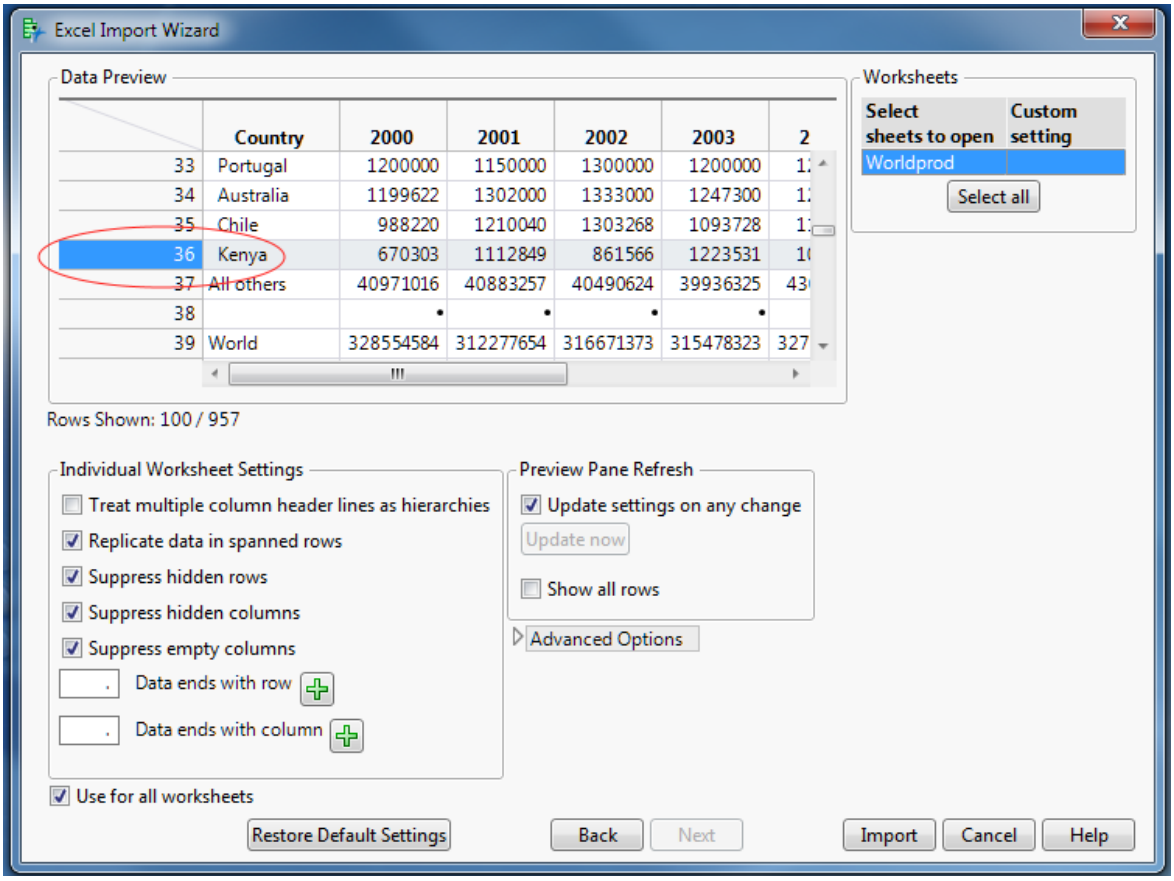
Working with Excel – The Advanced Edition

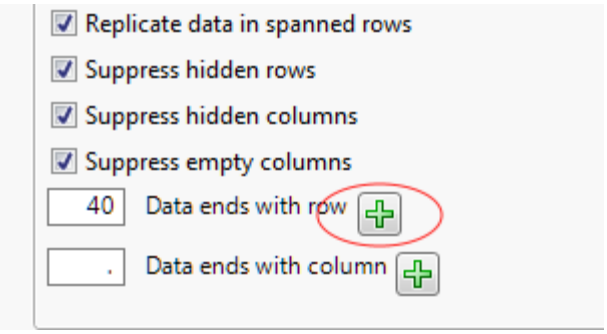
JMP Discovery Conference 2016
Brian Corcoran – SAS Institute

In version 11, JMP Development introduced the Excel Wizard for the Windows product. This was followed by a version for the Mac in JMP 12. The feature has proven to be extremely popular, and the addition of a variety of new capabilities since version 11 makes it an appropriate time to revisit the Wizard. Most of this paper will be devoted to the Advanced Options that become available in version 13. However, first it is worth noting a few changes that were made to version 12 of JMP.

