SAS Programming (BIOL-4V190)

Chapter 5

Creating Formats and Labels

Formats & labels can be added to your variables and data sets to increase the readability of your output and provide additional information about your data.

5.1 Adding Labels to Your Variables

Labels can be added to your variables in either a data step or a proc step.

Adding a label to a variable in a data step creates a PERMANENT label. Adding a label to a variable in a proc step creates a TEMPORARY label.

Syntax:

```
label variable1='description'
      variable2='another description here'
      variable3='more text here';
```

Just like format statements, when a label statement is applied in a data step, the label associated with a variable is said to be a permanent label. This is because in a data set, the label is stored with the variable.

Further, when a labelled variable is used in a procedure, in some procedures, only the variable label will be displayed, while in other procedures, both the variable label and the variable name will be displayed.

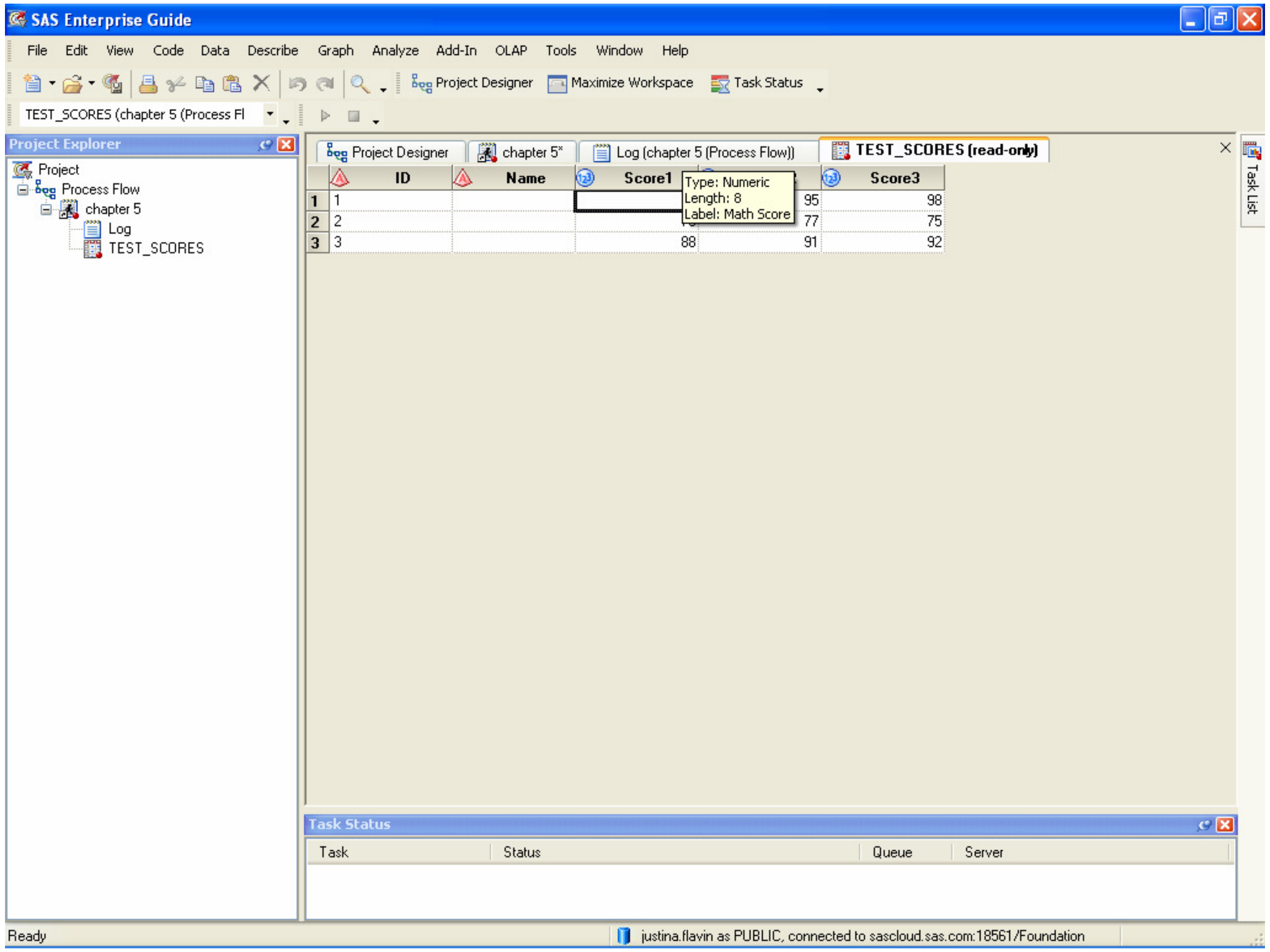
f you wish to display a variable using a different label in a procedure, a label statement can be used. This will create a temporary label that is only applied in that procedure.

Finally, to remove a label from a variable, add a null label to the variable as follows: `label id=‘ ‘;`

Run the code for Program 5-1.

Hovering over the column for one of the labelled variables, we can see the variable label.

You may also wish to run a PROC CONTENTS on this data set and see how the addition of the label statement changes that output.



```
   ID   Name   Score1        Score3
1  1                    95       98
2  2                    77       75
3  3            88      91       92
```

Type: Numeric
Length: 8
Label: Math Score

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

5.2 Using Formats to Enhance Your Output

SAS provides some built-in formats, but you can also create your own formats using a procedure called PROC FORMAT.
You must always create a format before you can apply it to variables.

Syntax:

```
proc format;
  /* numeric format */
  value formatname
    value1 = ‘description’
    value2 = ‘another description’
     ;
  /* character format */
  value $formatname
    ‘value1’ = ‘description’
    ‘value2’ = ‘another description’
      ;
run;
```

If you are creating a format that will be used with a character variable, the “$” character must precede the format name.

The data values are listed on the left side, and the descriptions or formats are listed on the right side of the “=“.

We will now discuss the formats shown in Program 5-2:

```
proc format;
  value $gender 'M' = 'Male'
                'F' = 'Female‘
                ' ' = 'Not entered'
                other = 'Miscoded';

  value age low-29  = 'Less than 30'
            30-50  = '30 to 50'
            51-high = '51+';

   value $likert '1' = 'Strongly disagree'
                 '2' = 'Disagree'
                 '3' = 'No opinion'
                 '4' = 'Agree'
                 '5' = 'Strongly agree';

   value pets 1 = 'Luna'
              2 = 'Azuel'
              3 = 'Killian'
              6 = 'Charlie';
run;
```

```
value $gender 'M' = 'Male'
              'F' = 'Female'
              ' ' = 'Not entered'
              other = 'Miscoded';
```

This creates a character format called gender.

Values of M and F in the data receive the formats ‘Male’ and ‘Female’ respectively.

Missing values in the data will be formatted as ‘Not entered’.

Other is a keyword that can be used in PROC FORMAT to specify all other data values.

So anything other than M, F, or a missing value in the data is formatted as Miscoded.

```
value age low-29  = 'Less than 30'
          30-50   = '30 to 50'
          51-high = '51+';
```

This creates a numeric format called age.

Low and High are keywords that can be used in place of the lowest and highest data values for the given variable.

So low-29 specifies all data values from the lowest value through 29, 30-50 specifies all data values between 30 and 50, 51-high specifies all data values from 51 to the highest value.

Should your data contain a noninteger value such as 29.5, that value would not fall into any of the specified categories and would appear on your output as 29.5.

```
value $likert '1' = 'Strongly disagree'
             '2' = 'Disagree'
             '3' = 'No opinion'
             '4' = 'Agree'
             '5' = 'Strongly agree';
```

This creates a character format called likert.

Although the values on the left appear to be numbers, to SAS they are characters since they are enclosed in quotes.

When this format is applied, the data will appear on the output as the character descriptions on the right hand side rather than the numeric values.

