Making SAS Talk To Excel

Jack Hamilton

Kaiser Foundation Health Plan
Oakland, California
jack.hamilton@kp.org
jfh@alumni.stanford.org

Overview

- I plan to give a general discussion of the ExcelXP tagset, followed by a "cookbook" showing how to achieve various effects.

- There will be a few technical diversions, but in general this presentation is aimed at teaching by example, not theory.

- I don't expect you to remember the code. You can look that up. Just remember what you can do.

What do I mean by SAS?

- SAS 9.1.3.

What do I mean by Excel?

- Microsoft Excel XP, 2003, maybe some others I don't have around to test.
- OpenOffice.org 2.0 (not extensively tested, but I know that not everything works)
- ***Not*** Excel 5.
- Not, at the moment, Google spreadsheets. Maybe someday.

What do I mean by “talk”?

- I mean "create a file that Excel can open without having to do anything special, that will look like a regular Excel workbook, and that uses spreadsheet features such as formatting."

Why do you need to do this?

- Maybe you don't.
- But many organizations, from single-proprietor consultants to huge multi-national corporations, use Excel, or an equivalent, to work with sets of numbers. Google "What We Know About Spreadsheet Errors" for some numbers.

Why do you need to do this?

- If you have SAS, there's a good chance you will want to get numbers from SAS into Excel without a lot of fuss and bother.
- Excel is a good way to present data, but don't use it for statistics. Or programming. See the above-mentioned paper about spreadsheet errors. The chances that your manually-created spreadsheet has an error are very high.

Why do you need to do this?

- Use SAS to create your numbers in a structured, documented way (what SAS is good at), and use Excel to display the results to a wider audience (what Excel is good at).
- The ExcelXP tagset lets you create and distribute Excel reports with a minimum of human intervention.

What are some alternatives?

- The traditional approach was to create a flat file, with fixed columns or comma separated values, and use the Excel import wizard to read it in.
- This has worked for a very long time.
- You don't get any kind of formatting, and you get one worksheet, and it's a manual process.

What are some alternatives?

- Another alternative, in more recent versions of SAS, is PROC EXPORT. It's worth studying, but it is not as flexible as the ExcelXP tagset.

What are some alternatives?

- The SAS Business Intelligence product offers an Excel add-in that talks to a server. It looks good, but I haven't tried it. Also, it seems to require user interaction – it's a pull solution, not a push solution – and that might not meet your needs.
- There's also the MSOffice2K tagset, which offers a different set of options.

What are some alternatives?

- Some SAS products running under Windows can create Excel worksheets. For example, SAS for Windows and Enterprise Guide. The problem with all of those solutions is that they require either a Windows SAS license or some kind of server licensing. ExcelXP requires only base SAS, and it doesn't have to be on Windows.

What are some alternatives?

- Several years ago, I wrote a tagset that creates SYLK files. It's primitive, but if by some chance you want to import into Lotus 1-2-3 for DOS it might be the way to go.

What do I mean by Tagset?

- A tagset is a way to define data markup in SAS.
- "Markup" is a structured way to express logical and display characteristics of data. HTML is one type of markup you're probably familiar with.
- Other examples of markup are Adobe PDF and RTF.

Some technical details

- Tagsets are created by PROC TEMPLATE and invoked by the Output Delivery System. There's a special tagset language.
- Tagsets are compiled and can be stored permanently, just like formats and compiled macros.

Some technical details

- You can create your own tagsets, or modify existing ones.
- You don't need to know anything about this unless you want to further customize your output. In the case of the ExcelXP tagset, lots of options are built in, so you probably won't need to customize it.

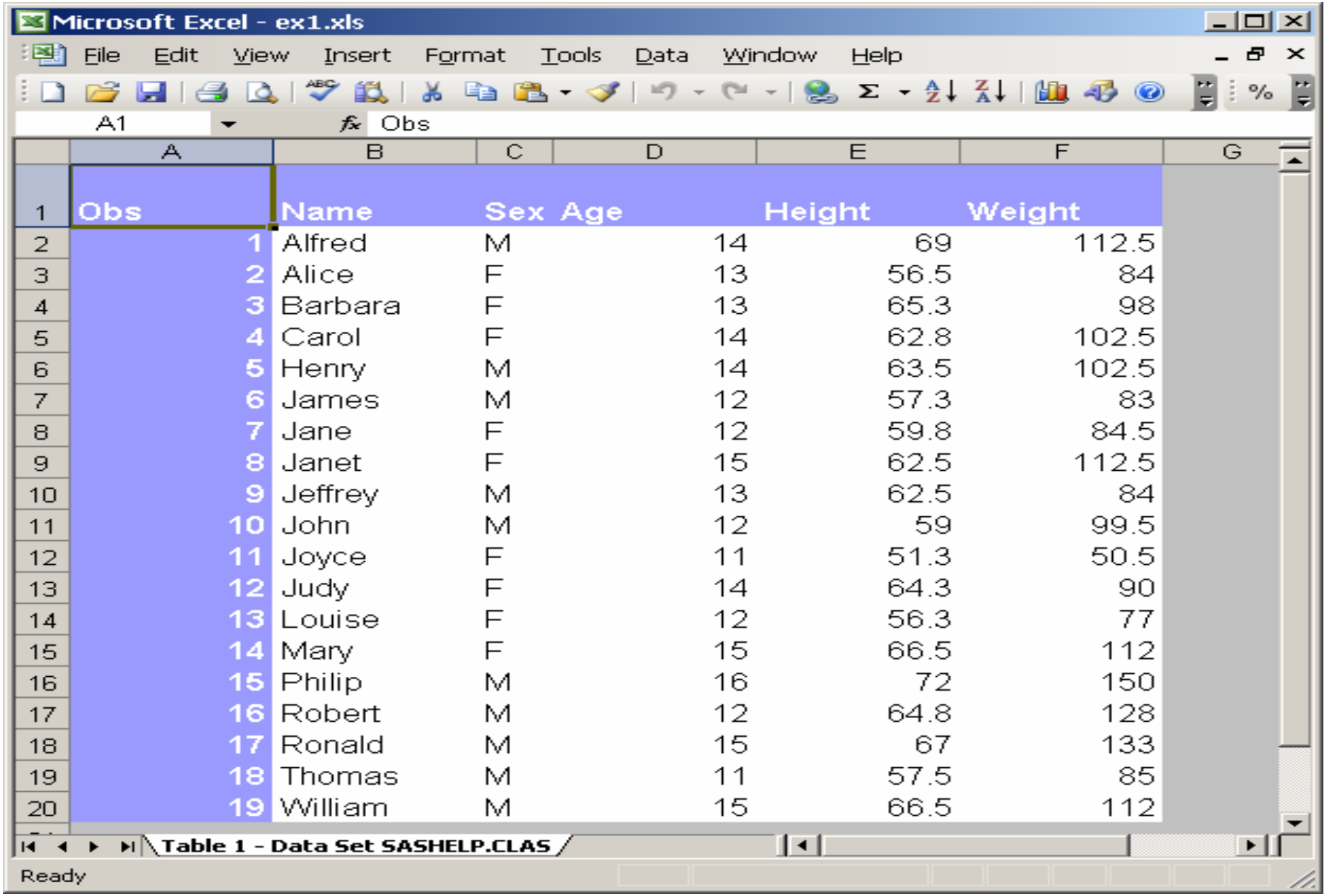
But Is It Really Excel?

- No
- It's XML (Extensible Markup Language), which recent versions of Excel know how to read. Other programs can read it too, such as OpenOffice.org.
- Because it's not really Excel, you should save it in native Excel format before sending it to someone who might not have a recent version of Excel.

