... 186
12.8 Searching for Characters .............................................................. 189
12.9 Searching for Individual Characters ................................................. 190
12.10 Searching for Words in a String ...................................................... 191
12.11 Searching for Character Classes .................................................... 192
12.12 Using the NOT Functions for Data Cleaning ................................. 193
12.13 Extracting Part of a String ............................................................ 194
12.14 Dividing Strings into Words .......................................................... 196
12.15 Performing a Fuzzy Match ............................................................ 197
12.16 Substituting Strings or Words ....................................................... 199
12.17 Problems .................................................................................... 201

Chapter 13: Working with Arrays ......................................................... 205
13.1 Introduction .................................................................................. 205
13.2 Setting Values of 999 to a SAS Missing Value for Several Numeric Variables ... 206
13.3 Setting Values of NA and ? to a Missing Character Value .................. 207
13.4 Converting All Character Values to Propercase ............................. 208
13.5 Using an Array to Create New Variables ....................................... 209
13.6 Changing the Array Bounds ......................................................... 210
13.7 Temporary Arrays ........................................................................ 210
13.8 Loading the Initial Values of a Temporary Array from a Raw Data File ... 211
13.9 Using a Multidimensional Array for Table Lookup ......................... 212
13.10 Problems .................................................................................. 215

Part 3: Presenting and Summarizing Your Data .............................. 217

Chapter 14: Displaying Your Data ....................................................... 219
14.1 Introduction .................................................................................. 219
14.2 The Basics .................................................................................. 219
14.3 Changing the Appearance of Your Listing ..................................... 220
14.4 Changing the Appearance of Values ............................................. 222
14.5 Controlling the Observations That Appear in Your Listing ............. 223
14.6 Adding Titles and Footnotes to Your Listing ................................ 224
14.7 Changing the Order of Your Listing ............................................. 226
14.8 Sorting by More Than One Variable ............................................ 228
14.9 Labeling Your Column Headings .................................................. 229
14.10 Adding Subtotals and Totals to Your Listing .................................. 230
14.11 Making Your Listing Easier to Read ............................................ 233
14.12 Adding the Number of Observations to Your Listing ................... 234
## Chapter 16: Summarizing Your Data

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>16.1 Introduction</td>
<td>261</td>
</tr>
<tr>
<td>16.2 PROC MEANS—Starting from the Beginning</td>
<td>261</td>
</tr>
<tr>
<td>16.3 Adding a BY Statement to PROC MEANS</td>
<td>263</td>
</tr>
<tr>
<td>16.4 Using a CLASS Statement with PROC MEANS</td>
<td>264</td>
</tr>
<tr>
<td>16.5 Applying a Format to a CLASS Variable</td>
<td>265</td>
</tr>
<tr>
<td>16.6 Deciding between a BY Statement and a CLASS Statement</td>
<td>266</td>
</tr>
<tr>
<td>16.7 Creating Summary Data Sets Using PROC MEANS</td>
<td>266</td>
</tr>
<tr>
<td>16.8 Outputting Other Descriptive Statistics with PROC MEANS</td>
<td>267</td>
</tr>
<tr>
<td>16.9 Asking SAS to Name the Variables in the Output Data Set</td>
<td>268</td>
</tr>
<tr>
<td>16.10 Outputting a Summary Data Set: Including a BY Statement</td>
<td>269</td>
</tr>
<tr>
<td>16.11 Outputting a Summary Data Set: Using a CLASS Statement</td>
<td>270</td>
</tr>
<tr>
<td>16.12 Using Two CLASS Variables with PROC MEANS</td>
<td>271</td>
</tr>
<tr>
<td>16.13 Selecting Different Statistics for Each Variable</td>
<td>276</td>
</tr>
<tr>
<td>16.14 Printing all Possible Combinations of Your Class Variables</td>
<td>276</td>
</tr>
<tr>
<td>16.15 Problems</td>
<td>278</td>
</tr>
</tbody>
</table>

## Chapter 17: Counting Frequencies

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>17.1 Introduction</td>
<td>281</td>
</tr>
<tr>
<td>17.2 Counting Frequencies</td>
<td>281</td>
</tr>
<tr>
<td>17.3 Selecting Variables for PROC FREQ</td>
<td>284</td>
</tr>
<tr>
<td>17.4 Using Formats to Label the Output</td>
<td>286</td>
</tr>
<tr>
<td>17.5 Using Formats to Group Values</td>
<td>288</td>
</tr>
<tr>
<td>17.6 Problems Grouping Values with PROC FREQ</td>
<td>289</td>
</tr>
<tr>
<td>17.7 Displaying Missing Values in the Frequency Table</td>
<td>290</td>
</tr>
<tr>
<td>17.8 Changing the Order of Values in PROC FREQ</td>
<td>292</td>
</tr>
<tr>
<td>17.9 Producing Two-Way Tables</td>
<td>294</td>
</tr>
</tbody>
</table>
Chapter 18: Creating Tabular Reports ................................................. 299
18.1 Introduction .................................................................................. 299
18.2 A Simple PROC TABULATE Table .................................................. 300
18.3 Describing the Three PROC TABULATE Operators .......................... 301
18.4 Using the Keyword ALL ................................................................. 303
18.5 Producing Descriptive Statistics ...................................................... 304
18.6 Combining CLASS and Analysis Variables in a Table ....................... 306
18.7 Customizing Your Table ................................................................. 307
18.8 Demonstrating a More Complex Table ............................................. 309
18.9 Computing Row and Column Percentages ........................................ 310
18.10 Displaying Percentages in a Two-Dimensional Table ....................... 312
18.11 Computing Column Percentages .................................................... 313
18.12 Computing Percentages on Numeric Variables ................................. 314
18.13 Understanding How Missing Values Affect PROC TABULATE Output ... 315
18.14 Problems ..................................................................................... 318

Chapter 19: Introducing the Output Delivery System .......................... 325
19.1 Introduction .................................................................................. 325
19.2 Sending SAS Output to an HTML File ............................................ 325
19.3 Creating a Table of Contents .......................................................... 327
19.4 Selecting a Different HTML Style .................................................... 328
19.5 Choosing Other ODS Destinations ................................................... 329
19.6 Selecting or Excluding Portions of SAS Output ................................. 330
19.7 Sending Output to a SAS Data Set .................................................... 331
19.8 Problems ..................................................................................... 335

Chapter 20: Creating Charts and Graphs .............................................. 337
20.1 Introduction .................................................................................. 337
20.2 Creating Bar Charts ....................................................................... 337
20.3 Displaying Statistics for a Response Variable .................................... 340
20.4 Creating Scatter Plots ..................................................................... 341
20.5 Adding a Regression Line and Confidence Limits to the Plot ............. 342
20.6 Generating Time Series Plots .......................................................... 343
20.7 Describing Two Methods of Generating Smooth Curves .................... 345
20.8 Generating Histograms .................................................................... 346
20.9 Generating a Simple Box Plot .......................................................... 347
20.10 Producing a Box Plot with a Grouping Variable ............................... 348
20.11 Demonstrating Overlays and Transparency ..................................... 349
20.12 Problems ..................................................................................... 351