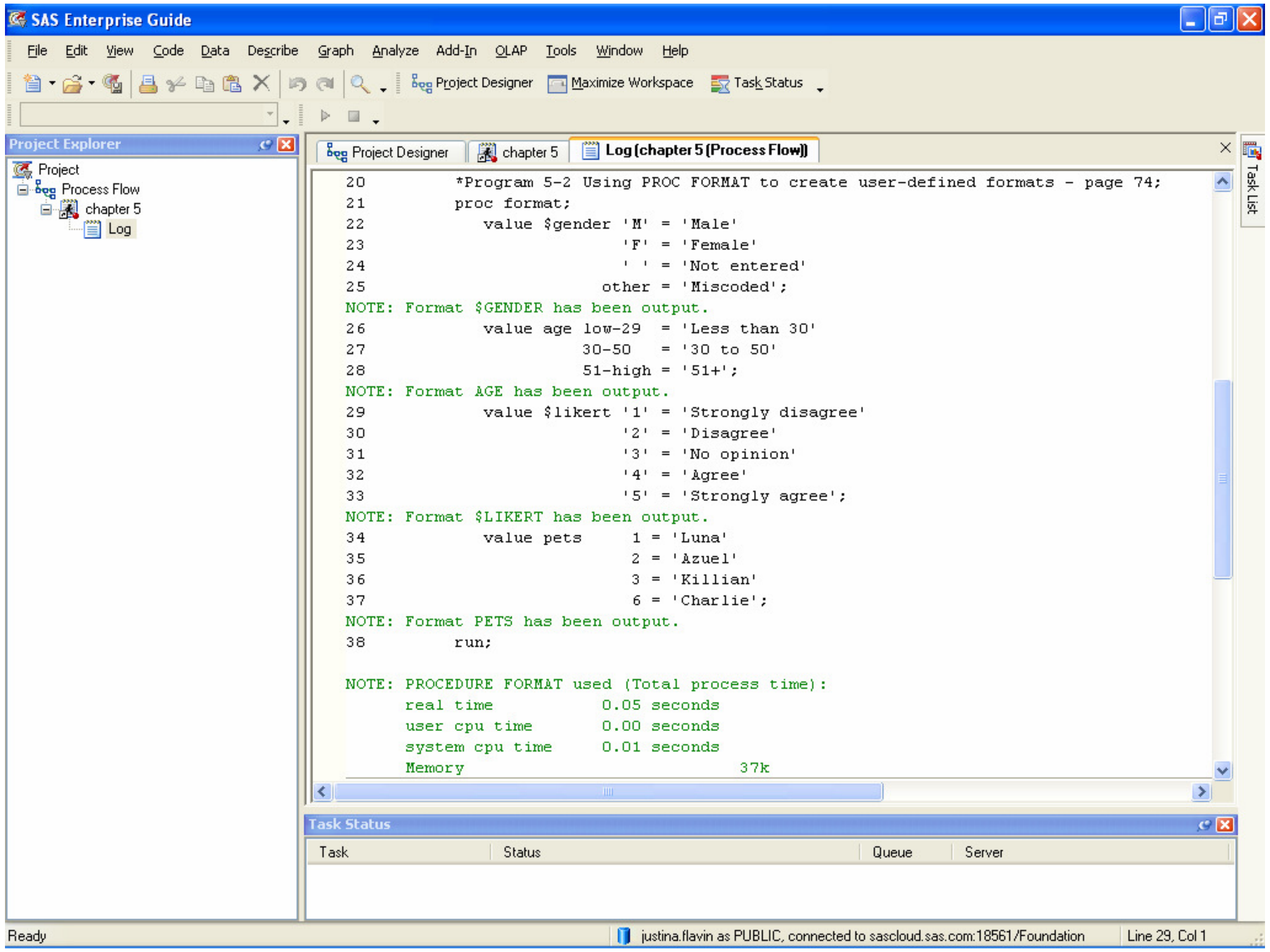
Like the previous example, should a value of ‘6’ be present in the data, it would be printed as 6 since no formatted value has been defined for ‘6’.

```
value pets     1 = 'Luna'
              2 = 'Azuel'
              3 = 'Killian'
              6 = 'Charlie';
```

This creates a numeric format called pets.

Notice that the use of this format will cause a numeric variable to look like a character variable in any output produced.

Running the format code does not produce any output, but generates a message in the log that indicates that the formats have been created and can now be used.



```
20         *Program 5-2 Using PROC FORMAT to create user-defined formats - page 74;
21         proc format;
22            value $gender 'M' = 'Male'
23                          'F' = 'Female'
24                          ' ' = 'Not entered'
25                        other = 'Miscoded';
NOTE: Format $GENDER has been output.
26            value age low-29  = 'Less than 30'
27                      30-50   = '30 to 50'
28                      51-high = '51+';
NOTE: Format AGE has been output.
29            value $likert '1' = 'Strongly disagree'
30                          '2' = 'Disagree'
31                          '3' = 'No opinion'
32                          '4' = 'Agree'
33                          '5' = 'Strongly agree';
NOTE: Format $LIKERT has been output.
34            value pets     1 = 'Luna'
35                           2 = 'Azuel'
36                           3 = 'Killian'
37                           6 = 'Charlie';
NOTE: Format PETS has been output.
38         run;

NOTE: PROCEDURE FORMAT used (Total process time):
      real time           0.05 seconds
      user cpu time       0.00 seconds
      system cpu time     0.01 seconds
      Memory                              37k
```

Before looking at an example where we apply these formats, we will return to a discussion of PROC PRINT.

By default, PROC PRINT prints the observation number on the left side of the output.

There are two ways to suppress printing of the observation numbers.

(1) Add an ID statement

Syntax:

```
proc print;
  id varname1 varname2...varname'n';
  var varname1 varname2...varname'n'
run;
```

Note: Usually you do not include a variable on both the **ID** and **VAR** statements, otherwise the variable appears twice in the output.

(2) Use the NOOBS option on the PROC PRINT statement.

Syntax:

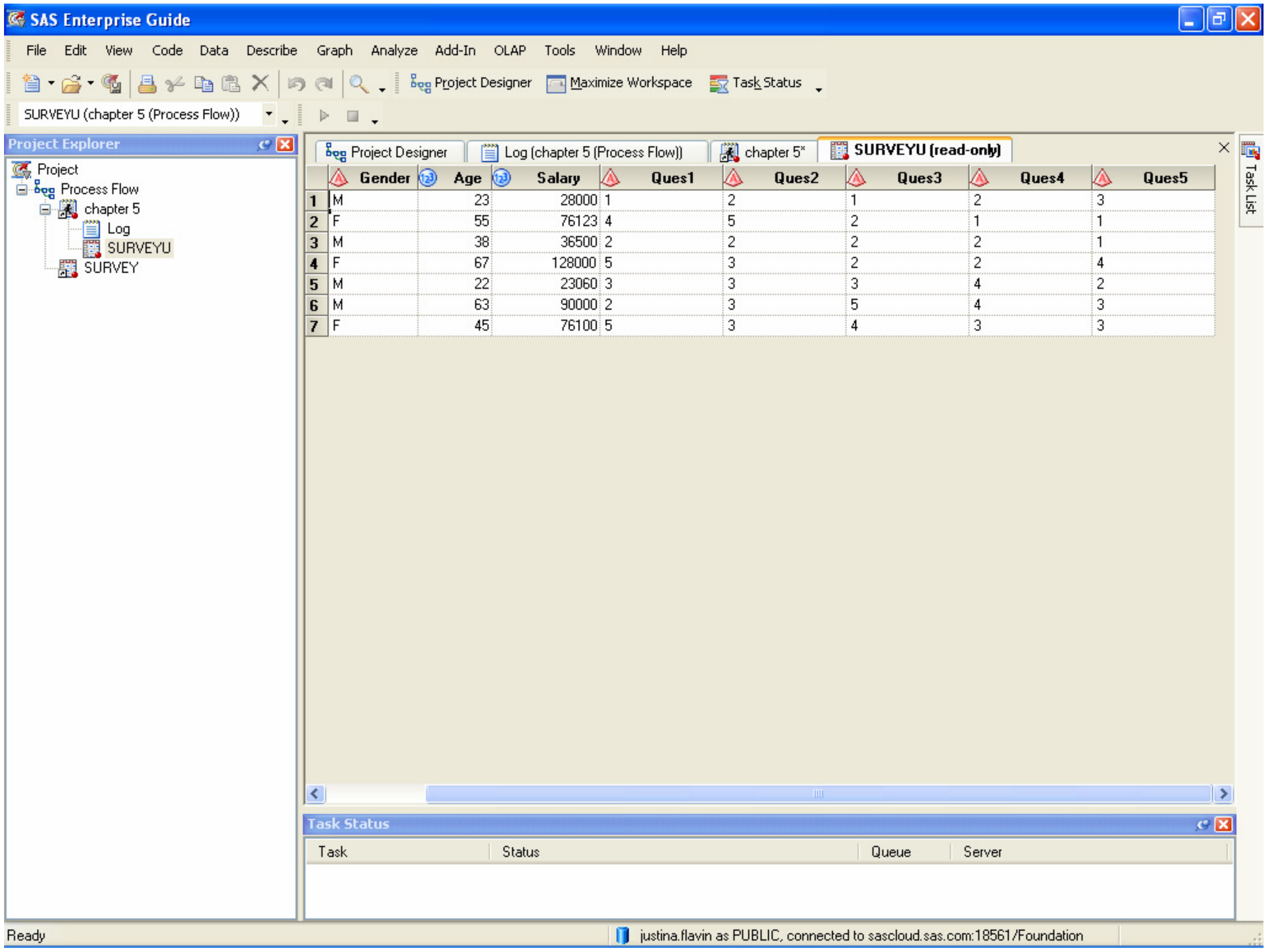
```
proc print noobs;
  var varname1 varname2...varname'n'
run;
```