A simple example

```
ods tagsets.excelxp
  file='ex1.xls';

proc print data=sashelp.class;
run;

ods tagsets.excelxp close;
```

A simple example



	A	B	C	D	E	F	G
1	Obs	Name	Sex	Age	Height	Weight	
2	1	Alfred	M	14	69	112.5	
3	2	Alice	F	13	56.5	84	
4	3	Barbara	F	13	65.3	98	
5	4	Carol	F	14	62.8	102.5	
6	5	Henry	M	14	63.5	102.5	
7	6	James	M	12	57.3	83	
8	7	Jane	F	12	59.8	84.5	
9	8	Janet	F	15	62.5	112.5	
10	9	Jeffrey	M	13	62.5	84	
11	10	John	M	12	59	99.5	
12	11	Joyce	F	11	51.3	50.5	
13	12	Judy	F	14	64.3	90	
14	13	Louise	F	12	56.3	77	
15	14	Mary	F	15	66.5	112	
16	15	Philip	M	16	72	150	
17	16	Robert	M	12	64.8	128	
18	17	Ronald	M	15	67	133	
19	18	Thomas	M	11	57.5	85	
20	19	William	M	15	66.5	112	

Making the ExcelXP tagset available

- It's unlikely that you have the latest version of the tagset available by default – it doesn't follow the same release schedule as base SAS. You can:
- Download the source once, save it, compile it, and save the results in a permanent location, or

Making the ExcelXP tagset available

- Download the source once, save it, and compile it in each job, or
- Download the source from the web and compile it in each job.

Making the ExcelXP tagset available

- The source can be found at

  `http://support.sas.com:80/rnd/base/topics/odsmarkup/excltags.tpl`

- Compiled versions are not distributed.

Making the ExcelXP tagset available

- To download and compile it each time, use this code:

```
filename tagset http
  "http://support.sas.com:80/
    rnd/base/topics/odsmarkup/
    excltags.tpl";
  %include tagset / nosource2;
  filename tagset clear;
```

- You must have an internet connection for this to work.

Making the ExcelXP tagset available

- Explaining how to save the compiled version is beyond the scope of this presentation. Ask your local SAS guru.

Using the ExcelXP tagset

- Use an ODS statement to open a tagset for output. The first time, specify FILE=.
- Each subsequent ODS statement which applies to the same workbook should not specify FILE=.
- One or more worksheets will be written for each proc or data step which creates output.

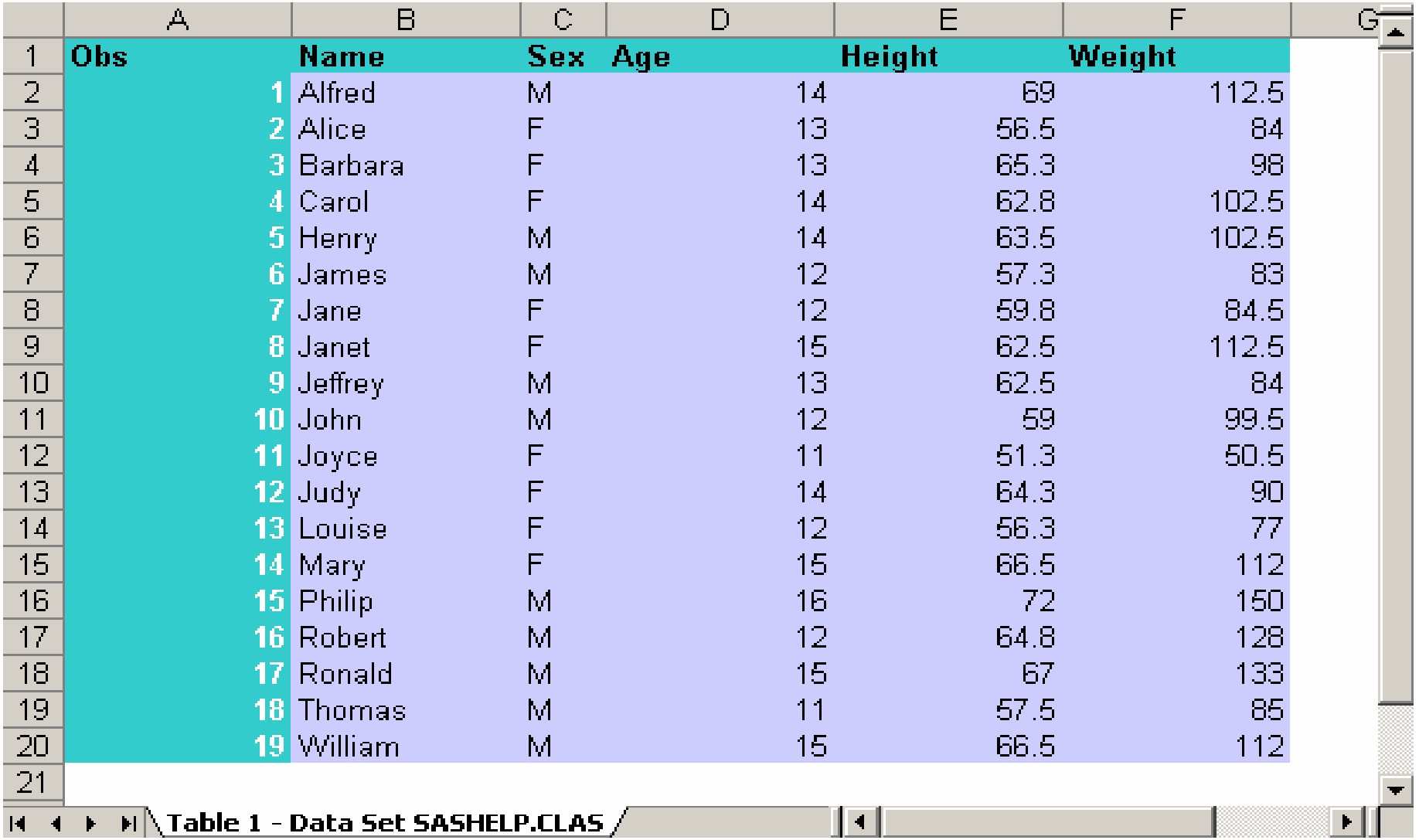
Using the ExcelXP tagset

- When finished, close the tagset.

Styles

- The ExcelXP tagset will use ODS styles to format your output. There are lots of prefined styles, or you can roll your own.
- Don't get carried away with lots of different fonts and colors.
- The default's not bad.
- Many procedures let you specify the style of individual elements.