Part 4: Advanced Topics .................................................................... 357

Chapter 21: Using Advanced INPUT Techniques ................................... 359
21.1 Introduction .................................................................................. 359
21.2 Handling Missing Values at the End of a Line .................................. 360
21.3 Reading Short Data Lines ............................................................... 362
Chapter 22: Using Advanced Features of User-Defined Formats and Informats ........................................ 377
22.1 Introduction ............................................................................................................. 377
22.2 Using Formats to Recode Variables ........................................................................ 377
22.3 Using Formats with a PUT Function to Create New Variables ............................... 379
22.4 Creating User-Defined Informats ............................................................................ 380
22.5 Reading Character and Numeric Data in One Step .................................................... 382
22.6 Using Formats (and Informats) to Perform Table Lookup .......................................... 384
22.7 Using a SAS Data Set to Create a Format .................................................................. 386
22.8 Updating and Maintaining Your Formats ................................................................. 390
22.9 Using Formats within Formats .................................................................................. 392
22.10 Multilabel Formats .................................................................................................. 394
22.11 Using the INPUTN Function to Perform a More Complicated Table Lookup .......... 398
22.12 Problems ................................................................................................................ 399

Chapter 23: Restructuring SAS Data Sets ........................................................................ 403
23.1 Introduction ............................................................................................................. 403
23.2 Converting a Data Set with One Observation per Subject to a Data Set with Several Observations per Subject: Using a DATA Step ................................................................. 403
23.3 Converting a Data Set with Several Observations per Subject to a Data Set with One Observation per Subject: Using a DATA Step ................................................................. 406
23.4 Converting a Data Set with One Observation per Subject to a Data Set with Several Observations per Subject: Using PROC TRANSPOSE ................................................................. 407
23.5 Converting a Data Set with Several Observations per Subject to a Data Set with One Observation per Subject: Using PROC TRANSPOSE ................................................................. 408
23.6 Problems .................................................................................................................. 409

Chapter 24: Working with Multiple Observations per Subject ..................................... 413
24.1 Introduction ............................................................................................................. 413
24.2 Identifying the First or Last Observation in a Group ................................................ 413
24.3 Counting the Number of Visits Using PROC FREQ ................................................ 416
24.4 Computing Differences between Observations ....................................................... 418
24.5 Computing Differences between the First and Last Observation in a BY Group Using the LAG Function .................................................................................................................. 420
24.6 Computing Differences between the First and Last Observation in a BY Group Using a RETAIN Statement ................................................................................................................. 421
Preface to the Second Edition

It’s been almost 11 years since the first edition of this book was published, and I felt that it was time to bring it up-to-date. One major change is that all the SAS output is now displayed as HTML images, using the default style HTMLBlue. Not only does it look nicer than the old monospaced listing output, but the book figures are now images that can be viewed and sized on e-devices.

While we are on the topic of major changes, I should mention that a new chapter on Perl regular expressions was added, and the previous SAS/GRAPH chapter was replaced by a chapter on PROC SGPLOT. This procedure is easier to use than SAS/GRAPH, and (drum roll please) it is included with Base SAS.

People have come up to me at SAS conferences and told me that they learned SAS programming from the first edition of this book. That is really nice to hear. I hope that this edition will do the same for a whole new generation of programmers.

Ron Cody

Summer, 2018
About This Book

What Does This Book Cover?
This book teaches SAS programming from very basic concepts to more advanced topics. Because many programmers prefer examples over reference-type syntax, this book uses short examples to explain each topic. The second edition has brought this classic book about SAS programming up to the latest SAS version. There are new chapters that cover topics such as PROC SGPLOT (replacing the older chapter about SAS/GRAPH) and a completely new chapter about Perl regular expressions. This is a book that belongs on the shelf (or e-book reader) of every person who programs in SAS.

Is This Book for You?
This book has been used by people with no programming experience who want to learn SAS as well as intermediate and even advanced SAS programmers who want to learn new techniques or see new ways to accomplish existing tasks.

What Are the Prerequisites for This Book?
There are no prerequisites for this book. It is for EVERYONE.

What’s New in This Edition?
A new chapter about Perl regular expressions was added, and an old chapter about SAS/GRAPH was replaced by one describing PROC SGPLOT. This procedure can re-create all the output that was previously created by SAS/GRAPH but in a much simpler manner. All the programs in the second edition were examined to determine whether there was a newer, better way to accomplish the task. Finally, all the output is shown in the default HTML style.

What Should You Know about the Examples?
You can download every program and data set that is used in this book so that you can try the programs on your own—a valuable learning experience.

Software Used to Develop the Book’s Content
The only SAS software that you need is Base SAS or the SAS University Edition. Because the latter is available to anyone as a free download, anyone can learn how to program using SAS even if he or she does not currently have access to SAS.
Example Code and Data
You can access the example code and data for this book by linking to its author page at support.sas.com/cody. If you are using the SAS University edition, you must copy the programs and data files to one of your shared folders. An example is C:\SASUniversityEdition\Myfolders.

SAS University Edition
This book is compatible with SAS University Edition.

Where Are the Exercise Solutions?
Solutions to the odd-numbered problems are printed at the back of the book and are also included in the free download link on the author page. Professors can obtain copies of the solutions to the even-numbered problems, and self-learners can also request these solutions.

We Want to Hear from You
SAS Press books are written by SAS Users for SAS Users. We welcome your participation in their development and your feedback on SAS Press books that you are using. Please visit sas.com/books to do the following:

- Sign up to review a book.
- Recommend a topic.
- Request information about how to become a SAS Press author.
- Provide feedback on a book.

Do you have questions about a SAS Press book that you are reading? Contact the author through saspress@sas.com or https://support.sas.com/author_feedback.

SAS has many resources to help you find answers and expand your knowledge. If you need additional help, see our list of resources: sas.com/books.
About the Author

Ron Cody, EdD, is a retired professor from the Rutgers Robert Wood Johnson Medical School who now works as a national instructor for SAS and as an author of books on SAS and statistics. A SAS user since 1977, Ron's extensive knowledge and innovative style have made him a popular presenter at local, regional, and national SAS conferences. He has authored or co-authored numerous books, as well as countless articles in medical and scientific journals.

Learn more about this author by visiting his author page at http://support.sas.com/cody. There you can download free book excerpts, access example code and data, read the latest reviews, get updates, and more.
Chapter 16: Summarizing Your Data

16.1 Introduction

You may have thought of PROC MEANS (or PROC SUMMARY) primarily as a way to generate summary reports, reporting the sums and means of your numeric variables. However, these procedures are much more versatile and can be used to create summary data sets that can then be analyzed with more DATA or PROC steps.