Specifying the starting and ending rows, in particular the ending rows, can be cumbersome. Version 11 of the Wizard required the manual entry of a valid row number for the ending row if you didn't want to scan to the end of the worksheet. This could get confusing, because you have to add in the number of rows before the start of the data, and the number of hidden rows that might occur before the end of the data. Version 12 introduced green plus icons for all of the starting and ending type fields. All you need to do is highlight a row within the data preview, and then press the green **+**. JMP will calculate the row number within the worksheet, and put that row number into the edit field. The spin buttons still remain for those who prefer that method.

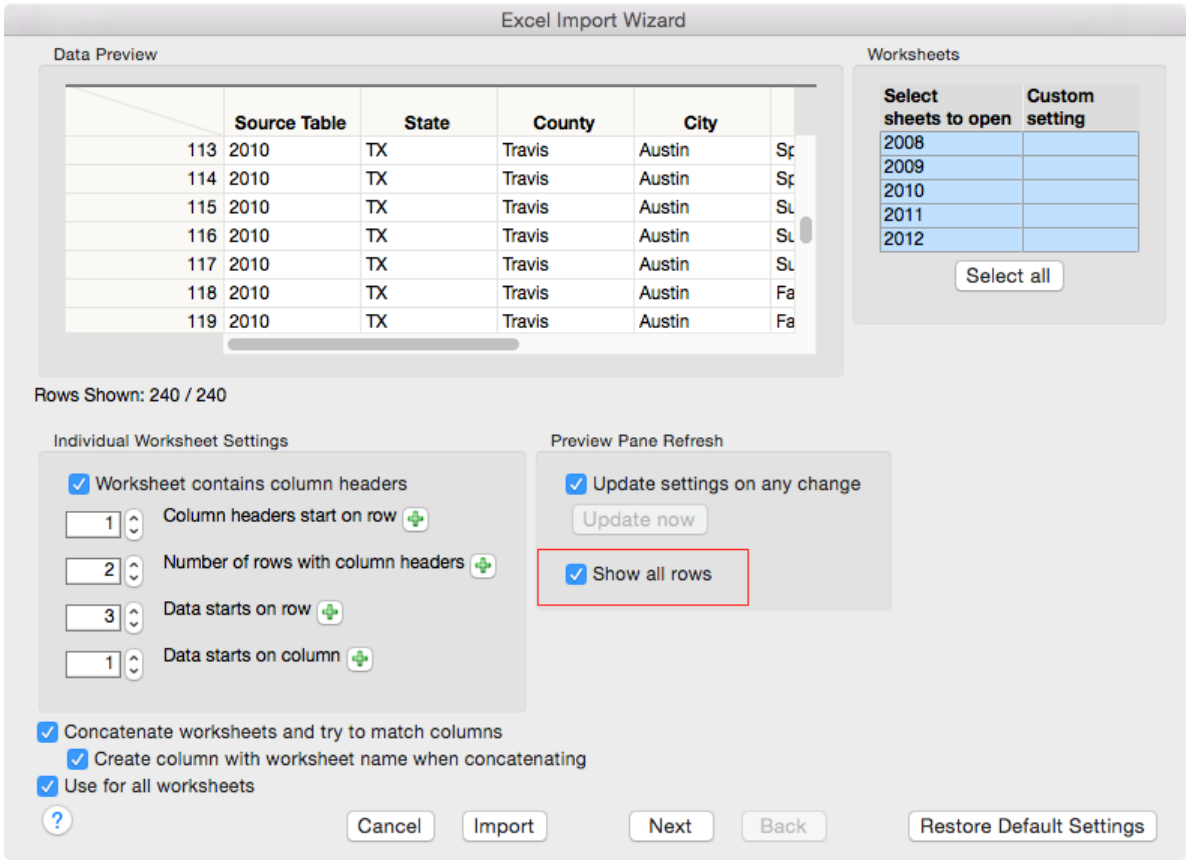


Data ends example – Highlight the ending row



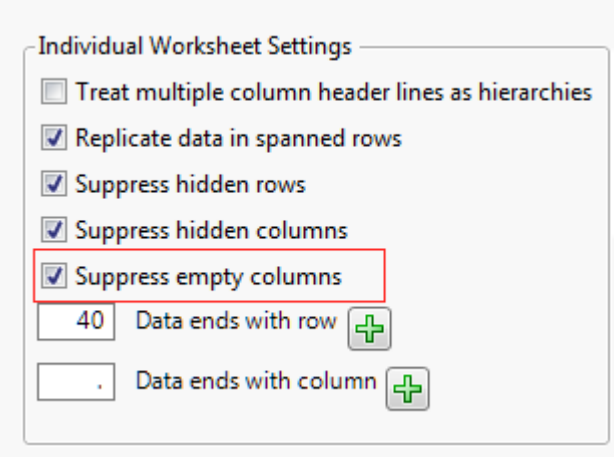
Data ends example – Press the green plus button

The **Show all rows** option was also added in version 12. By default, the preview window shows the first 100 rows of data in the table, if there are more than 100 rows. Some users have said that they would like to see all of the data, so the option for **Show all rows** was added. When you select this, all of the rows in the table will be shown, and all of the rows will be used to determine the data type for the column as shown in the preview. Please note that by default, all rows are used to determine the column data type when the actual import is done. The preview limits the number of rows shown by default to make the performance better when making changes to the wizard.



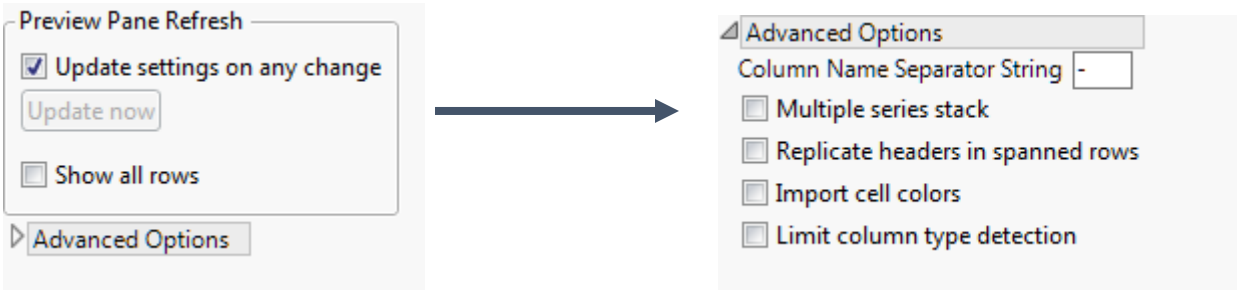
Show all rows option