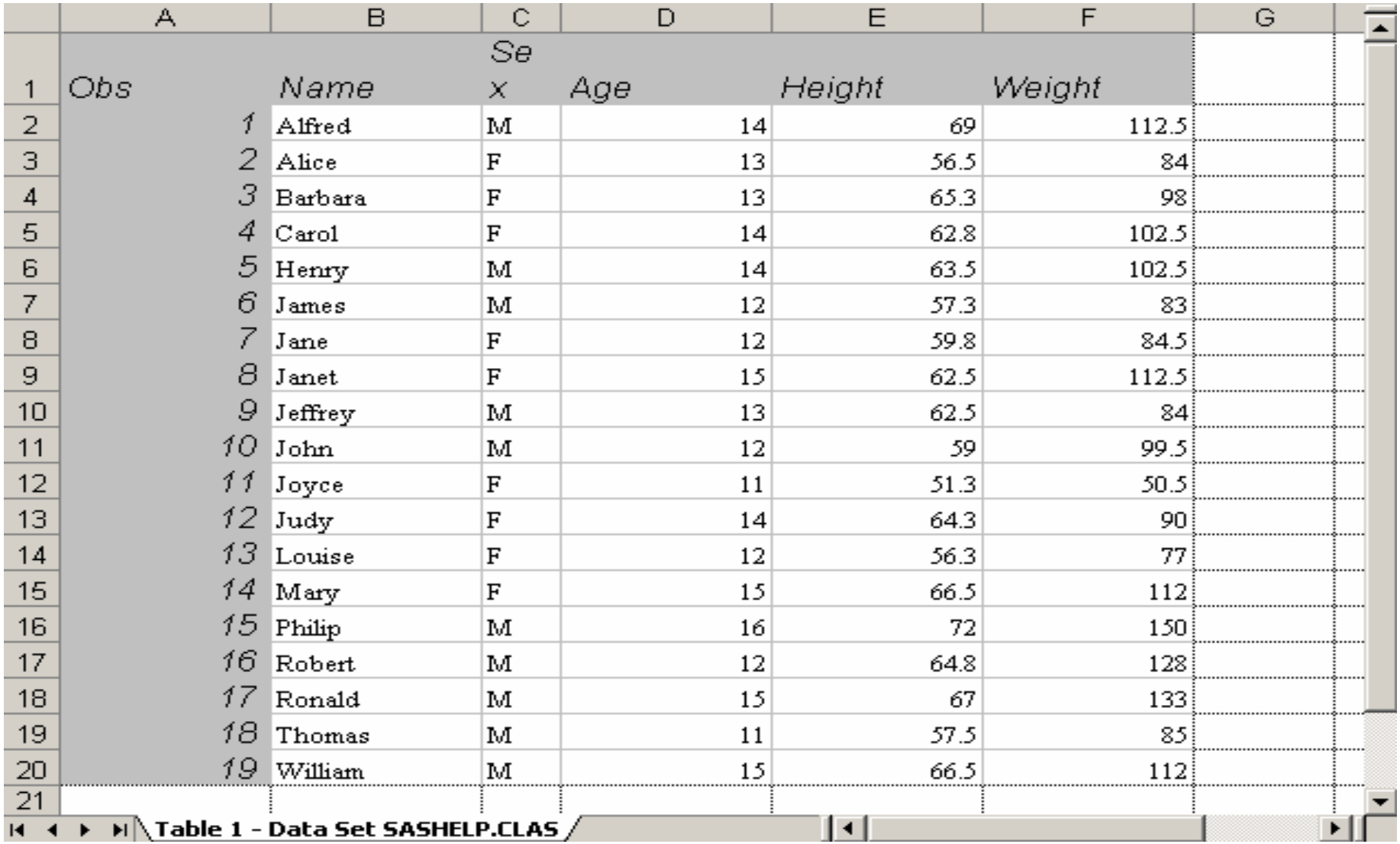
The barrettsblue style



Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84
10	John	M	12	59	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90
13	Louise	F	12	56.3	77
14	Mary	F	15	66.5	112
15	Philip	M	16	72	150
16	Robert	M	12	64.8	128
17	Ronald	M	15	67	133
18	Thomas	M	11	57.5	85
19	William	M	15	66.5	112

Table 1 - Data Set SASHELP.CLAS

The fancyprinter style



	A	B	C	D	E	F	G
1	Obs	Name	Sex	Age	Height	Weight	
2	1	Alfred	M	14	69	112.5	
3	2	Alice	F	13	56.5	84	
4	3	Barbara	F	13	65.3	98	
5	4	Carol	F	14	62.8	102.5	
6	5	Henry	M	14	63.5	102.5	
7	6	James	M	12	57.3	83	
8	7	Jane	F	12	59.8	84.5	
9	8	Janet	F	15	62.5	112.5	
10	9	Jeffrey	M	13	62.5	84	
11	10	John	M	12	59	99.5	
12	11	Joyce	F	11	51.3	50.5	
13	12	Judy	F	14	64.3	90	
14	13	Louise	F	12	56.3	77	
15	14	Mary	F	15	66.5	112	
16	15	Philip	M	16	72	150	
17	16	Robert	M	12	64.8	128	
18	17	Ronald	M	15	67	133	
19	18	Thomas	M	11	57.5	85	
20	19	William	M	15	66.5	112	
21							

Table 1 - Data Set SASHELP.CLAS

ExcelXP Options

- The ExcelXP tagset accepts a number of options, which affect how cells, rows, and columns appear, when new worksheets are created, and how the print output looks, and a few miscellaneous items.
- `ods tagsets.excelxp options(…)`

The first option you'll want to use

```
ods tagsets.excelxp
options(doc='help')
file='ex2.xls';
proc print data=sashelp.class;
run;
ods tagsets.excelxp close;
```

The first option you'll want to use

```
367  ods tagsets.excelxp options(doc='help')
 file='ex4.xls';
NOTE: Writing TAGSETS.EXCELXP Body file: ex4.xls
================================================================
 ====================
The EXCELXP Tagset Help Text.

This Tagset/Destination creates Microsoft's spreadsheetML
 XML.
It is used specifically for importing data into Excel.

Each table will be placed in its own worksheet within a
 workbook.
This destination supports ODS styles, traffic lighting,
 and custom formats
```

Options affecting columns

These are what I use most often.

Your mileage may vary.

- autofilter
- width_fudge
- absolute_column_width
- frozen_headers

Autofilter

- ods tagsets.excelxp
  options(autofilter='all')
  file='ex5.xls';

- ods tagsets.excelxp
  options(**autofilter='2-4'**)
  file='ex6.xls';

Autofilter='all'

	A	B	C	D	E	F
1	Obs	Name	Se	Age	Height	Weight
2	1	Alfred	M	14	69	112.5
3	2	Alice	F	13	56.5	84
4	3	Barbara	F	13	65.3	98
5	4	Carol	F	14	62.8	102.5
6	5	Henry	M	14	63.5	102.5
7	6	James	M	12	57.3	83
8	7	Jane	F	12	59.8	84.5
9	8	Janet	F	15	62.5	112.5
10	9	Jeffrey	M	13	62.5	84
11	10	John	M	12	59	99.5
12	11	Joyce	F	11	51.3	50.5
13	12	Judy	F	14	64.3	90
14	13	Louise	F	12	56.3	77
15	14	Mary	F	15	66.5	112
16	15	Philip	M	16	72	150
17	16	Robert	M	12	64.8	128
18	17	Ronald	M	15	67	133
19	18	Thomas	M	11	57.5	85
20	19	William	M	15	66.5	112
21						

Table 1 - Data Set SASHELP.CLAS

Autofilter='2-4'

	A	B	C	D	E	F
1	Obs	Name	Se	Age	Height	Weight
2	1	Alfred	M	14	69	112.5
3	2	Alice	F	13	56.5	84
4	3	Barbara	F	13	65.3	98
5	4	Carol	F	14	62.8	102.5
6	5	Henry	M	14	63.5	102.5
7	6	James	M	12	57.3	83
8	7	Jane	F	12	59.8	84.5
9	8	Janet	F	15	62.5	112.5
10	9	Jeffrey	M	13	62.5	84
11	10	John	M	12	59	99.5
12	11	Joyce	F	11	51.3	50.5
13	12	Judy	F	14	64.3	90
14	13	Louise	F	12	56.3	77
15	14	Mary	F	15	66.5	112
16	15	Philip	M	16	72	150
17	16	Robert	M	12	64.8	128
18	17	Ronald	M	15	67	133
19	18	Thomas	M	11	57.5	85
20	19	William	M	15	66.5	112
21						