Here is an example illustrating the use of both the ID statement and some of the user-defined formats which were created:

```
*Program 5-3 Adding a FORMAT statement in PROC PRINT - page 75;
title "Data Set SURVEY with Formatted Values - page 75";
proc print data=learn.surveyu;
  id ID;
  var Gender Age Salary Ques1-Ques5;
  format Gender      $gender.
         Age          age.
         Ques1-Ques5 $likert.
         Salary       dollar11.2;
run;
```

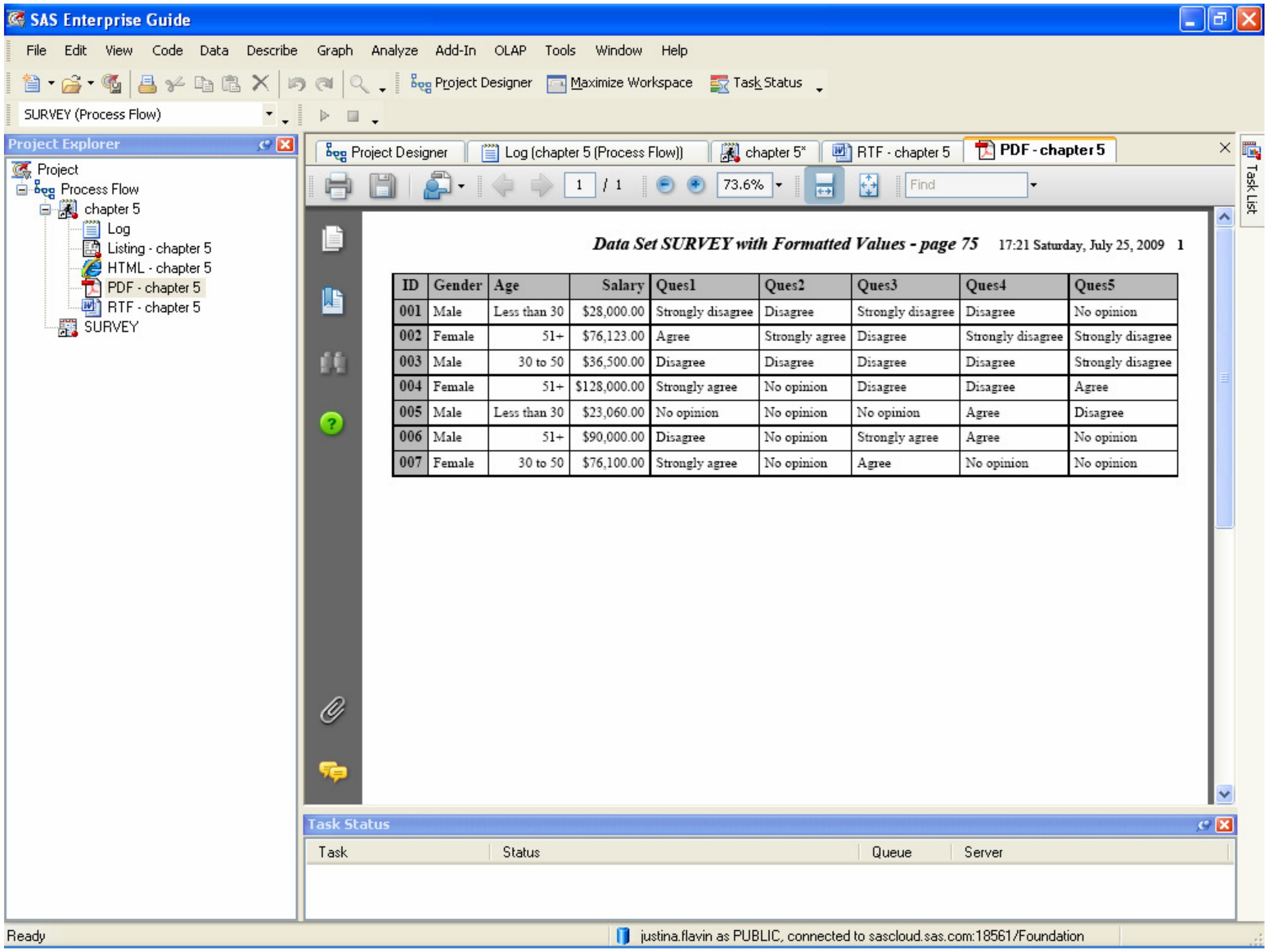
Notice in this example that Salary is formatted with one of the automatically available formats (dollar.). Also notice that the format $likert. is applied to 5 variables, Ques1-Ques5.

First let's take a look at the raw (unformatted) data.


SAS Enterprise Guide
File Edit View Code Data Describe Graph Analyze Add-In OLAP Tools Window Help
Project Designer Maximize Workspace Task Status
SURVEYU (chapter 5 (Process Flow))
Project Explorer
Project
Process Flow
chapter 5
Log
SURVEYU
SURVEY
Project Designer Log (chapter 5 (Process Flow)) chapter 5* SURVEYU (read-only)
Task List
Task Status
Task Status Queue Server
Ready
justina.flavin as PUBLIC, connected to sascloud.sas.com:18561/Foundation

	Gender	Age	Salary	Ques1	Ques2	Ques3	Ques4	Ques5
1	M	23	28000	1	2	1	2	3
2	F	55	76123	4	5	2	1	1
3	M	38	36500	2	2	2	2	1
4	F	67	128000	5	3	2	2	4
5	M	22	23060	3	3	3	4	2
6	M	63	90000	2	3	5	4	3
7	F	45	76100	5	3	4	3	3

And here is the output with the formats applied to the variables:



Data Set SURVEY with Formatted Values - page 75 17:21 Saturday, July 25, 2009 1

ID	Gender	Age	Salary	Ques1	Ques2	Ques3	Ques4	Ques5
001	Male	Less than 30	$28,000.00	Strongly disagree	Disagree	Strongly disagree	Disagree	No opinion
002	Female	51+	$76,123.00	Agree	Strongly agree	Disagree	Strongly disagree	Strongly disagree
003	Male	30 to 50	$36,500.00	Disagree	Disagree	Disagree	Disagree	Strongly disagree
004	Female	51+	$128,000.00	Strongly agree	No opinion	Disagree	Disagree	Agree
005	Male	Less than 30	$23,060.00	No opinion	No opinion	No opinion	Agree	Disagree
006	Male	51+	$90,000.00	Disagree	No opinion	Strongly agree	Agree	No opinion
007	Female	30 to 50	$76,100.00	Strongly agree	No opinion	Agree	No opinion	No opinion

5.3 Regrouping Values Using Formats

Formats can be used to regroup or recategorise data for data displays without changing the underlying data values.