All the examples in this chapter use PROC MEANS rather than PROC SUMMARY, even when all you want is an output data set. The reason for this is that using PROC MEANS with a NOPRINT option is identical to using PROC SUMMARY when you are creating an output data set.

16.2 PROC MEANS—Starting from the Beginning

You can begin by running PROC MEANS with all the defaults, using the permanent SAS data set Blood.
Here it is:

**Program 16.1: PROC MEANS with All the Defaults**

```sas
title "PROC MEANS With All the Defaults";
proc means data=Learn.Blood;
run;
```

Here is the resulting output:

**Figure 16.1: Output from Program 16.1**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Label</th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject</td>
<td>WBC</td>
<td>908</td>
<td>7042.97</td>
<td>1003.37</td>
<td>4070.00</td>
<td>10550.00</td>
</tr>
<tr>
<td>RBC</td>
<td></td>
<td>916</td>
<td>5.4353262</td>
<td>0.9841158</td>
<td>1.710000</td>
<td>8.750000</td>
</tr>
<tr>
<td>Chol</td>
<td>Cholesterol</td>
<td>795</td>
<td>201.4352201</td>
<td>49.8867157</td>
<td>17.000000</td>
<td>331.000000</td>
</tr>
</tbody>
</table>

By default, PROC MEANS produces statistics on all the numeric variables in the input SAS data set. Looking at the output, you see the default statistics produced are N (number of nonmissing values), Mean (average), Std Dev (standard deviation), Minimum, and Maximum. The next step is to gain some control over this process.

You can control which variables to include in the report by supplying a VAR statement. Statistics are chosen by selecting options in the PROC MEANS statement. Here is a partial list of some of the more commonly used options.

<table>
<thead>
<tr>
<th>PROC MEANS Option</th>
<th>Statistic Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Number of nonmissing values</td>
</tr>
<tr>
<td>NMISS</td>
<td>Number of missing values</td>
</tr>
<tr>
<td>MEAN</td>
<td>Mean or Average</td>
</tr>
<tr>
<td>CLM</td>
<td>Confidence Limit for the Mean</td>
</tr>
<tr>
<td>SUM</td>
<td>Sum of the values</td>
</tr>
<tr>
<td>MIN</td>
<td>Minimum (nonmissing) value</td>
</tr>
<tr>
<td>MAX</td>
<td>Maximum value</td>
</tr>
<tr>
<td>MEDIAN</td>
<td>Median value</td>
</tr>
<tr>
<td>STD</td>
<td>Standard deviation</td>
</tr>
<tr>
<td>VAR</td>
<td>Variance</td>
</tr>
<tr>
<td>CLM</td>
<td>95% confidence interval for the mean</td>
</tr>
<tr>
<td>Q1</td>
<td>Value of the first quartile (25th percentile)</td>
</tr>
<tr>
<td>Q3</td>
<td>Value of the third quartile (75th percentile)</td>
</tr>
<tr>
<td>QRANGE</td>
<td>Interquartile range (equal to Q3–Q1)</td>
</tr>
</tbody>
</table>

Besides these statistics, the option MAXDEC=value is especially useful. This value controls the number of places to the right of the decimal point that are printed in the output.
Let’s use some of these options to customize the output. You will compute the number of missing and nonmissing values, the mean, median, minimum, and maximum for the variables RBC (red blood cells) and WBC (white blood cells) in the Blood data set. Finally, you will report all statistics to the nearest 10th. Here is the program:

**Program 16.2: Adding a VAR Statement and Requesting Specific Statistics with PROC MEANS**

```sas
title "Selected Statistics Using PROC MEANS";
proc means data=Learn.Blood n nmiss mean median
   min max maxdec=1;
   var RBC WBC;
run;
```

The output follows:

**Figure 16.2: Output from Program 16.2**

<table>
<thead>
<tr>
<th>Variable</th>
<th>N</th>
<th>N Miss</th>
<th>Mean</th>
<th>Median</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBC</td>
<td>916</td>
<td>84</td>
<td>5.5</td>
<td>5.5</td>
<td>1.7</td>
<td>8.8</td>
</tr>
<tr>
<td>WBC</td>
<td>908</td>
<td>92</td>
<td>7043.0</td>
<td>7040.0</td>
<td>4070.0</td>
<td>10550.0</td>
</tr>
</tbody>
</table>

You now have only the statistics you requested on the variables listed in the VAR list. Notice also that all the statistics are reported to one decimal place, due to the `MAXDEC=1` option. (Unfortunately, the number of decimal places you choose is applied to all variables. If you need more control over the number of printed decimal places for each variable, you can use PROC TABULATE.) Values for N and NMISS are always integers.

### 16.3 Adding a BY Statement to PROC MEANS

If you want to see descriptive statistics for each level of another variable, you can include a BY statement, listing one or more BY variables. Remember that you have to sort your data set first by the same variable or variables you list in the BY statement. Here are the same statistics displayed by Figure 16.2, broken down by gender:

**Program 16.3: Adding a BY Statement to PROC MEANS**

```sas
proc sort data=Learn.Blood out=Blood;
   by Gender;
run;

title "Adding a BY Statement to PROC MEANS";
proc means data=Blood n nmiss mean median
   min max maxdec=1;
   by Gender;
   var RBC WBC;
run;
```
You now have your descriptive statistics for males and females separately, as shown next:

Figure 16.3: Output from Program 16.3

<table>
<thead>
<tr>
<th>Variable</th>
<th>N</th>
<th>N Miss</th>
<th>Mean</th>
<th>Median</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBC</td>
<td>409</td>
<td>31</td>
<td>5.5</td>
<td>5.5</td>
<td>1.7</td>
<td>8.8</td>
</tr>
<tr>
<td>WBC</td>
<td>403</td>
<td>37</td>
<td>7112.4</td>
<td>7150.0</td>
<td>4620.0</td>
<td>10260.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>N</th>
<th>N Miss</th>
<th>Mean</th>
<th>Median</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBC</td>
<td>507</td>
<td>53</td>
<td>6.5</td>
<td>5.5</td>
<td>2.3</td>
<td>8.4</td>
</tr>
<tr>
<td>WBC</td>
<td>505</td>
<td>55</td>
<td>6987.5</td>
<td>6930.0</td>
<td>4070.0</td>
<td>10550.0</td>
</tr>
</tbody>
</table>

16.4 Using a CLASS Statement with PROC MEANS

PROC MEANS lets you use a CLASS statement in place of a BY statement. The CLASS statement performs a similar function to the BY statement, with some significant differences. If you are using PROC MEANS to print a report and are not creating a summary output data set, the differences in the printed output between a BY and CLASS statement are basically cosmetic. The main difference, from a programmer’s perspective, is that you do not have to sort your data set before using a CLASS statement. To demonstrate this, run Program 16.3 again, substituting the CLASS statement for the BY statement, as follows:

Program 16.4: Using a CLASS Statement with PROC MEANS
```
title "Using a CLASS Statement with PROC MEANS";
proc means data=Learn.Blood n nmiss mean median
   min max maxdec=1;
   class Gender;
   var RBC WBC;
run;
```

Notice that you are using the permanent SAS data set (that may not be sorted) instead of the temporary sorted data set used in Program 16.3. There are some minor differences in the appearance of the printed output, as shown here:
16.5 Applying a Format to a CLASS Variable

One very nice feature of using a CLASS statement (besides not having to sort your data) is that SAS uses formatted values of the CLASS variable(s). You can use this to your advantage by adding a FORMAT statement to the procedure and changing how the CLASS variable groups your data, all without having to modify your original data set.