Table 1 - Data Set SASHELP.CLAS

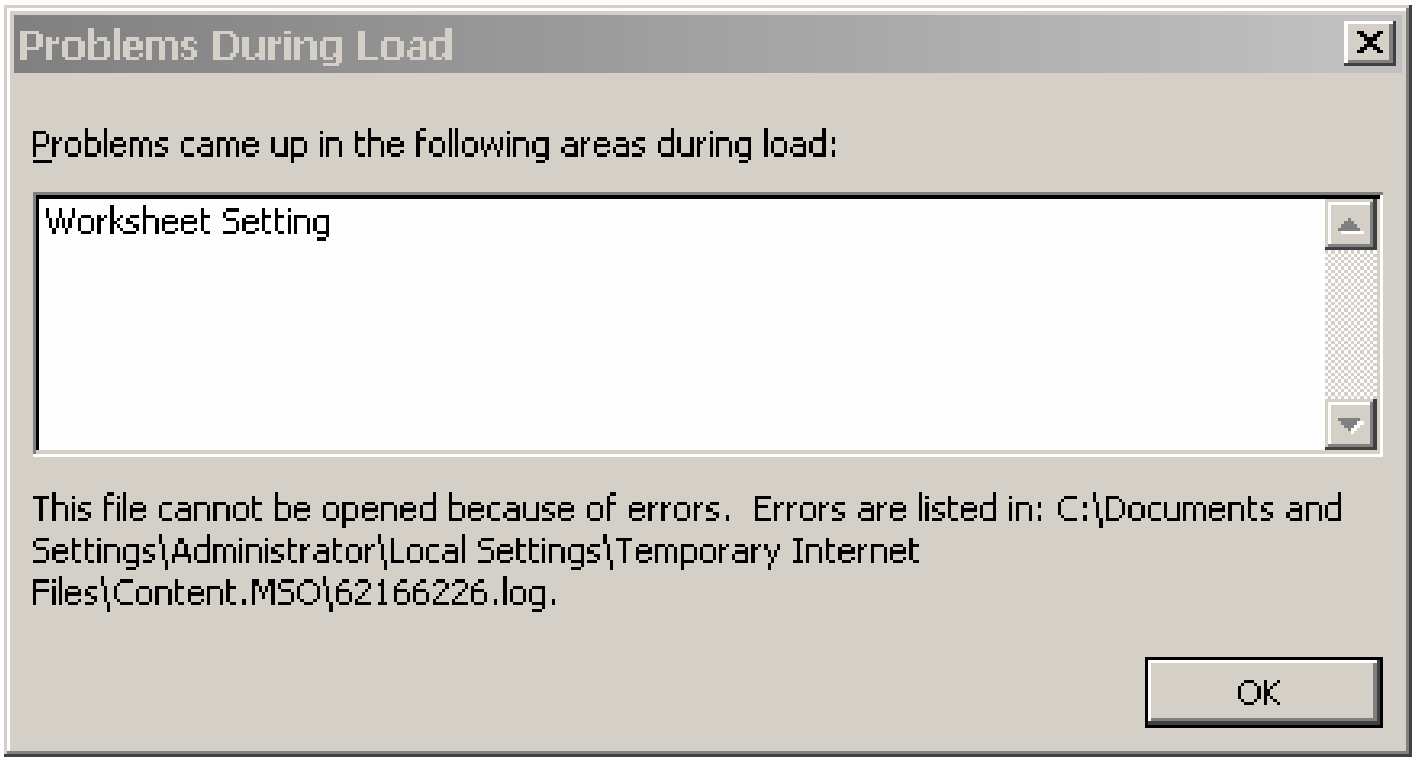
A diversion – Errors

- Suppose you tried this:

```
ods tagsets.excelxp
      options(autofilter='2,4')
      file='ex7.xls';
```

- This isn't valid, but the tagset doesn't detect that. It passes the erroneous specification into the XML, and you'll get a mysterious message when you try to open the file.

The Mysterious Error Message



- These errors can be hard to track down, so until you're accustomed to using the tagset, add features one at a time and test after every change.

Width_fudge

options(width_fudge='1')

	A	B	C	D	E	F
1	Obs	Name	Sex	Age	Height	Weight
2	1	Alfred	M	14	69	112.5
3	2	Alice	F	13	56.5	84
4	3	Barbara	F	13	65.3	98
5	4	Carol	F	14	62.8	102.5
6	5	Henry	M	14	63.5	102.5
7	6	James	M	12	57.3	83
8	7	Jane	F	12	59.8	84.5

Absolute_column_width

```
Options=(
  absolute_column_width='5,10,5,5,5,5'
  autofilter='2-4')
```

	A	B	C	D	E	F
1	Obs	Name	Sex	Age	Height	Weight
2	1	Alfred	M	14	69	112.5
3	2	Alice	F	13	56.5	84
4	3	Barbara	F	13	65.3	98
5	4	Carol	F	14	62.8	102.5

Frozen_headers

```
options(frozen_headers='1')
```

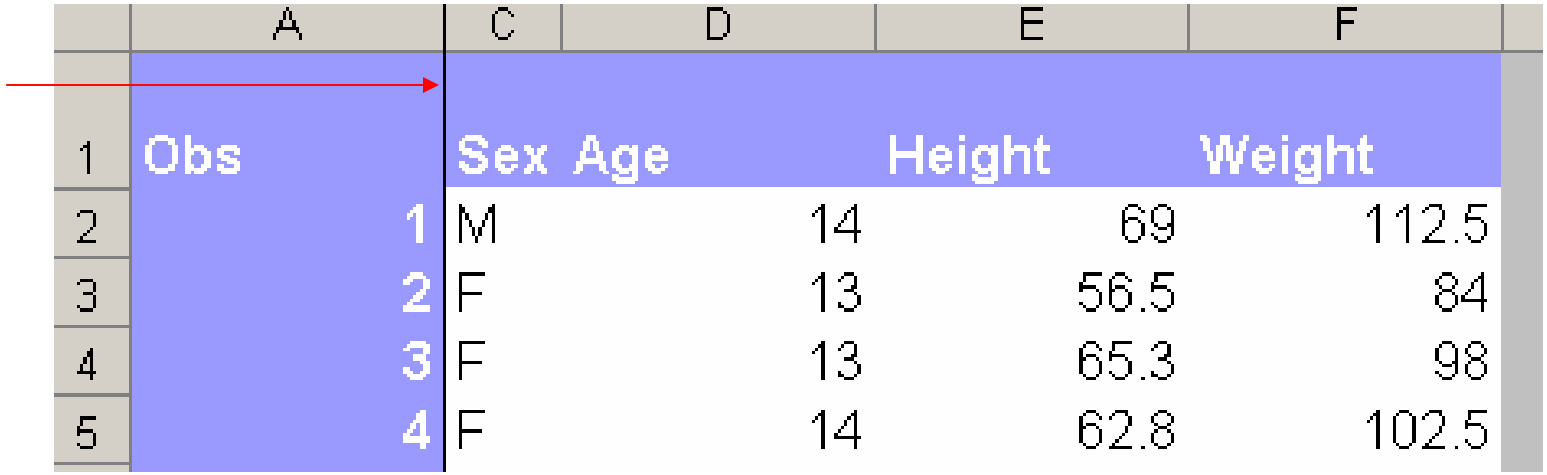
	A	B	C	D	E	F
1	Obs	Name	Sex	Age	Height	Weight
2	1	Alfred	M	14	69	112.5
3	2	Alice	F	13	56.5	84
4	3	Barbara	F	13	65.3	98

	A	B	C	D	E	F	G
1	Obs	Name	Sex	Age	Height	Weight	
18	17	Ronald	M	15	67	133	
19	18	Thomas	M	11	57.5	85	
20	19	William	M	15	66.5	112	
21							

Options affecting rows

```
options(frozen_rowheaders='1')
```

You can, and often will, specify both frozen_headers and frozen_rowheaders.