In this example, we create a new format and apply it in a procedure:

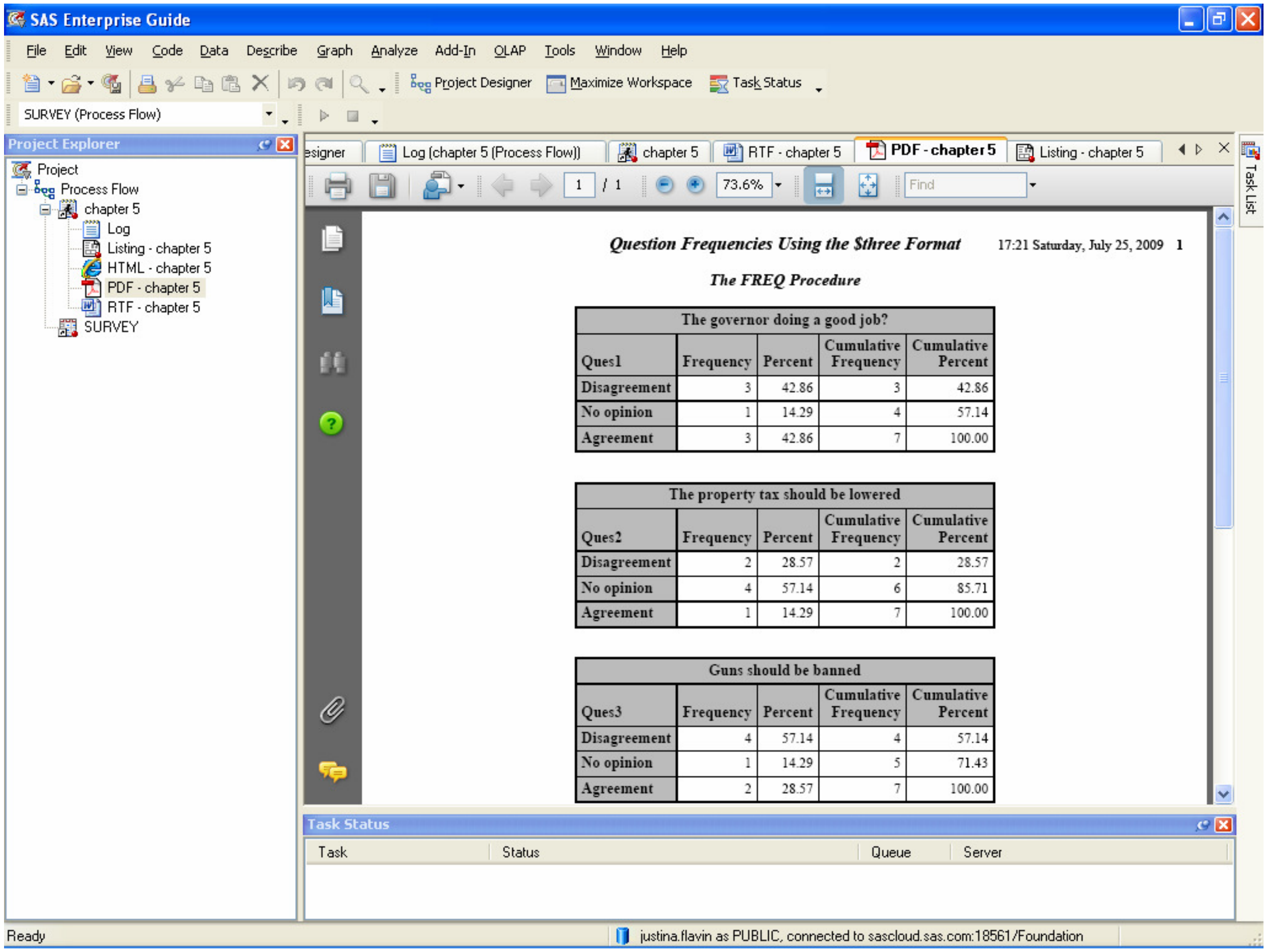
```
*Program 5-4 Regrouping values using a format - page 77;
proc format;
  value $three '1','2' = 'Disagreement'
               '3'     = 'No opinion'
               '4','5' = 'Agreement';
run;

*Program 5-5 Applying the new format to several variables with PROC FREQ - page 77;
proc freq data=learn.survey;
  title "Question Frequencies Using the $three Format";
  tables Ques1-Ques5;
  format Ques1-Ques5 $three.;
run;
```

The frequency counts are produced based upon the formatted values of the data.

Notice too that the variable labels are displayed in the output. But we did not add labels in the PROC FREQ code, so where did the labels come from?

View the data set LEARN.SURVEY and you will see that there are labels on these variables.



Question Frequencies Using the $three Format 17:21 Saturday, July 25, 2009 1

The FREQ Procedure

The governor doing a good job?

Ques1	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Disagreement	3	42.86	3	42.86
No opinion	1	14.29	4	57.14
Agreement	3	42.86	7	100.00

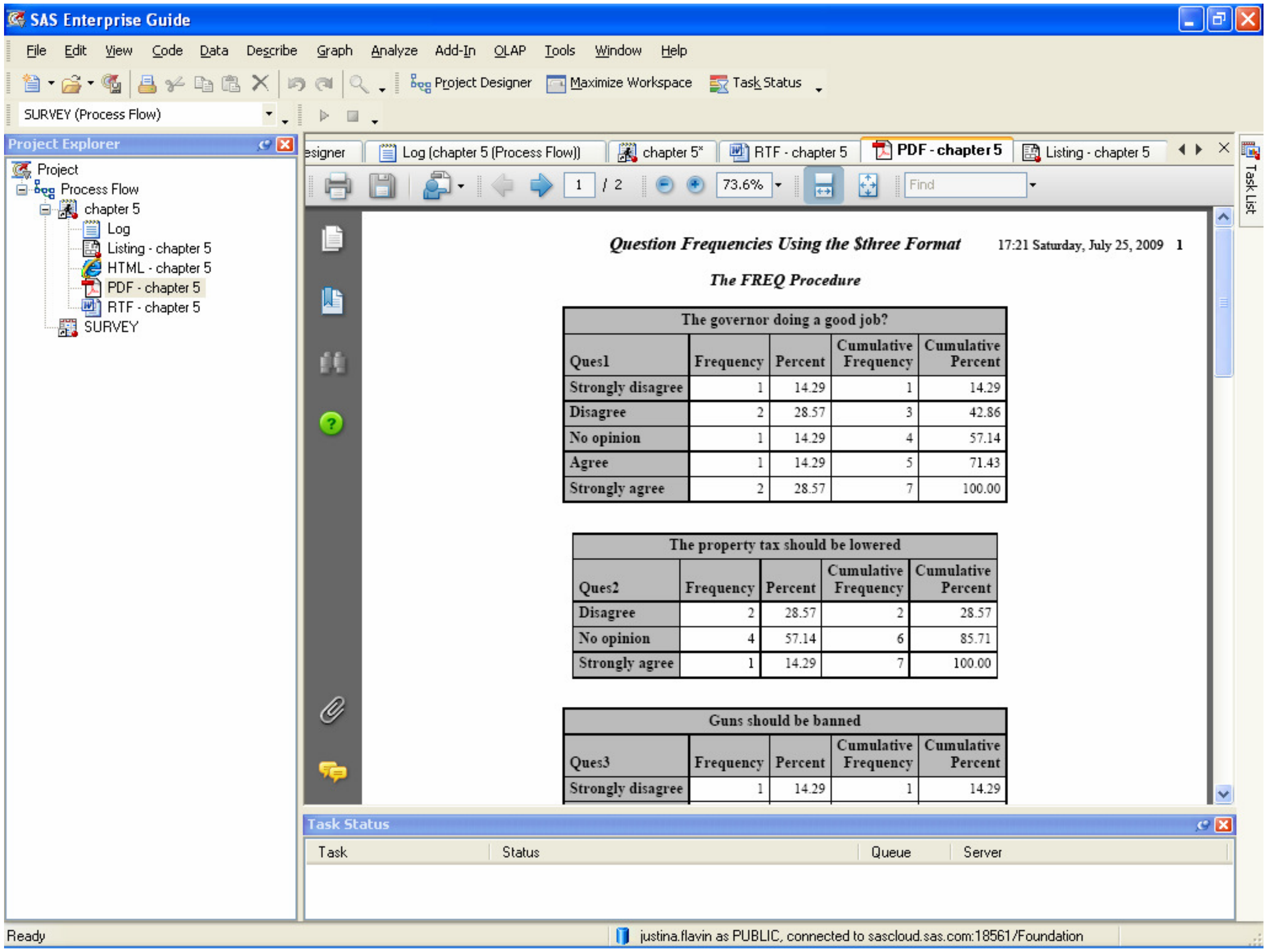
The property tax should be lowered

Ques2	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Disagreement	2	28.57	2	28.57
No opinion	4	57.14	6	85.71
Agreement	1	14.29	7	100.00

Guns should be banned

Ques3	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Disagreement	4	57.14	4	57.14
No opinion	1	14.29	5	71.43
Agreement	2	28.57	7	100.00

Now remove the format statement and rerun the code. Notice the frequency counts. Again, the values displayed are formatted values, but the categories are different. But the format statement was removed so how can these be formatted values? Again, view the data set LEARN.SURVEY and you will see that there are formats on these variables. Adding a format statement in PROC FREQ causes a new format to be applied temporarily, removing it causes the permanent format (that is stored with the data on the data set) to be used.