Suppose you want to see some basic statistics (mean and median) for the two blood count variables (RBC and WBC), broken down by subjects with cholesterol levels below 200 versus subjects with cholesterol levels at or above 200. See how easy this is to do with a CLASS statement and a FORMAT statement:

Program 16.5: Demonstrating the Effect of a Formatted CLASS Variable

```sas
proc format;
  value Chol_Group
    low <= 200 = 'Low'
    200 - high = 'High';
run;

title "Using a CLASS Statement with PROC MEANS";
proc means data=Learn.Blood n nmiss mean median maxdec=1;
  class Chol;
  format Chol Chol_Group.;
  var RBC WBC;
run;
```

Do you see the tremendous power in this method? You can try different groupings of CLASS variables by supplying a new format. Here is the output:
Even though the variable Chol (Cholesterol) is a continuous value, the output displays statistics for the formatted values of the variable.

**Note:** The technique of using formats with CLASS variables is very powerful and efficient. You can create summary reports for different groupings of a continuous variable without having to make modifications to your original data set.

### 16.6 Deciding between a BY Statement and a CLASS Statement

There are a number of considerations in deciding whether to use a BY or a CLASS statement when you want to use PROC MEANS to produce printed output (the differences are more important if you are using PROC MEANS to create summary output data sets).

First, if you have a very large data set that is not sorted, you may want to use a CLASS statement. However, if the data set is already in the correct sorted order, you can use either a CLASS or a BY statement. If you have a large number of CLASS variables and there are many distinct values for each of these variables, you may need considerable computer memory to run the procedure. If you have relatively small data sets, choose the statement that produces the style of printed output that you prefer.

### 16.7 Creating Summary Data Sets Using PROC MEANS

You can use PROC MEANS (or SUMMARY) to create a new data set that contains summary information such as sums and means. This data set can then be used for further analysis. This first example shows how to compute two means and output them to a SAS data set.

**Program 16.6: Creating a Summary Data Set Using PROC MEANS**

```sas
proc means data=Learn.Blood noprint;
   var RBC WBC;
   output out = My_Summary
      mean = MeanRBC MeanWBC;
run;

title "Listing of My_Summary";
proc print data=My_Summary noobs;
run;
```
An OUTPUT statement tells PROC MEANS that you want to create a summary SAS data set. The keyword OUT= is used to name the new data set. In this example, you are going to name your output data set My_Summary. Next, you can specify exactly what statistics you want in this data set. You can output most of the statistics listed earlier in this chapter. Following the keyword and an equal sign, you provide a list of the variable names that correspond to these values. In this program, the variable MeanRBC is the mean of all the RBC values, and MeanWBC is the mean of all the WBC values. You can name these variables anything you want—they are arbitrary. The order of the variables here corresponds to the order of the variables in the VAR list. It is convenient to name these variables in a way that helps you remember what they stand for. Shortly, you will see that SAS can name these variables for you in a very convenient way. Finally, if you want to create a SAS data set but do not want any printed output from PROC MEANS, use the NOPRINT option. As we mentioned earlier, if you choose to use PROC SUMMARY instead of PROC MEANS, you do not need a NOPRINT option—it is the default for that procedure.

Program 16.6 also includes a PROC PRINT statement so that you can see the contents of the output data set. Here is the listing of the summary data set My_Summary:

**Figure 16.6: Output from Program 16.6**

<table>
<thead>
<tr>
<th>Listing of My_Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>TYPE</em></td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

This data set has only one observation. It includes the two means plus two additional variables created by PROC MEANS. We will discuss these two variables later when we run the procedure with a CLASS statement.

When you are outputting only a single statistic (such as a mean in the previous example), you can omit the variable list following the keyword MEAN= if you want. If you do, SAS gives these summary variables the same names as the variables listed in the VAR statement. **It is strongly recommended that you do not do this.** It is easy to make a mistake when you have the same variable name in two data sets, with one representing individual values and the other representing a summary statistic.

### 16.8 Outputting Other Descriptive Statistics with PROC MEANS

As we mentioned earlier, you can output more than one statistic in your output data set. As an example, let’s output the mean, the number of nonmissing observations, the number of missing observations, and the median. Here is the program:
Program 16.7: Outputting More Than One Statistic with PROC MEANS

```sas
proc means data=Learn.Blood noprint;
   var RBC WBC;
   output out = Many_Stats
      mean    = M_RBC M_WBC
      n       = N_RBC N_WBC
      nmiss   = Miss_RBC Miss_WBC
      median  = Med_RBC Med_WBC;
run;
```

So, you get the idea. You can output as many statistics as you want and name them anything you want. A listing of this data set, using PROC PRINT, is shown here:

Figure 16.7: Listing of Data Set Many_Stats

<table>
<thead>
<tr>
<th><em>TYPE</em></th>
<th><em>FREQ</em></th>
<th>M_RBC</th>
<th>M_WBC</th>
<th>N_RBC</th>
<th>N_WBC</th>
<th>Miss_RBC</th>
<th>Miss_WBC</th>
<th>Med_RBC</th>
<th>Med_WBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1000</td>
<td>5.49353</td>
<td>7042.97</td>
<td>915</td>
<td>908</td>
<td>84</td>
<td>92</td>
<td>5.62</td>
<td>7040</td>
</tr>
</tbody>
</table>

The two variables M_RBC and M_WBC represent the means of RBC and WBC, respectively. The two variables Med_RBC and Med_WBC are the medians of the two variables. N_RBC and N_WBC represent the number of nonmissing values; Miss_RBC and Miss_WBC represent the number of missing values for these two variables. As a check, notice that N_RBC plus Miss_RBC equals 1,000, the total number of observations in data set Blood.