	A	C	D	E	F
1	Obs	Sex	Age	Height	Weight
2	1	M	14	69	112.5
3	2	F	13	56.5	84
4	3	F	13	65.3	98
5	4	F	14	62.8	102.5

Worksheet and Print options

Worksheet

- Sheet_name
- Embedded_titles

Print

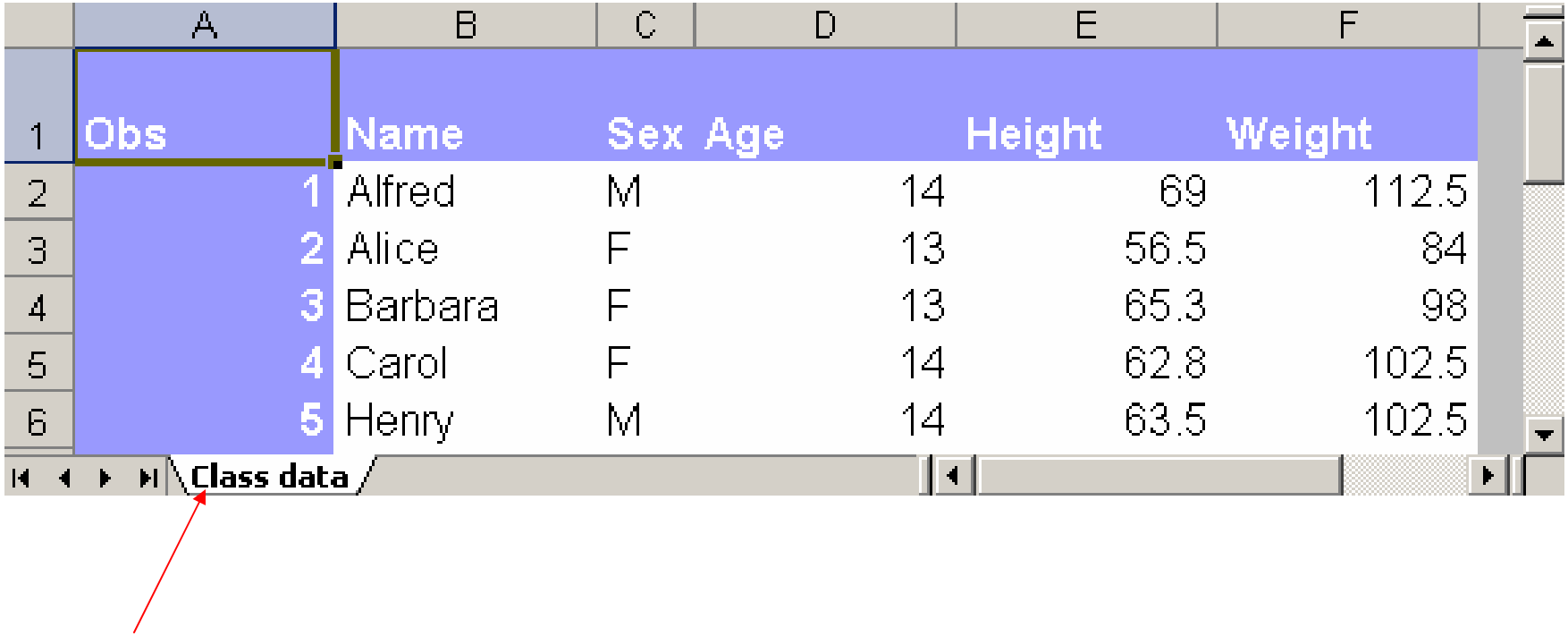
- Orientation
- Print_header / Print_footer
- Column_repeat / Row_repeat

Sheet_name

- This establishes the name of the current sheet. I almost always use it. The good folks at SAS have spend a lot of time trying to create a good algorithm for automatic sheet names, but I like my own better.
- Also look at the option Sheet_label.

Sheet_name

```
options(sheet_name='Class data')
```



	A	B	C	D	E	F
1	Obs	Name	Sex	Age	Height	Weight
2	1	Alfred	M	14	69	112.5
3	2	Alice	F	13	56.5	84
4	3	Barbara	F	13	65.3	98
5	4	Carol	F	14	62.8	102.5
6	5	Henry	M	14	63.5	102.5

Class data

Reusing sheet names

ExcelXP remembers options, so you'll want to specify a new sheet name for each procedure call. Two identical calls to PROC PRINT:

	A	B	C	D	E	F
1	Obs	Name	Sex	Age	Height	Weight
2	1	Alfred	M	14	69	112.5
3	2	Alice	F	13	56.5	84
4	3	Barbara	F	13	65.3	98
5	4	Carol	F	14	62.8	102.5
6	5	Henry	M	14	63.5	102.5

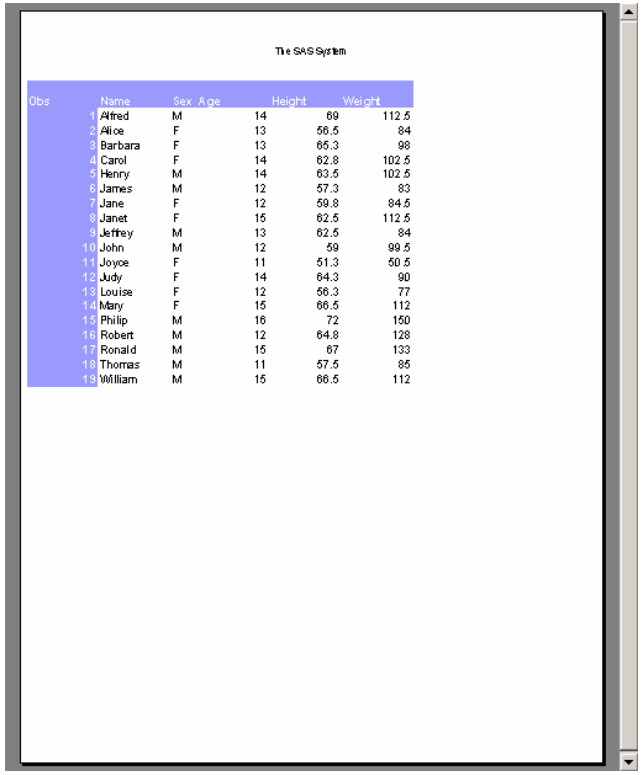
Class data | Class data 2

Orientation

- The results of this option are visible only when you print (or print preview the Excel workbook. Each sheet in a multiple sheet workbook can have a different orientation.
- The default orientation is Portrait, so you'll have to specify this option only if you want to use landscape, or earlier specified landscape and want to switch back to portrait.