Question Frequencies Using the $three Format 17:21 Saturday, July 25, 2009 1

The FREQ Procedure

The governor doing a good job?

Ques1	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Strongly disagree	1	14.29	1	14.29
Disagree	2	28.57	3	42.86
No opinion	1	14.29	4	57.14
Agree	1	14.29	5	71.43
Strongly agree	2	28.57	7	100.00

The property tax should be lowered

Ques2	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Disagree	2	28.57	2	28.57
No opinion	4	57.14	6	85.71
Strongly agree	1	14.29	7	100.00

Guns should be banned

Ques3	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Strongly disagree	1	14.29	1	14.29

5.4 More on Format Ranges

The ‘less than’ operator “<“ can be used in format statements.
However, the ‘greater than’ operator “>” cannot be used.

Here is an alternative way to create the format for age.
If there was a data value of 29.5 in the data, it would now appear in the output formatted as ‘Less than 30’.

```
proc format;
  value age low-<30  = 'Less than 30'
            30-<51   = '30 to 50'
            51-high = '51+';
run;
```

5.5 Storing Your Formats in a Format Library

Storing your user defined formats in a library makes them easily accessible in programs and prevents having to include and rerun the PROC FORMAT code in your programs.
To store user defined formats in a library, add a libname statement and a library statement option on the PROC FORMAT statement as follows:

Syntax:

```
libname libref ‘directory name and path’;
proc format library=libref;
  value formatname
        value1 = ‘description’
        value2 = ‘another description’;
run;
```

Here is the example code for creating a permanent format library.

Note: Because of the limitations of writing to the server, it will not be possible for you to execute and create a permanent format library.

However, I have executed the code below to create the format library:

```
*Program 5-6 Creating a permanent format library;
libname myfmts "/courses/u_ucsd.edu1/i_536036/c_629/saslib";

proc format library=myfmts;
  value $gender 'M' = 'Male'
                'F' = 'Female'
                ' ' = 'Not entered'
                other = 'Miscoded';
  value age low-29  = 'Less than 30'
            30-50   = '30 to 50'
            51-high = '51+';
  value $likert '1' = 'Strongly disagree'
                '2' = 'Disagree'
                '3' = 'No opinion'
                '4' = 'Agree‘
                '5' = 'Strongly agree';
run;
```

5.6 Permanent Data Set Attributes

Adding labels and formats to variables in a data step creates a PERMANENT association with the variables.

If the formats are user defined, it then becomes necessary to “tell” SAS where to find these formats.

This can be accomplished by (1) including the PROC FORMAT statement in the code prior to assigning the format in the data step or (2) storing the format in a library and referencing it in the program using a global statement called the OPTIONS statement.

Syntax:

```
libname libref1 ‘directory name and path’;
libname libref2 ‘directory name and path’;
options fmtsearch=(libref1 libref2);
```

On the server, our format catalog must be stored in the same directory as the data sets:

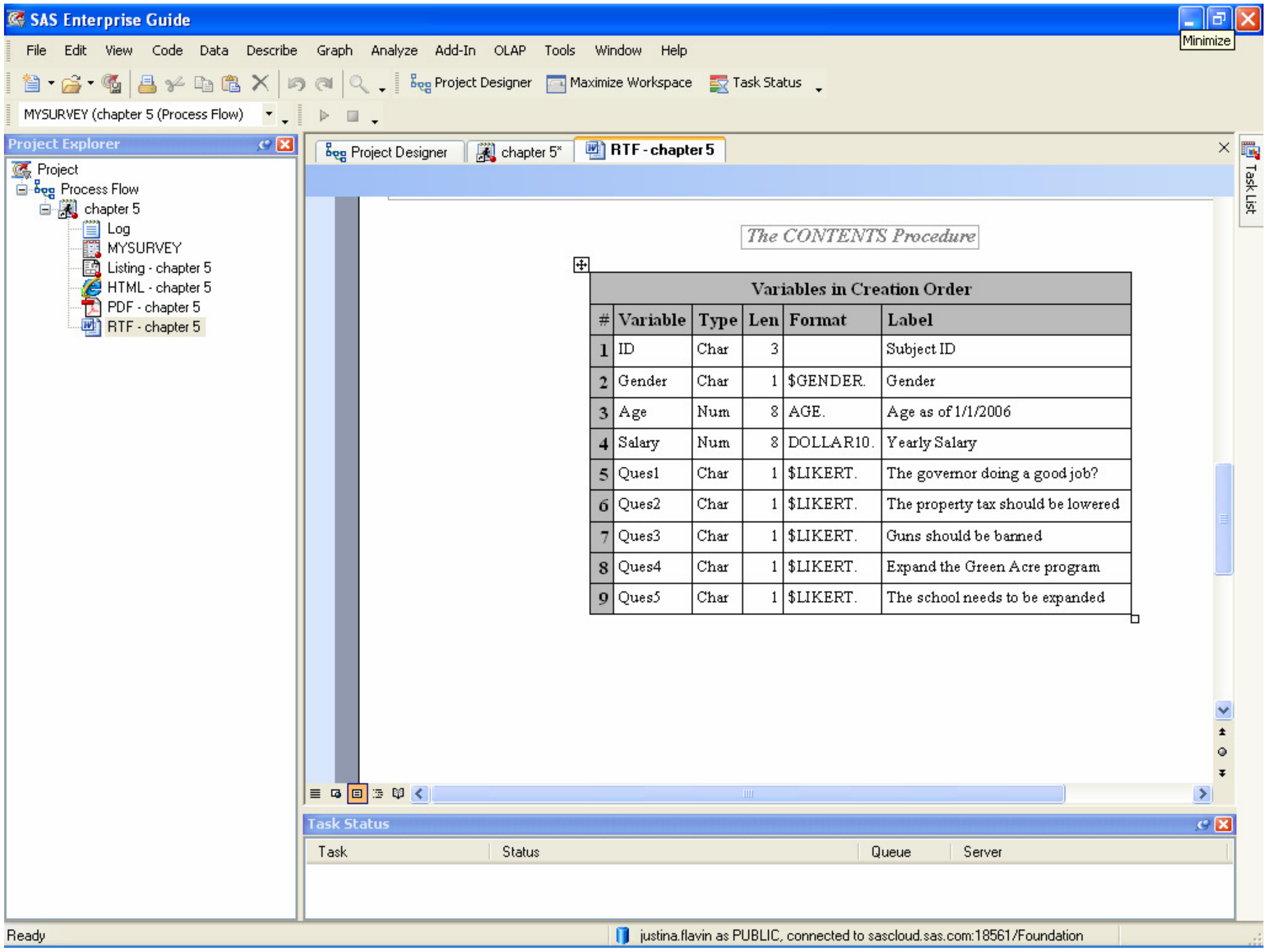
```
*Program 5-7 Adding label and format statements in the DATA step - page 81;
libname myfmts "/courses/u_ucsd.edu1/i_536036/c_629/saslib" access=readonly;
options fmtsearch=(myfmts);
```

Now we can run the code to create the data set mysurvey which has variables with permanent formats and labels. These will be visible on the PROC CONTENTS output.

```
data mysurvey;
  infile "/courses/u_ucsd.edu1/i_536036/c_629/survey.txt" pad;
  input ID : $3.
        Gender : $1.
        Age
        Salary
        (Ques1-Ques5)(: $1.);
  format Gender      $gender.
         Age         age.
         Ques1-Ques5 $likert.
         Salary      dollar10.0;
  label ID     = 'Subject ID'
        Gender = 'Gender'
        Age    = 'Age as of 1/1/2006'
        Salary = 'Yearly Salary'
        Ques1  = 'The governor doing a good job?'
        Ques2  = 'The property tax should be lowered'
        Ques3  = 'Guns should be banned'
        Ques4  = 'Expand the Green Acre program'
        Ques5  = 'The school needs to be expanded';
run;

*Program 5-8 Running PROC CONTENTS on a data set with labels and formats - page 81;
title "Data set SURVEY";
proc contents data=mysurvey varnum;
run;
```

Here is the PROC CONTENTS output.