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

You can ask SAS to create variable names for any of the summary statistics produced by PROC MEANS. The OUTPUT option AUTONAME causes PROC MEANS to create variable names for the selected statistics by using the variable names in the VAR statement and adding an underscore character followed by the name of the statistic. The best way to see this is to look at a program and a listing of the output data set:

Program 16.8: Demonstrating the OUTPUT Option AUTONAME

```sas
proc means data=Learn.Blood noprint;
   var RBC WBC;
   output out = Many_Stats
      mean    =
      n       =
      nmiss   =
      median  = / autoname;
run;
```

Because AUTONAME is a statement option, it follows a slash in the OUTPUT statement. Take a look at a listing of this data set to see how SAS named these variables:
The OUTPUT option AUTONAME is quite useful, and it creates consistent and easy-to-understand variable names. Be aware that the suffix used for standard deviation is StdDev and not the abbreviation Std that is commonly used. Also remember that SAS variable names cannot be longer than 32 characters—this includes the suffix that PROC MEANS adds. Although it is unlikely, if adding the suffix causes the variable name to exceed 32 characters, the original variable name will be truncated.

### 16.10 Outputting a Summary Data Set: Including a BY Statement

Although there are many uses for a summary data set containing summary statistics on an entire data set, there are times when you would like to output summary statistics for each level of one or more classification variables. For example, you might want to see n's and means of RBC and WBC broken down by gender. Remember that to use a BY statement, the data set must be sorted in the same order. Program 16.9 creates a summary data set containing the mean values of RBC and WBC for males and females.

**Program 16.9: Adding a BY Statement to PROC MEANS**

```sas
proc sort data=Learn.Blood out=Blood;
  by Gender;
run;

proc means data=Blood noprint;
  by Gender;
  var RBC WBC;
  output out = By_Gender
    mean = / autoname;
run;
```

A listing of the output data set follows:

**Figure 16.9: Listing of Data Set By_Gender**

<table>
<thead>
<tr>
<th>Gender</th>
<th><em>TYPE</em></th>
<th><em>FREQ</em></th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>0</td>
<td>440</td>
<td>5.49848</td>
<td>7112.43</td>
<td>409</td>
<td>403</td>
</tr>
<tr>
<td>Male</td>
<td>0</td>
<td>560</td>
<td>5.47146</td>
<td>6987.54</td>
<td>507</td>
<td>505</td>
</tr>
</tbody>
</table>

In this data set, _FREQ_ represents the number of observations for each value of gender (it doesn’t matter if there are missing values for RBC or WBC). For example, you see that there are 440 females in the data...
set and that the means for RBC and WBC are 5.49848 and 7112.43, respectively. The variables RBC_N and WBC_N represent the number of nonmissing values that were used to compute the two means.

### 16.11 Outputting a Summary Data Set: Using a CLASS Statement

As you saw earlier in this chapter, you can use a CLASS statement with PROC MEANS to obtain results similar to those obtained using a BY statement. The difference between the two becomes more apparent when you are creating output data sets. The program here is a repeat of Program 16.9, with a CLASS statement replacing the BY statement (and the PROC SORT omitted):

**Program 16.10: Adding a CLASS Statement to PROC MEANS**

```sas
proc means data=Learn.Blood noprint;
   class Gender;
   var RBC WBC;
   output out = With_Class
      mean =
         n    = / autoname;
run;
```

The resulting data set is listed here:

**Figure 16.10: Listing of Data Set With_Class**

<table>
<thead>
<tr>
<th>Gender</th>
<th><em>TYPE</em></th>
<th><em>FREQ</em></th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>1</td>
<td>440</td>
<td>5.49848</td>
<td>7112.43</td>
<td>409</td>
<td>403</td>
</tr>
<tr>
<td>Male</td>
<td>1</td>
<td>560</td>
<td>5.47146</td>
<td>6987.54</td>
<td>507</td>
<td>505</td>
</tr>
</tbody>
</table>

Notice that you now have three observations instead of just two. The first observation in this data set, with _TYPE_ equal to 0, is the mean for both males and females. Statisticians call this the *grand mean*. The other two observations with _TYPE_ equal to 1 represent the means for females and males. We will discuss the interpretation of the _TYPE_ variable in the next section, which demonstrates the use of more than one CLASS variable.

If you only want the means broken down by gender and do not want an observation with the grand mean in your output data set, use the NWAY option of PROC MEANS. Using this option makes the output data set using a CLASS statement identical (except for the value of _TYPE_) to an output data set using a BY statement. Program 16.11 demonstrates the effect of the NWAY option:
Program 16.11: Adding the NWAY Option to PROC MEANS

```
proc means data=Learn.Blood noprint nway;
  class Gender;
  var RBC WBC;
  output out = With_Class
    mean =
    n    = / autoname;
run;
```

The resulting data set is shown below:

**Figure 16.11: Listing of Data Set With_Class**

<table>
<thead>
<tr>
<th>Gender</th>
<th><em>TYPE</em></th>
<th><em>FREQ</em></th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>1</td>
<td>440</td>
<td>5.49848</td>
<td>7112.43</td>
<td>409</td>
<td>403</td>
</tr>
<tr>
<td>Male</td>
<td>1</td>
<td>560</td>
<td>5.47146</td>
<td>6987.54</td>
<td>507</td>
<td>505</td>
</tr>
</tbody>
</table>

Notice that the extra observation containing the grand mean is no longer present.

**Note:** It is very important to remember to use the NWAY option if you only want to see your descriptive statistics at each level of the CLASS variable.

16.12 Using Two CLASS Variables with PROC MEANS

Things start to get more complicated when you have more than one CLASS variable. Suppose you want to compute the mean and the number of nonmissing values for RBC and WBC for each combination of AgeGroup and Gender. You place these two variables in the CLASS statement and create an output data set just as you did earlier with only one CLASS variable. Here are the PROC statements to create such an output data set:

Program 16.12: Using Two CLASS Variables with PROC MEANS

```
proc means data=Learn.Blood noprint;
  class Gender AgeGroup;
  var RBC WBC;
  output out = Summary
    mean =
    n    = / autoname;
run;
```
And here is a listing of the output data set:

Figure 16.12: Listing of Data Set Summary

<table>
<thead>
<tr>
<th>Gender</th>
<th>AgeGroup</th>
<th><em>TYPE</em></th>
<th><em>FREQ</em></th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>1000</td>
<td>5.48353</td>
<td>7042.97</td>
<td>916</td>
<td>908</td>
</tr>
<tr>
<td>Old</td>
<td></td>
<td>1</td>
<td>598</td>
<td>5.45779</td>
<td>7011.56</td>
<td>551</td>
<td>540</td>
</tr>
<tr>
<td>Young</td>
<td></td>
<td>1</td>
<td>402</td>
<td>5.52238</td>
<td>7089.08</td>
<td>365</td>
<td>368</td>
</tr>
<tr>
<td>Female</td>
<td></td>
<td>2</td>
<td>440</td>
<td>5.49848</td>
<td>7112.43</td>
<td>409</td>
<td>403</td>
</tr>
<tr>
<td>Male</td>
<td></td>
<td>2</td>
<td>560</td>
<td>5.47146</td>
<td>6987.54</td>
<td>507</td>
<td>505</td>
</tr>
<tr>
<td>Female</td>
<td>Old</td>
<td>3</td>
<td>258</td>
<td>5.47921</td>
<td>7105.98</td>
<td>242</td>
<td>234</td>
</tr>
<tr>
<td>Female</td>
<td>Young</td>
<td>3</td>
<td>182</td>
<td>5.52541</td>
<td>7121.36</td>
<td>167</td>
<td>169</td>
</tr>
<tr>
<td>Male</td>
<td>Old</td>
<td>3</td>
<td>340</td>
<td>5.44100</td>
<td>6939.35</td>
<td>309</td>
<td>306</td>
</tr>
<tr>
<td>Male</td>
<td>Young</td>
<td>3</td>
<td>220</td>
<td>5.51899</td>
<td>7061.66</td>
<td>198</td>
<td>199</td>
</tr>
</tbody>
</table>