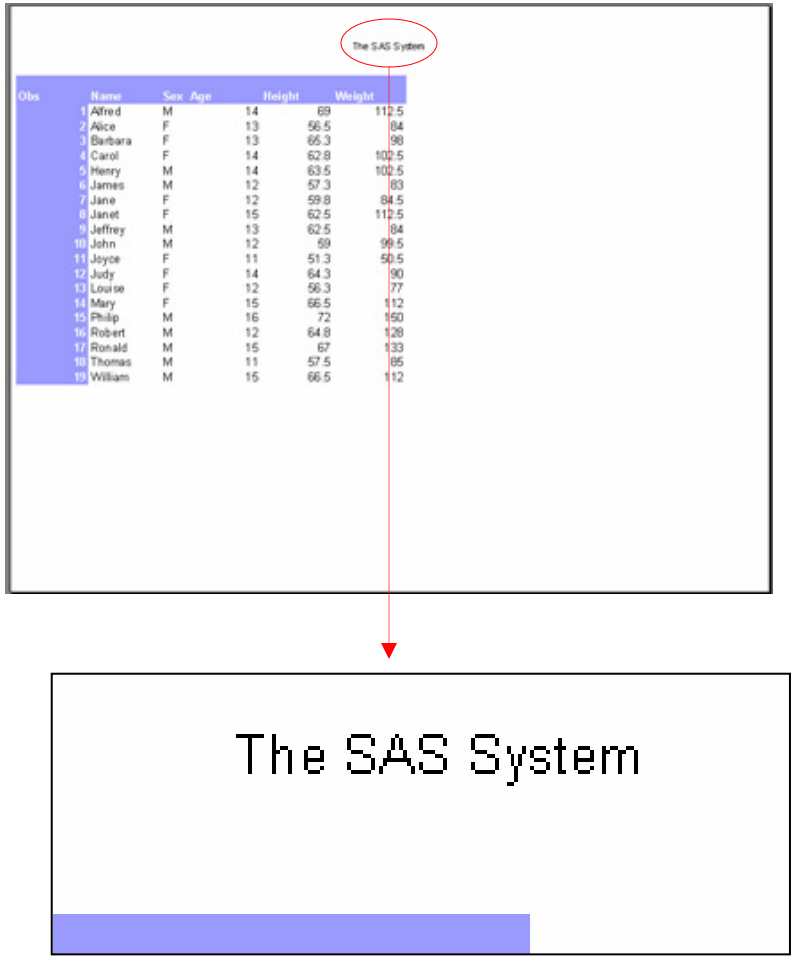
```
options(orientation='landscape')
```

Orientation



The SAS System

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84
10	John	M	12	59	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90
13	Louise	F	12	56.3	77
14	Mary	F	15	66.5	112
15	Philip	M	16	72	150
16	Robert	M	12	64.8	128
17	Ronald	M	15	67	133
18	Thomas	M	11	57.5	85
19	William	M	15	66.5	112

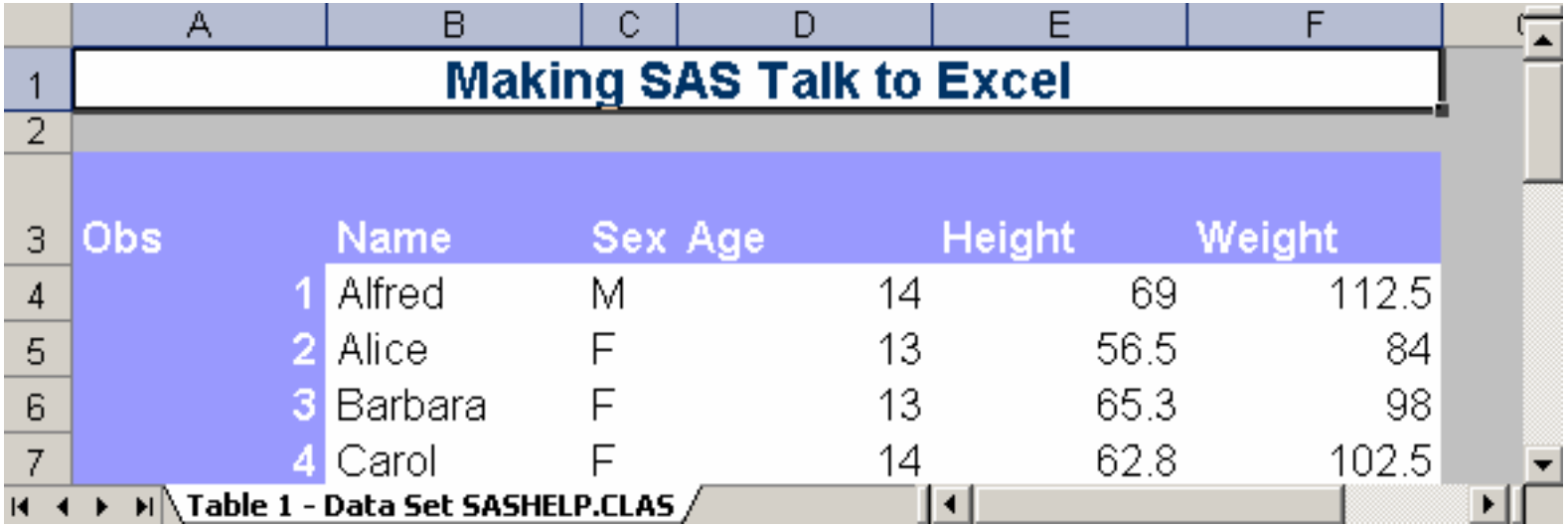


Embedded_titles

- As you saw in the previous example, the system title appears on the sheet when printed. If you also want it to appear in the normal view, use:
- `options(`**`embedded_titles='yes'`**`)`
- If you use this option, you will probably want to expand freeze_headers as well.

Embedded_titles

```
title 'Making SAS Talk to Excel';
 proc print data=sashelp.class;
 run;
```



	A	B	C	D	E	F
1	Making SAS Talk to Excel					
2						
3	Obs	Name	Sex	Age	Height	Weight
4	1	Alfred	M	14	69	112.5
5	2	Alice	F	13	56.5	84
6	3	Barbara	F	13	65.3	98
7	4	Carol	F	14	62.8	102.5

Table 1 - Data Set SASHELP.CLAS

Print_header / Print_footer

- This gives you complete control over the Excel page headers and footers. It uses the same syntax as Excel – it is just passed through. This is passed as XML, so you'll need to use character entities for special characters such as &.
- See the documentation for a very complicated example.

Print_header / Print_footer

```
options(print_footer='&amp;L&amp;D&amp;C&amp;A&amp;RPage &amp;P of &amp;N')
```

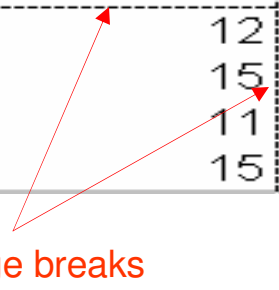
10/29/2006	Table 1 - Data Set SASHELP.CLAS	Page 1 of 1

Column_repeat / Row_repeat

- These options cause column and row headers to repeat across pages. If you have many rows or columns, you will probably want to use these options (or one of them).
- `options(column_repeat='1'`
  `row_repeat='1')`
- Page size set to A6

Column_repeat / Row_repeat

	A	B	C	D	E	F
1	Obs	Name	Sex	Age	Height	Weight
2	1	Alfred	M	14	69	112.5
3	2	Alice	F	13	56.5	84
4	3	Barbara	F	13	65.3	98
5	4	Carol	F	14	62.8	102.5
6	5	Henry	M	14	63.5	102.5
7	6	James	M	12	57.3	83
8	7	Jane	F	12	59.8	84.5
9	8	Janet	F	15	62.5	112.5
10	9	Jeffrey	M	13	62.5	84
11	10	John	M	12	59	99.5
12	11	Joyce	F	11	51.3	50.5
13	12	Judy	F	14	64.3	90
14	13	Louise	F	12	56.3	77
15	14	Mary	F	15	66.5	112
16	15	Philip	M	16	72	150
17	16	Robert	M	12	64.8	128
18	17	Ronald	M	15	67	133
19	18	Thomas	M	11	57.5	85
20	19	William	M	15	66.5	112



Column_repeat / Row_repeat

Making SAS Talk to Excel

Obs	Height	Weight
16	64.8	128
17	67	133
18	57.5	85
19	66.5	112

Multiple worksheets

- By default, you'll get a new worksheet for each procedure and each by-group within a procedure (just as you would get new pages in listing output).
- The interaction between procedures, by-groups, pages, and sheet labels is probably the most complicated aspect of the ExcelXP tagset.