The CONTENTS Procedure

Variables in Creation Order					
#	Variable	Type	Len	Format	Label
1	ID	Char	3		Subject ID
2	Gender	Char	1	$GENDER.	Gender
3	Age	Num	8	AGE.	Age as of 1/1/2006
4	Salary	Num	8	DOLLAR10.	Yearly Salary
5	Ques1	Char	1	$LIKERT.	The governor doing a good job?
6	Ques2	Char	1	$LIKERT.	The property tax should be lowered
7	Ques3	Char	1	$LIKERT.	Guns should be banned
8	Ques4	Char	1	$LIKERT.	Expand the Green Acre program
9	Ques5	Char	1	$LIKERT.	The school needs to be expanded

In this previous example, a new global statement, the system options statement, was introduced:

Syntax:

```
options option(s);
```

System options are instructions that affect your SAS session.

They control the way that SAS performs operations such as SAS System initialization, hardware and software interfacing, and the input, processing, and output of jobs and SAS files.

As a global statement, the options statement is a standalone statement that you place outside of a data step or a procedure.

There are many *options* that can be added to the statement, and as the course proceeds you will see others.

For additional information about global options and the options statement, refer to the online documentation:

http://support.sas.com/documentation/onlinedoc/91pdf/index.html and select SAS OnlineDoc 9.1 for the Web.

Then navigate to Base SAS -> SAS Language Reference: Dictionary -> Dictionary of Language Elements -> SAS System Options

Two options that relate to format libraries are FMTSEARCH & FMTERR.

5.7 Accessing a Permanent SAS Data Set with User-Defined Formats

If you have a formatted variable in a data set, by default, the formatted values are displayed in procedure output.

If you wish to remove the format either temporarily or permanently, you can use a null format statement to 'unformat' the variables.

Syntax:

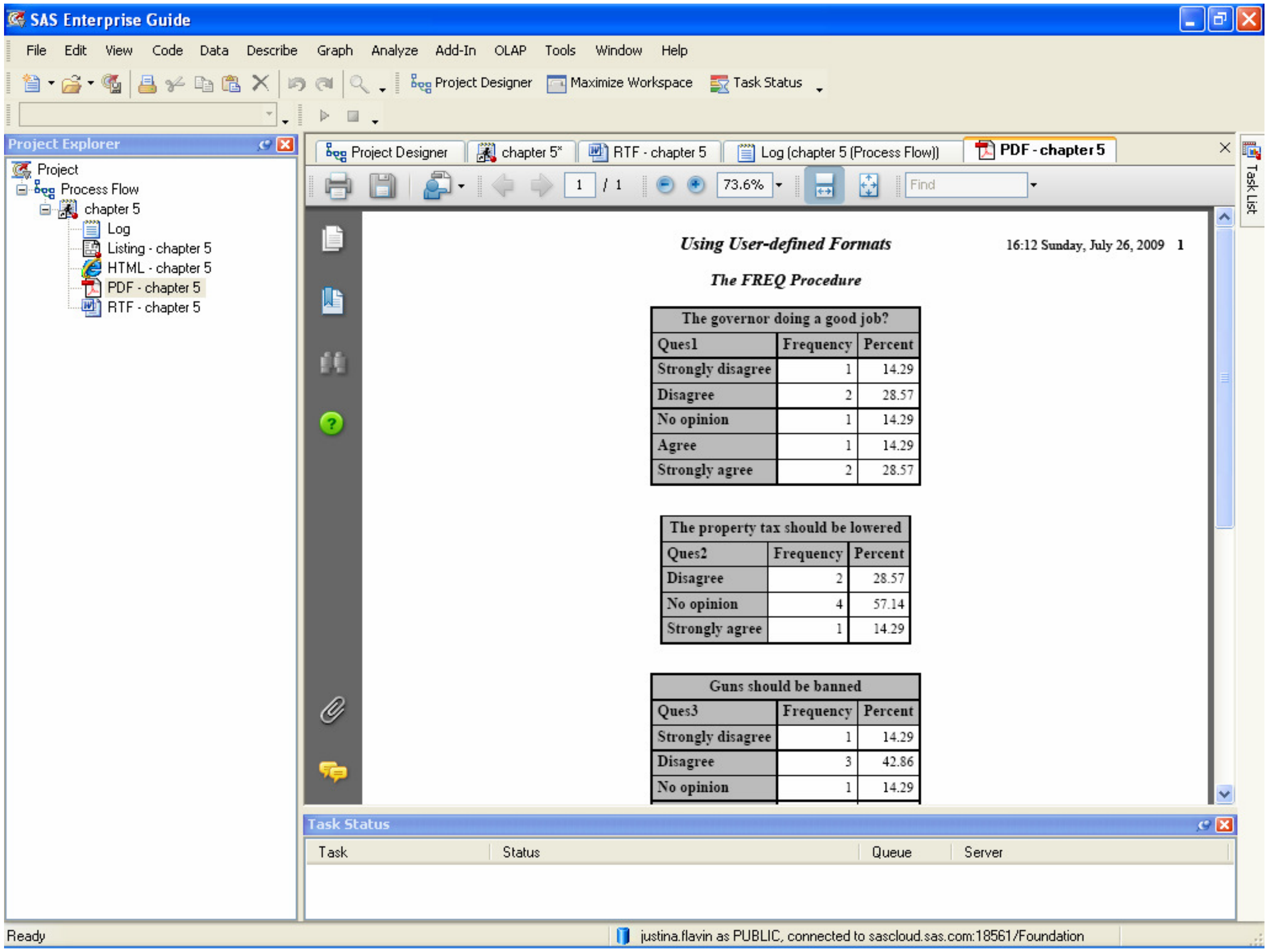
```
format variablename1 variablename2 variablename3;
```

Let's look at the difference in these outputs:

```
title "Using User-defined Formats";
proc freq data=learn.survey;
  tables Ques1-Ques5 / nocum;
run;

title Removing Formats;
proc freq data=learn.survey;
  tables Ques1-Ques5 / nocum;
  format Ques1-Ques5;
run;
```

Since Ques1-Ques5 are formatted variables, the output are displayed using the formatted values.