Notice that things are getting more complicated. Even though there are only two levels of Gender and two levels of AgeGroup, there are nine observations in the output data set with four values of _TYPE_. The _TYPE_ = 0 observation is the same as you saw earlier—it represents the grand mean, the mean of all nonmissing values in the data set. To understand the other observations in this data set, you have to know either how to count in binary or how to use another PROC MEANS option—CHARTYPE. Let’s take the easy way out and rerun this program with the CHARTYPE procedure option as follows:

Program 16.13: Adding the CHARTYPE Procedure Option to PROC MEANS

```sas
proc means data=Learn.Blood noprint chartype;
  class Gender AgeGroup;
  var RBC WBC;
  output out = Summary
      mean =
      n   = / autoname;
run;
```
A listing of the output now looks like this:

**Figure 16.13: Listing of Data Set Summary (with CHARTYPE Procedure Option)**

<table>
<thead>
<tr>
<th>Gender</th>
<th>AgeGroup</th>
<th><em>TYPE</em></th>
<th><em>FREQ</em></th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1000</td>
<td>5.48353</td>
<td>7042.97</td>
<td>916</td>
<td>908</td>
</tr>
<tr>
<td></td>
<td>Old</td>
<td>0</td>
<td>598</td>
<td>5.45779</td>
<td>7011.56</td>
<td>551</td>
<td>540</td>
</tr>
<tr>
<td></td>
<td>Young</td>
<td>0</td>
<td>402</td>
<td>5.52238</td>
<td>7089.08</td>
<td>365</td>
<td>368</td>
</tr>
<tr>
<td>Female</td>
<td>1</td>
<td>0</td>
<td>440</td>
<td>5.48848</td>
<td>7112.43</td>
<td>409</td>
<td>403</td>
</tr>
<tr>
<td>Male</td>
<td>1</td>
<td>0</td>
<td>560</td>
<td>5.47146</td>
<td>6987.54</td>
<td>507</td>
<td>505</td>
</tr>
<tr>
<td>Female</td>
<td>Old</td>
<td>1</td>
<td>258</td>
<td>5.47921</td>
<td>7105.98</td>
<td>242</td>
<td>234</td>
</tr>
<tr>
<td>Female</td>
<td>Young</td>
<td>1</td>
<td>182</td>
<td>5.52641</td>
<td>7121.36</td>
<td>167</td>
<td>169</td>
</tr>
<tr>
<td>Male</td>
<td>Old</td>
<td>1</td>
<td>340</td>
<td>5.44100</td>
<td>6939.35</td>
<td>309</td>
<td>306</td>
</tr>
<tr>
<td>Male</td>
<td>Young</td>
<td>1</td>
<td>220</td>
<td>5.51899</td>
<td>7061.66</td>
<td>198</td>
<td>199</td>
</tr>
</tbody>
</table>

This may not look any simpler, but it is now easier to explain what the variable _TYPE_ represents. First of all, the option CHARTYPE (this stands for character type) causes the _TYPE_ variable to be a character string of 1’s and 0’s. If you are familiar with counting in binary, you will observe that the character strings under the _TYPE_ column represent the previous _TYPE_ values, except they are now in a binary representation. (It’s OK if this last sentence is meaningless to you—that’s why SAS created the CHARTYPE option in the first place.)

If you look at the CLASS statement, you see the two variables Gender and AgeGroup. If you imagine writing the left-most character of _TYPE_ under Gender and the other character of _TYPE_ under AgeGroup, there is a simple rule for interpreting the summary statistic for any value of _TYPE_. The following table displays the CLASS variables with the values of _TYPE_ written underneath (we will use the mean as an example):

<table>
<thead>
<tr>
<th>Gender</th>
<th>AgeGroup</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>TYPE</em></td>
<td>0</td>
<td>Mean for all Genders and all AgeGroups</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Mean for each level of AgeGroup</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Mean for each level of Gender</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Mean for each combination of Gender and AgeGroup</td>
</tr>
</tbody>
</table>

Here is the rule: if there is a 1 under the CLASS variable, the statistics in the output data set represent the statistics computed on each level of that variable. For example, if you are computing means, the mean with a value of _TYPE_ equal to 10 is a mean computed at each level of Gender (for all age groups). If you only want the statistics broken down by each level of the CLASS variables, you can use the NWAY option. This selects only the largest value of _TYPE_. (If you use the CHARTYPE option, NWAY selects the value of _TYPE_ containing all 1’s.)
You can use the _TYPE_ variable to select the appropriate breakdown. For example, you can use the _TYPE_ values in a WHERE statement in a PROC PRINT statement, like this:

**Program 16.14: Using the _TYPE_ Variable to Select CLASS Variable Breakdowns**

```sas
title "Statistics Broken Down by Gender";
proc print data=Summary(drop = _freq_) noobs;
   where _TYPE_ = '10';
run;
```

**Figure 16.14: Output from Program 16.14**

<table>
<thead>
<tr>
<th>Gender</th>
<th>AgeGroup</th>
<th><em>TYPE</em></th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td></td>
<td>10</td>
<td>5.49848</td>
<td>7112.43</td>
<td>409</td>
<td>403</td>
</tr>
<tr>
<td>Male</td>
<td></td>
<td>10</td>
<td>5.47146</td>
<td>6987.54</td>
<td>507</td>
<td>505</td>
</tr>
</tbody>
</table>

Because the values are broken down by Gender, AgeGroup has a missing value in each observation (it was left in the listing for demonstration purposes).

You can also use this summary data set to create separate data sets: one for the grand mean, one for means broken down by Gender, one broken down by AgeGroup, and one containing cell means. You can do this in one DATA step, like this:

**Program 16.15: Using a DATA Step to Create Separate Summary Data Sets**

```sas
data Grand(drop = Gender AgeGroup)
   By_gender(drop = AgeGroup)
   By_Age(drop = Gender)
   Cellmeans;
set Summary;
drop _type_
rename _freq_ = Number;
if _type_ = '00' then output Grand;
else if _type_ = '01' then output By_Age;
else if _type_ = '10' then output By_Gender;
else if _type_ = '11' then output Cellmeans;
run;
```

Because you want a different selection of variables in each data set, you can use a KEEP= or DROP= data set option to make your selection. Because you want to drop _TYPE_ from every data set, it is easier to do this with a DROP statement in the DATA step. A RENAME statement renames _FREQ_ to Number. Again, this change applies to all four data sets.