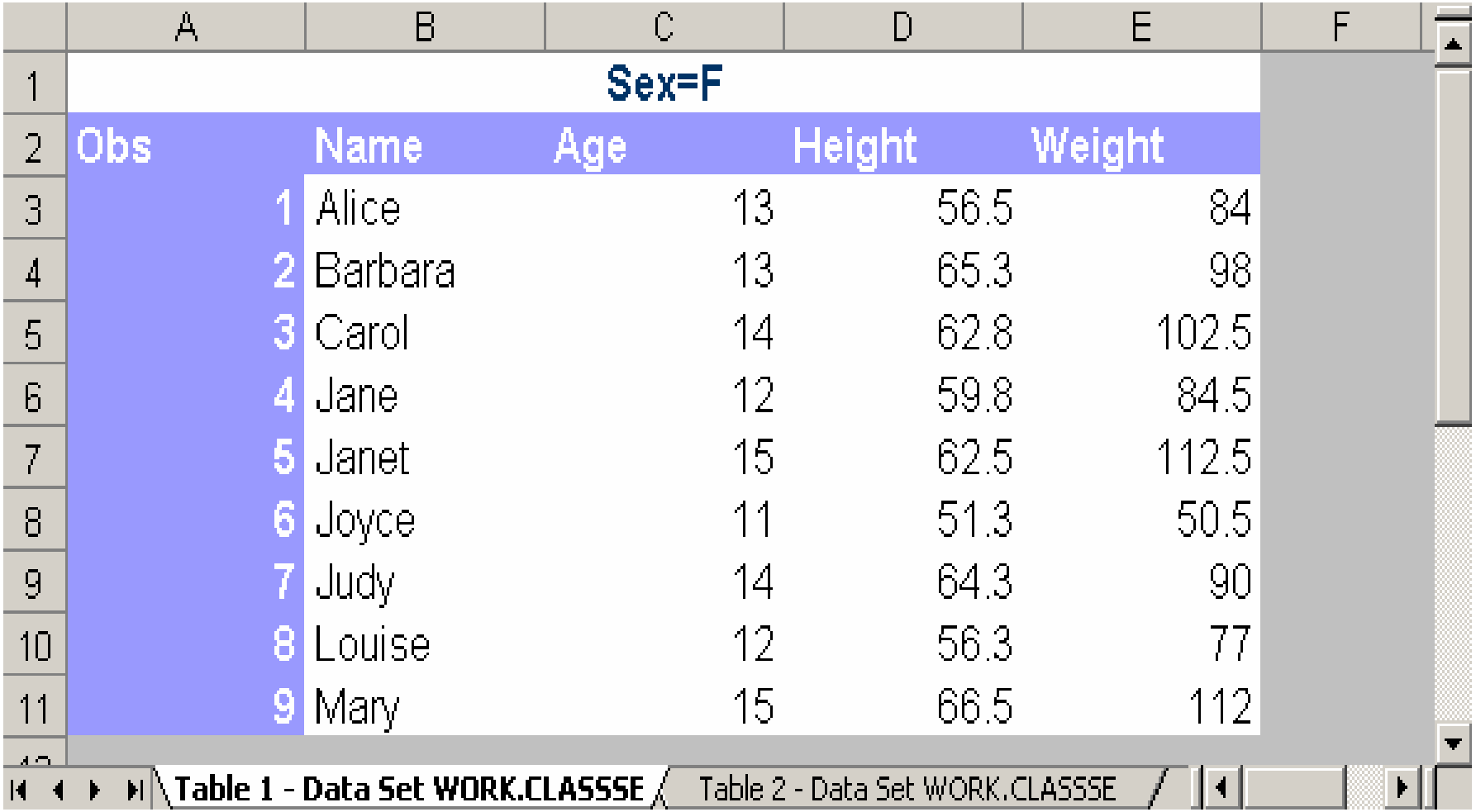
Multiple worksheets with BY-Groups

```
proc sort data=sashelp.class
          out=classsex;
   by sex;
run;

ods tagsets.excelxp
  file='ex18.xls';
```

Multiple worksheets with BY-Groups

```
proc print data=classsex;
    by sex;
run;

ods tagsets.excelxp close;
```

Multiple worksheets with BY-Groups



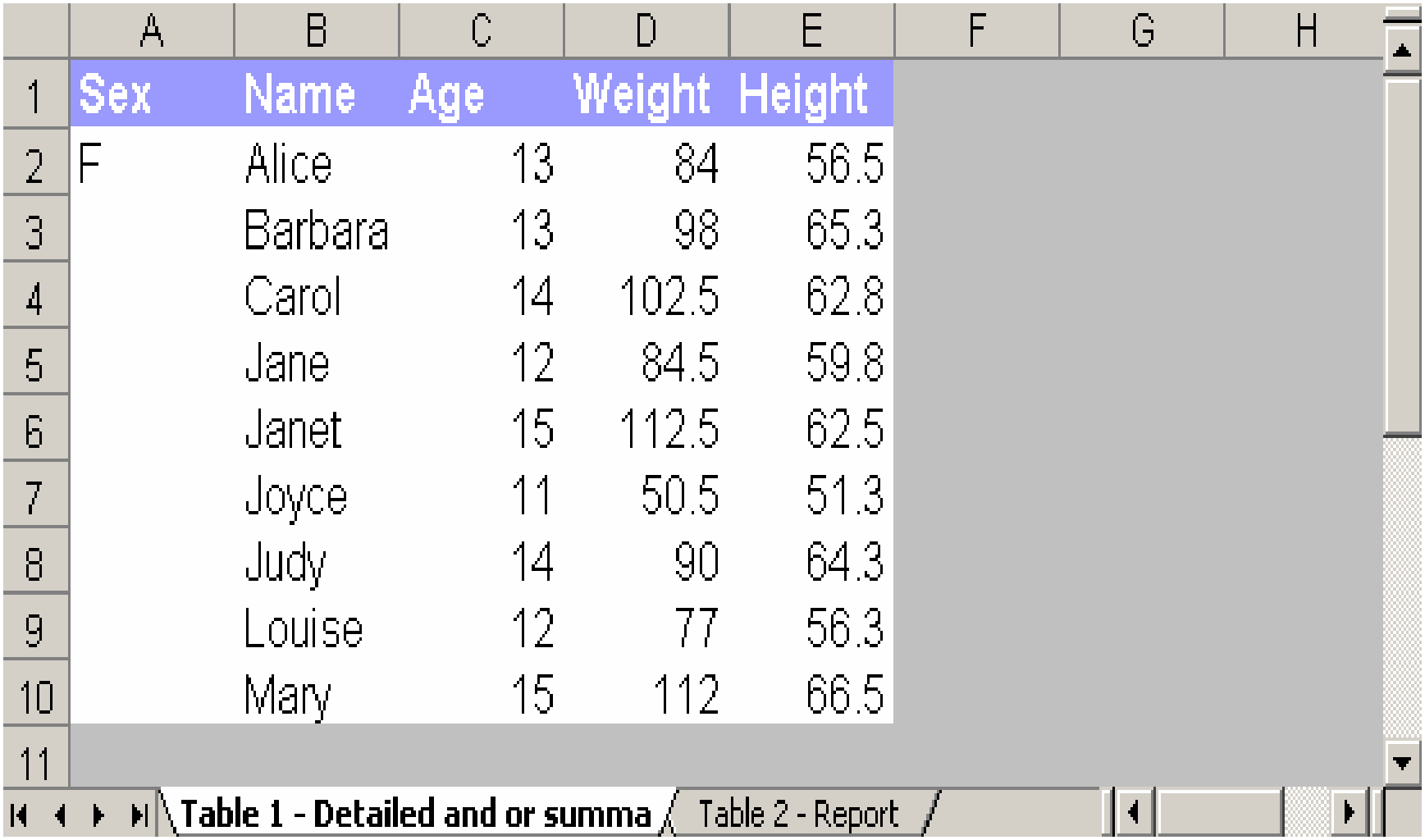
	A	B	C	D	E	F
1			Sex=F			
2	Obs	Name	Age	Height	Weight	
3	1	Alice	13	56.5	84	
4	2	Barbara	13	65.3	98	
5	3	Carol	14	62.8	102.5	
6	4	Jane	12	59.8	84.5	
7	5	Janet	15	62.5	112.5	
8	6	Joyce	11	51.3	50.5	
9	7	Judy	14	64.3	90	
10	8	Louise	12	56.3	77	
11	9	Mary	15	66.5	112	