Using User-defined Formats 16:12 Sunday, July 26, 2009 1

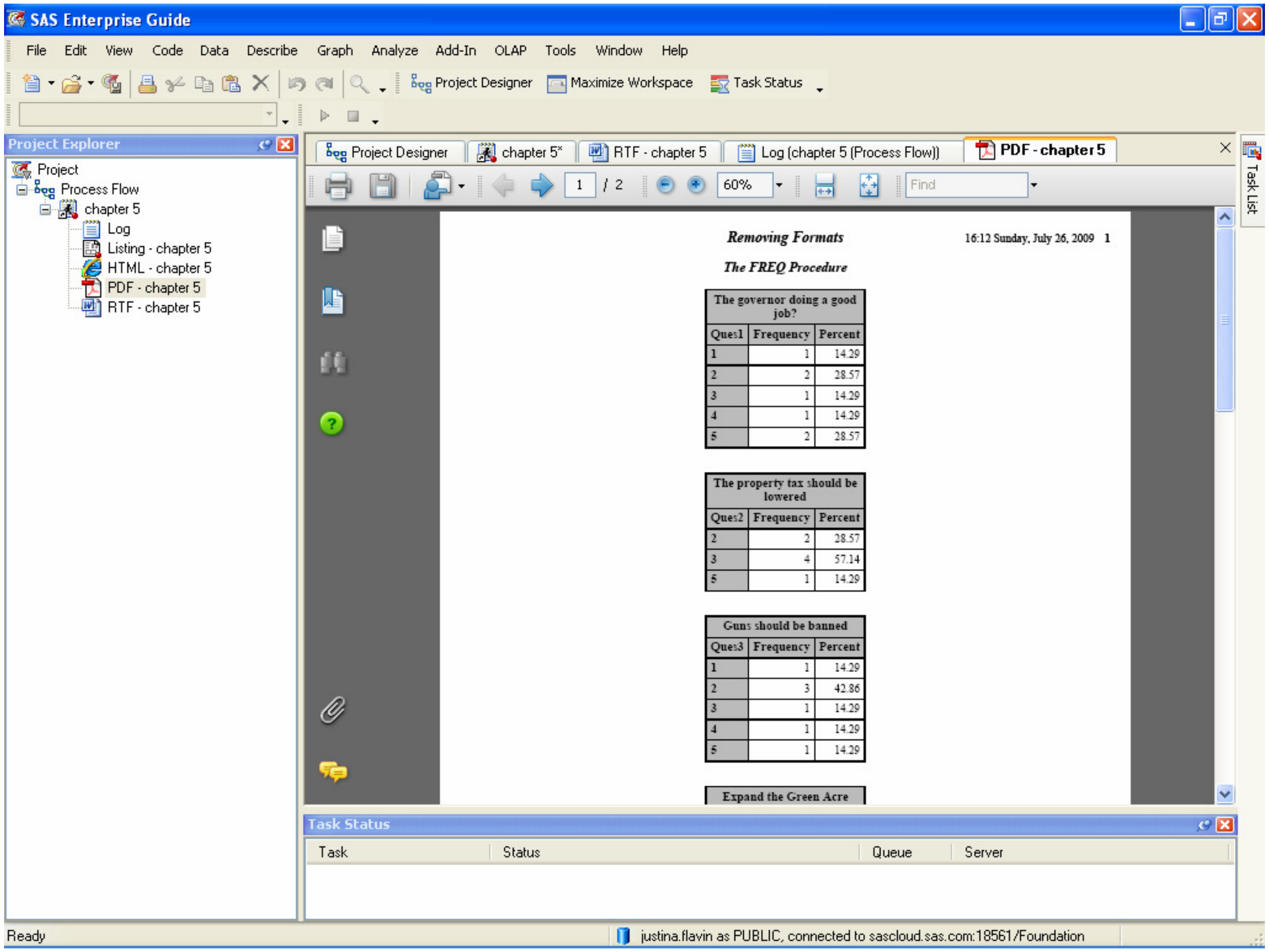
The FREQ Procedure

The governor doing a good job?		
Ques1	Frequency	Percent
Strongly disagree	1	14.29
Disagree	2	28.57
No opinion	1	14.29
Agree	1	14.29
Strongly agree	2	28.57

The property tax should be lowered		
Ques2	Frequency	Percent
Disagree	2	28.57
No opinion	4	57.14
Strongly agree	1	14.29

Guns should be banned		
Ques3	Frequency	Percent
Strongly disagree	1	14.29
Disagree	3	42.86
No opinion	1	14.29

By adding a null format statement, the output is displayed using the raw data values.
Because the null format statement is used in a procedure, the formats are removed temporarily for this procedure only.
There is no effect on the data set. The formats remain associated with the variables in the data set.



Removing Formats 16:12 Sunday, July 26, 2009 1

The FREQ Procedure

The governor doing a good job?		
Ques1	Frequency	Percent
1	1	14.29
2	2	28.57
3	1	14.29
4	1	14.29
5	2	28.57

The property tax should be lowered		
Ques2	Frequency	Percent
2	2	28.57
3	4	57.14
5	1	14.29

Guns should be banned		
Ques3	Frequency	Percent
1	1	14.29
2	3	42.86
3	1	14.29
4	1	14.29
5	1	14.29

Expand the Green Acre

5.8 Displaying Your Format Definitions

To create a listing of all formats in a library, use PROC FORMAT with the FMTLIB option.

Syntax:

```
libname libref 'directory name and path';
proc format library=libref fmtlib;
run;
```

On the server, our format library resides in the same directory as the data sets:
`/courses/u_ucsd.edu1/i_536036/c_629/saslib`

To view only certain formats in a library, use PROC FORMAT with a SELECT statement.

Syntax:

```
libname libref 'directory name and path';
proc format library=libref fmtlib;
  select formatname1 formatname2...formatname'n';
run;
```

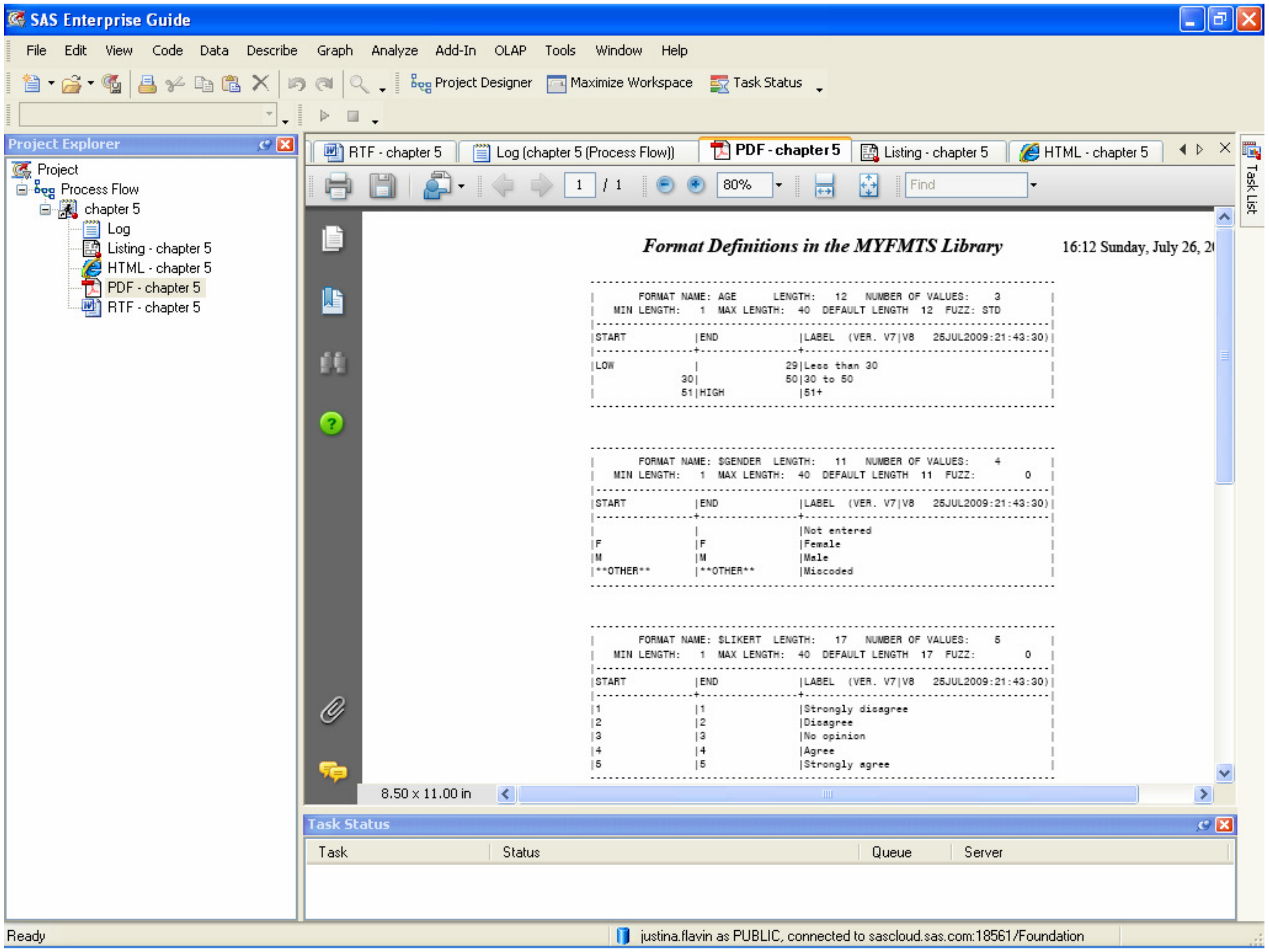
The book states that "when you use a SELECT statement, you do not also have to include the FMTLIB option".

Again because of the server limitations, this is not true for this class – you DO need the FMTLIB option.