_TYPE_ is a character variable because of the CHARTYPE option, and you can use it to route each observation to the proper output data set.

Below are listings of the four data sets produced by Program 16.15:
Figure 16.15: Listing of the Four Data Sets Produced by Program 16.15

**Listing of Data Set Grand**

<table>
<thead>
<tr>
<th>Number</th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>5.48353</td>
<td>7042.97</td>
<td>916</td>
<td>908</td>
</tr>
</tbody>
</table>

**Listing of Data Set By_Gender**

<table>
<thead>
<tr>
<th>Gender</th>
<th>Number</th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>440</td>
<td>5.49848</td>
<td>7112.43</td>
<td>409</td>
<td>403</td>
</tr>
<tr>
<td>Male</td>
<td>560</td>
<td>5.47146</td>
<td>6987.54</td>
<td>507</td>
<td>505</td>
</tr>
</tbody>
</table>

**Listing of Data Set By_Age**

<table>
<thead>
<tr>
<th>AgeGroup</th>
<th>Number</th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old</td>
<td>598</td>
<td>5.45779</td>
<td>7011.56</td>
<td>551</td>
<td>540</td>
</tr>
<tr>
<td>Young</td>
<td>402</td>
<td>5.52238</td>
<td>7089.08</td>
<td>365</td>
<td>368</td>
</tr>
</tbody>
</table>

**Listing of Data Set Cellmeans**

<table>
<thead>
<tr>
<th>Gender</th>
<th>AgeGroup</th>
<th>Number</th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>Old</td>
<td>258</td>
<td>5.47921</td>
<td>7105.98</td>
<td>242</td>
<td>234</td>
</tr>
<tr>
<td>Female</td>
<td>Young</td>
<td>182</td>
<td>5.52641</td>
<td>7121.36</td>
<td>167</td>
<td>169</td>
</tr>
<tr>
<td>Male</td>
<td>Old</td>
<td>340</td>
<td>5.44100</td>
<td>6939.35</td>
<td>309</td>
<td>306</td>
</tr>
<tr>
<td>Male</td>
<td>Young</td>
<td>220</td>
<td>5.51899</td>
<td>7061.66</td>
<td>198</td>
<td>199</td>
</tr>
</tbody>
</table>
16.13 Selecting Different Statistics for Each Variable

There is an alternative way of selecting which variables and statistics you want in your summary data set, as shown in Program 16.16.

Program 16.16: Selecting Different Statistics for Each Variable Using PROC MEANS

```
proc means data=Learn.Blood noprint nway;
   class Gender AgeGroup;
   output out = Summary(drop = _:)
      mean(RBC WBC) =
      n(RBC WBC Chol) =
      median(Chol)    = / autoname;
run;
```

We have added several new features to this program. Each statistic keyword is followed by a list of variables (placed in parentheses) for which you want to compute this statistic. For example, this program computes means for RBC and WBC, the number of nonmissing values for all three variables (RBC, WBC, and Chol), and the median of Chol. Although you can name the newly created statistic following the equal sign as before, this program uses the AUTONAME output option to name them.

Finally, you may wonder about the rather strange-looking DROP= data set option in the SUMMARY data set. This DROP= option uses the colon wildcard notation. All variables beginning with an underscore character will be dropped. In general, **name**: used in any location where you would place a variable list refers to all variables beginning with the letters **name**. You can think of the colon the same way you use an asterisk as a DOS wildcard (if anyone still remembers DOS).

A listing of the Summary data set follows:

Figure 16.16: Listing of Data Set Summary

<table>
<thead>
<tr>
<th>Gender</th>
<th>AgeGroup</th>
<th>RBC_Mean</th>
<th>WBC_Mean</th>
<th>RBC_N</th>
<th>WBC_N</th>
<th>Chol_N</th>
<th>Chol_Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>Old</td>
<td>5.47921</td>
<td>7105.98</td>
<td>242</td>
<td>234</td>
<td>208</td>
<td>198</td>
</tr>
<tr>
<td>Female</td>
<td>Young</td>
<td>5.52641</td>
<td>7121.36</td>
<td>167</td>
<td>169</td>
<td>141</td>
<td>211</td>
</tr>
<tr>
<td>Male</td>
<td>Old</td>
<td>5.44100</td>
<td>6939.35</td>
<td>309</td>
<td>306</td>
<td>279</td>
<td>195</td>
</tr>
<tr>
<td>Male</td>
<td>Young</td>
<td>5.61899</td>
<td>7061.66</td>
<td>198</td>
<td>199</td>
<td>167</td>
<td>208</td>
</tr>
</tbody>
</table>

As you saw in the examples in this chapter, PROC MEANS can do a lot more than print summary statistics—it can create summary data sets broken down by one or more CLASS variables. By using options such as CHARTYPE and AUTONAME, you can simplify your programs and let SAS name your summary variables for you.

16.14 Printing all Possible Combinations of Your Class Variables

A very useful PROC MEANS option called PRINTALLTYPES prints summaries of every combination of your class variables. To demonstrate this, take a look at Program 16.17:
Program 16.17: Demonstrating the PRINTALLTYPES PROC MEANS Option

```latex
title "Demonstrating the PRINTALLTYPES Option";
proc means data=Learn.Blood printalltypes;
  class Gender AgeGroup;
  var RBC WBC;
run;
```

Adding this single option to PROC MEANS results in the following:

Figure 16.17: Output from Program 16.17

<table>
<thead>
<tr>
<th>N Obs</th>
<th>Variable</th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>RBC</td>
<td>916</td>
<td>5.4835252</td>
<td>0.9841158</td>
<td>1.710000</td>
<td>8.750000</td>
</tr>
<tr>
<td></td>
<td>WBC</td>
<td>908</td>
<td>7042.97</td>
<td>1003.37</td>
<td>4070.00</td>
<td>10550.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Age Group</th>
<th>N Obs</th>
<th>Variable</th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old</td>
<td>598</td>
<td>RBC</td>
<td>551</td>
<td>5.4577868</td>
<td>0.9900724</td>
<td>1.710000</td>
<td>8.430000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WBC</td>
<td>540</td>
<td>7011.56</td>
<td>1000.86</td>
<td>4290.00</td>
<td>10260.00</td>
</tr>
<tr>
<td>Young</td>
<td>402</td>
<td>RBC</td>
<td>365</td>
<td>5.52238336</td>
<td>0.9751199</td>
<td>2.920000</td>
<td>8.750000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WBC</td>
<td>368</td>
<td>7089.08</td>
<td>1006.63</td>
<td>4070.00</td>
<td>10650.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gender</th>
<th>N Obs</th>
<th>Variable</th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>440</td>
<td>RBC</td>
<td>409</td>
<td>5.4984841</td>
<td>0.9823118</td>
<td>1.710000</td>
<td>8.750000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WBC</td>
<td>403</td>
<td>7112.43</td>
<td>997.8255175</td>
<td>4620.00</td>
<td>10260.00</td>
</tr>
<tr>
<td>Male</td>
<td>560</td>
<td>RBC</td>
<td>507</td>
<td>5.4714596</td>
<td>0.9863729</td>
<td>2.330000</td>
<td>8.430000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WBC</td>
<td>505</td>
<td>6987.54</td>
<td>1005.31</td>
<td>4070.00</td>
<td>10550.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gender</th>
<th>Age Group</th>
<th>N Obs</th>
<th>Variable</th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>Old</td>
<td>258</td>
<td>RBC</td>
<td>242</td>
<td>5.4792149</td>
<td>0.9802759</td>
<td>1.710000</td>
<td>7.990000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>WBC</td>
<td>234</td>
<td>7105.98</td>
<td>1001.16</td>
<td>4620.00</td>
<td>10260.00</td>
</tr>
<tr>
<td></td>
<td>Young</td>
<td>182</td>
<td>RBC</td>
<td>167</td>
<td>5.5264072</td>
<td>0.9875345</td>
<td>2.920000</td>
<td>8.750000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>WBC</td>
<td>169</td>
<td>7121.36</td>
<td>996.0893384</td>
<td>4640.00</td>
<td>9820.00</td>
</tr>
<tr>
<td>Male</td>
<td>Old</td>
<td>340</td>
<td>RBC</td>
<td>309</td>
<td>5.4410032</td>
<td>0.9989426</td>
<td>2.330000</td>
<td>8.430000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>WBC</td>
<td>306</td>
<td>6939.35</td>
<td>996.2198541</td>
<td>4290.00</td>
<td>9360.00</td>
</tr>
<tr>
<td></td>
<td>Young</td>
<td>220</td>
<td>RBC</td>
<td>198</td>
<td>5.5189699</td>
<td>0.9670206</td>
<td>3.580000</td>
<td>7.720000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>WBC</td>
<td>199</td>
<td>7051.66</td>
<td>1017.20</td>
<td>4070.00</td>
<td>10550.00</td>
</tr>
</tbody>
</table>
This really useful option was not available when I wrote the last edition of this book. I'm happy to share it with you.

16.15 Problems

Solutions to odd-numbered problems are located at the back of this book. Solutions to all problems are available to professors. If you are a professor, visit the book’s companion website at support.sas.com/cody for information about how to obtain the solutions to all problems.

1. Using the SAS data set College, compute the mean, median, minimum, and maximum and the number of both missing and nonmissing values for the variables ClassRank and GPA. Report the statistics to two decimal places.

Repeat Problem 1, except compute the desired statistics for each combination of Gender and SchoolSize. Do this twice, once using a BY statement, and once using a CLASS statement. Note: The data set College has permanent formats for Gender, SchoolSize, and Scholarship. Either make this format catalog available to SAS (see Chapter 5), run the PROC FORMAT statements that follow, or use the system option NOFMTERR (no format error) that allows you to access SAS data sets that have permanent user-defined formats without causing an error.

```sas
proc format;
  value $Yesno 'Y','1' = 'Yes'
                 'N','0' = 'No'
                ' '   = 'Not Given';
  value $Size   'S' = 'Small'
                'M' = 'Medium'
                'L' = 'Large'
                ' ' = 'Missing';
  value $Gender 'F' = 'Female'
                'M' = 'Male'
                ' ' = 'Not Given';
run;
```

2. Using the SAS data set College, report the mean and median GPA and ClassRank broken down by school size (SchoolSize). Do this twice, once using a BY statement, and once using a CLASS statement.

3. Repeat Problem 3 (CLASS statement only), except group small and medium school sizes together. Do this by writing a new format for SchoolSize (values are S, M, and L). Do not use any DATA steps.

4. Using the SAS data set College, report the mean GPA for the following categories of ClassRank: 0–50 = bottom half, 51–74 = 3rd quartile, and 75 to 100 = top quarter. Do this by creating an appropriate format. Do not use a DATA step.

5. Using the SAS data set College, create a summary data set (call it Class_Summary) containing the n, mean, and median of ClassRank and GPA for each value of SchoolSize. Use a CLASS statement and be sure that the summary data set only contains statistics for each level of SchoolSize. Use the AUTONAME option to name the variables in this data set.

6. Using the SAS data set College, create four summary data sets containing the number of nonmissing and missing values and the mean, minimum, and maximum for ClassRank and GPA, broken down by Gender and SchoolSize. The first data set (Grand) should contain the statistics for all subjects, the second data set (ByGender) should contain the statistics broken down by Gender,
the third data set (BySize) should contain the statistics broken down by SchoolSize, and the fourth
data set (Cell) should contain the statistics broken down by Gender and SchoolSize. Do this by
using PROC MEANS (with a CLASS statement) and one DATA step.

**Hint:** Use the CHARTYPE procedure option.

7. Using the SAS data set SASHELP.Cars, compute the mean and standard deviation for Horsepower
broken down by Make and Origin. Include the number of nonmissing and missing values in the
output and print all statistics to one decimal place. Use the PRINTALLTYPES option to see all
possible combinations of the CLASS variables.

Use PROC PRINT (without any DATA steps) to create a listing like the one here.

**Note:** The variables in the Hosp data set are Subject, AdmitDate (Admission Date), DischrDate
(Discharge Date), and DOB (Date of Birth).

### Selected Patients from Hosp Data Set
### Admitted in September of 2004
### Older than 83 years of age

<table>
<thead>
<tr>
<th>Subject</th>
<th>Date of Birth</th>
<th>Admission Date</th>
<th>Discharge Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>03/21/1921</td>
<td>09/13/2004</td>
<td>09/22/2004</td>
</tr>
<tr>
<td>407</td>
<td>08/26/1920</td>
<td>09/13/2004</td>
<td>09/18/2004</td>
</tr>
<tr>
<td>409</td>
<td>01/01/1921</td>
<td>09/13/2004</td>
<td>10/02/2004</td>
</tr>
<tr>
<td>2577</td>
<td>04/30/1920</td>
<td>09/27/2004</td>
<td>09/27/2004</td>
</tr>
<tr>
<td>7495</td>
<td>02/11/1921</td>
<td>09/21/2004</td>
<td>09/22/2004</td>
</tr>
</tbody>
</table>

**Hints:** Variable labels replace variable names. The number of observations is printed at the
bottom of the listing. There are four title lines (the last one being a dashed line).
Ready to take your SAS® and JMP® skills up a notch?

Be among the first to know about new books, special events, and exclusive discounts.

support.sas.com/newbooks

Share your expertise. Write a book with SAS.

support.sas.com/publish

sas.com/books for additional books and resources.