Table 1 - Data Set WORK.CLASSSE | Table 2 - Data Set WORK.CLASSSE

Multiple worksheets with PROC REPORT

```
proc report data=sashelp.class
     nofs missing;
 column sex name age
        weight height;
   define sex / order;
   break after sex / page;
run;
```

Multiple worksheets with PROC REPORT



	A	B	C	D	E	F	G	H
1	Sex	Name	Age	Weight	Height			
2	F	Alice	13	84	56.5			
3		Barbara	13	98	65.3			
4		Carol	14	102.5	62.8			
5		Jane	12	84.5	59.8			
6		Janet	15	112.5	62.5			
7		Joyce	11	50.5	51.3			
8		Judy	14	90	64.3			
9		Louise	12	77	56.3			
10		Mary	15	112	66.5			
11								

Table 1 - Detailed and or summa | Table 2 - Report

Multiple Sheet Titles

- You may have noticed that the sheet titles weren't particularly satisfactory. This is a topic of ongoing debate, and I expect improvements in future releases.
- I usually run a separate proc step for each BY value using a WHERE clause, and specify the sheet_name option.
- This works for me because I have data sets with a small number of known values for my groups. It might not work for you.

An Alternative Way To Set Titles

- I created a modified version of the ExcelXP tagset which obtains the sheet titles from a macro variable. I then use PROC REPORT to set the macro variable whenever it creates a new page.
- I haven't included it here, but I will put in the file I mail out.
- Not supported by SAS, not supported by me.

Other Options

- There are numerous other options.

  Explaining them would make me run over my time limit Look at the documentation, and read the ODS community forum found through support.sas.com.

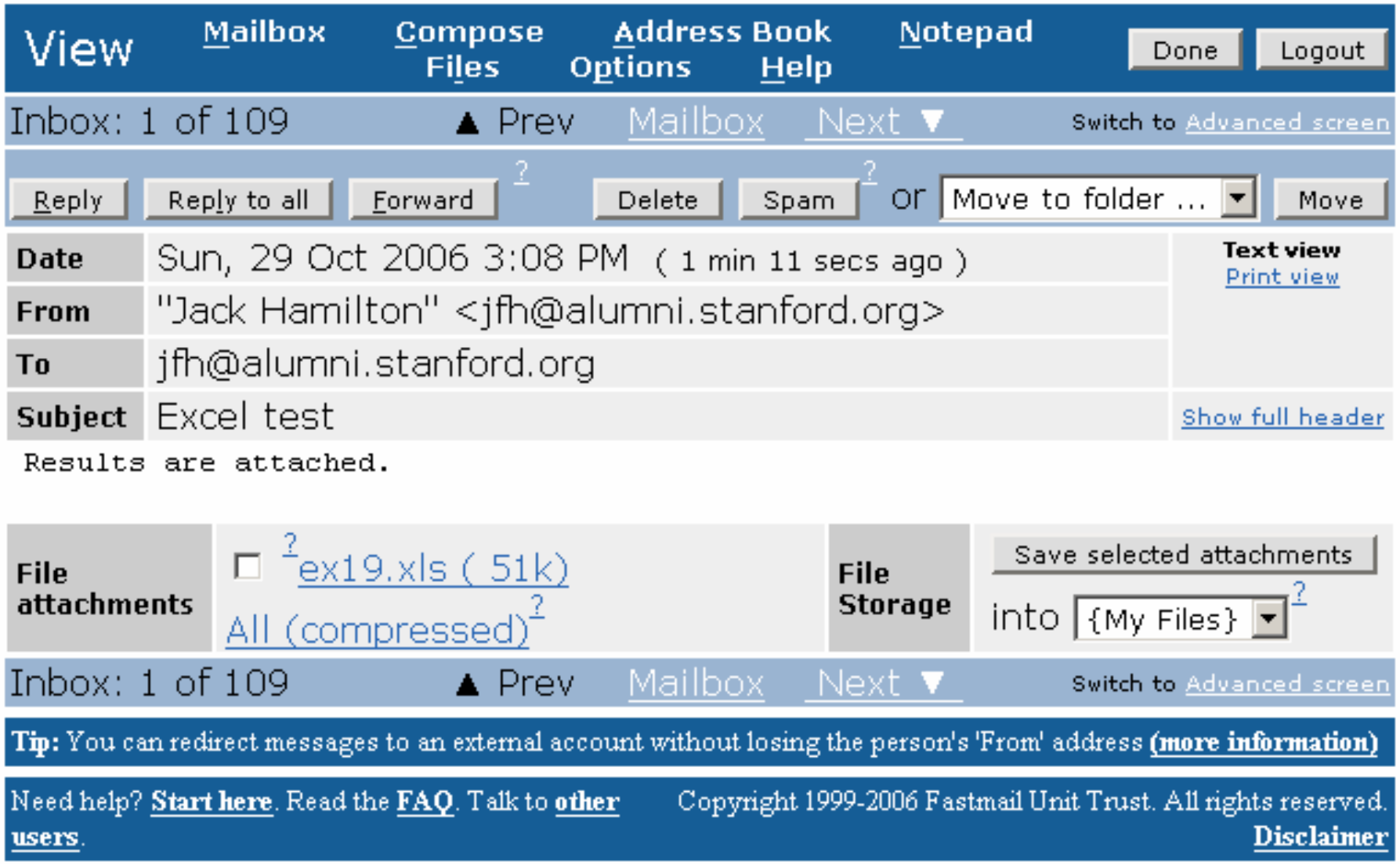
Mailing The Results

- I mail the results to myself if I am running SAS under MVS (the same thing would apply if I were running under Unix). This saves me from having to do an FTP.

Mailing The Results

```
filename email email
                attach=("ex19.xls")
                to='jfh@alumni.stanford.org'

  from='jfh@alumni.stanford.org'
                subject='Excel test ';

data _null_;
   file email;
   put 'Results are attached.';
run;
```

Mailing The Results



Further Documentation

- http://support.sas.com/rnd/base/topics/odsmarkup/excelxp_demo.html

- Contains some good examples, and covers options I didn't discuss.

Further Documentation

- http://support.sas.com/rnd/base/topics/odsmarkup/
- http://support.sas.com/rnd/base/topics/odsmarkup/excelxp_demo.html
- Many SUGI papers can be found at http://www.lexjansen.com/sugi/index.htm