When I created this slide, there were 3 formats stored in the format library.

A format library is additive, so once it is created, old formats remain and new formats are added, so you may find additional formats in the library when you execute this code.

Each box in the output provides the details about a format.



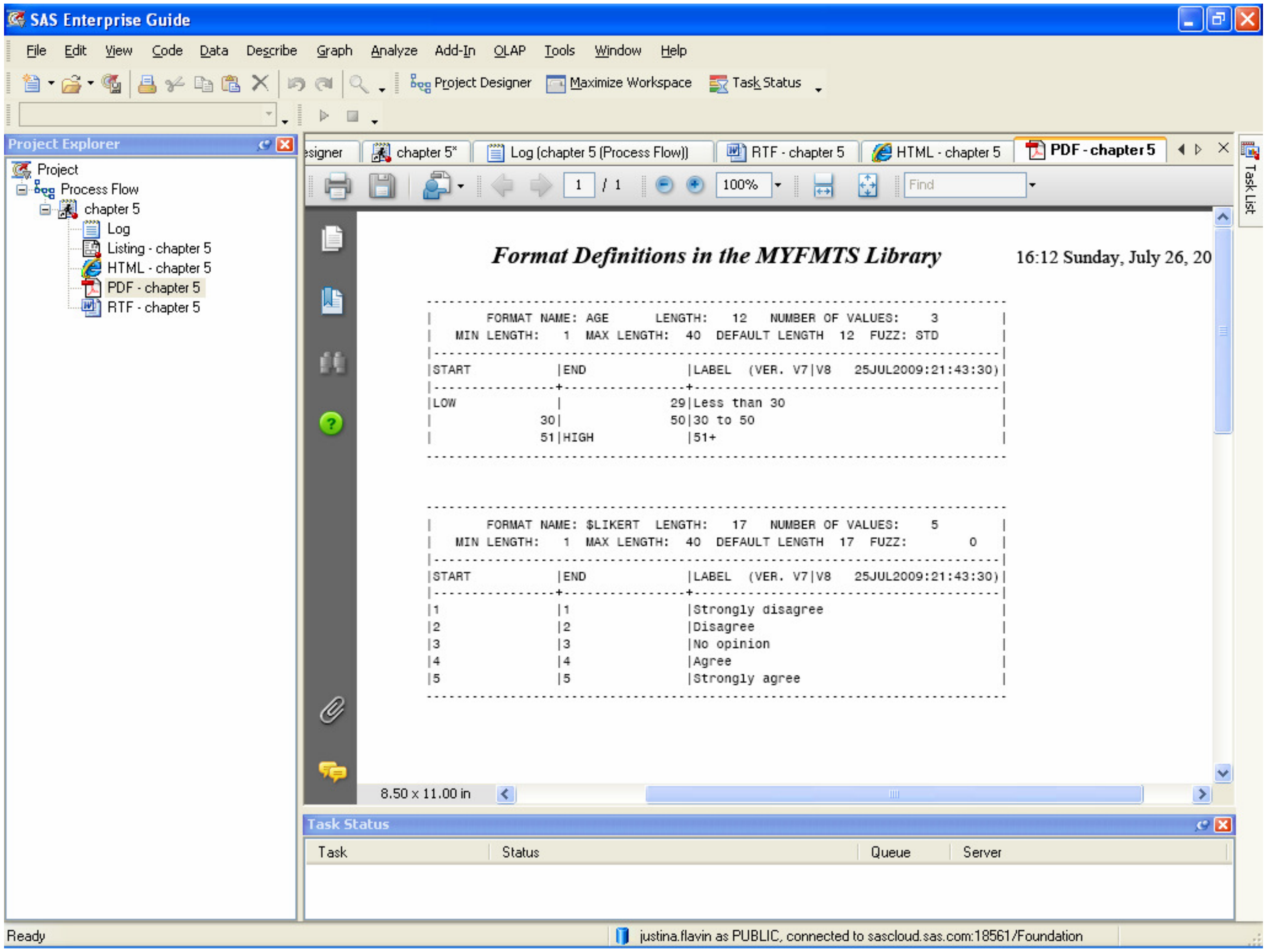
```
          Format Definitions in the MYFMTS Library          16:12 Sunday, July 26, 2

 ----------------------------------------------------------------------------
 |       FORMAT NAME: AGE      LENGTH:   12   NUMBER OF VALUES:    3        |
 |   MIN LENGTH:   1  MAX LENGTH:  40  DEFAULT LENGTH  12  FUZZ: STD        |
 |--------------------------------------------------------------------------|
 |START           |END             |LABEL  (VER. V7|V8   25JUL2009:21:43:30)|
 |----------------+----------------+----------------------------------------|
 |LOW             |              29|Less than 30                            |
 |              30|              50|30 to 50                                |
 |              51|HIGH            |51+                                     |
 ----------------------------------------------------------------------------


 ----------------------------------------------------------------------------
 |       FORMAT NAME: $GENDER  LENGTH:   11   NUMBER OF VALUES:    4        |
 |   MIN LENGTH:   1  MAX LENGTH:  40  DEFAULT LENGTH  11  FUZZ:        0   |
 |--------------------------------------------------------------------------|
 |START           |END             |LABEL  (VER. V7|V8   25JUL2009:21:43:30)|
 |----------------+----------------+----------------------------------------|
 |                |                |Not entered                             |
 |F               |F               |Female                                  |
 |M               |M               |Male                                    |
 |**OTHER**       |**OTHER**       |Miscoded                                |
 ----------------------------------------------------------------------------


 ----------------------------------------------------------------------------
 |       FORMAT NAME: $LIKERT  LENGTH:   17   NUMBER OF VALUES:    5        |
 |   MIN LENGTH:   1  MAX LENGTH:  40  DEFAULT LENGTH  17  FUZZ:        0   |
 |--------------------------------------------------------------------------|
 |START           |END             |LABEL  (VER. V7|V8   25JUL2009:21:43:30)|
 |----------------+----------------+----------------------------------------|
 |1               |1               |Strongly disagree                       |
 |2               |2               |Disagree                                |
 |3               |3               |No opinion                              |
 |4               |4               |Agree                                   |
 |5               |5               |Strongly agree                          |
 ----------------------------------------------------------------------------
```

Here is the output which displays only selected formats in the library:


SAS Enterprise Guide
File Edit View Code Data Describe Graph Analyze Add-In OLAP Tools Window Help
Project Designer Maximize Workspace Task Status
Project Explorer
Project
Process Flow
chapter 5
Log
Listing - chapter 5
HTML - chapter 5
PDF - chapter 5
RTF - chapter 5
signer chapter 5* Log (chapter 5 (Process Flow)) RTF - chapter 5 HTML - chapter 5 PDF - chapter 5
1 / 1 100% Find
8.50 x 11.00 in
Task Status
Task Status Queue Server
Ready
justina.flavin as PUBLIC, connected to sascloud.sas.com:18561/Foundation
Task List

Format Definitions in the MYFMTS Library 16:12 Sunday, July 26, 20

```
----------------------------------------------------------------------------
|       FORMAT NAME: AGE      LENGTH:   12   NUMBER OF VALUES:    3        |
|   MIN LENGTH:   1  MAX LENGTH:  40  DEFAULT LENGTH  12  FUZZ: STD        |
|--------------------------------------------------------------------------|
|START           |END             |LABEL  (VER. V7|V8   25JUL2009:21:43:30)|
|----------------+----------------+----------------------------------------|
|LOW             |              29|Less than 30                            |
|              30|              50|30 to 50                                |
|              51|HIGH            |51+                                     |
----------------------------------------------------------------------------


----------------------------------------------------------------------------
|       FORMAT NAME: $LIKERT  LENGTH:   17   NUMBER OF VALUES:    5        |
|   MIN LENGTH:   1  MAX LENGTH:  40  DEFAULT LENGTH  17  FUZZ:        0   |
|--------------------------------------------------------------------------|
|START           |END             |LABEL  (VER. V7|V8   25JUL2009:21:43:30)|
|----------------+----------------+----------------------------------------|
|1               |1               |Strongly disagree                       |
|2               |2               |Disagree                                |
|3               |3               |No opinion                              |
|4               |4               |Agree                                   |
|5               |5               |Strongly agree                          |
----------------------------------------------------------------------------
```