The final version 12 feature to mention is the **Suppress empty columns** option. Sometimes workbooks contain columns that are empty, but the column header itself conveys important information. By unchecking the **Suppress empty columns** option, the empty column will be imported. By default, JMP tries to remove this type of column. This option is on the second page of the Wizard.



Suppress empty columns

The JMP development team constantly accumulates and evaluates customer requests. A variety of requests were received for the Excel Wizard, and as certain themes evolved we considered how best to integrate those feature requests into the existing Wizard. The requests were uniformly of interest to a select audience, which hopefully means the original dialog got most of the common features correct. The decision was made to provide an Advanced Options panel. This panel appears on the second page of the Wizard, and by default is not shown. When unfolded, it will be reshown in subsequent invocations if any of the options are selected. If no option is selected, it will be hidden on subsequent invocations. The goal here is to prevent the options from cluttering the interface.



Advanced Options Folded **Advanced Options Unfolded**

The first option, **Column Name Separator String**, is the easiest to explain. In a situation where a column name spans several rows within Excel, we concatenate the row strings together with a dash in between. Here is an example of a situation where you might want both rows of the Excel header:

Winter		
Feb	Mar	Apr
0.2	1.82	0.83

The column headers imported into JMP will be “Winter-Feb”, “Winter-Mar” and “Winter-Apr”. The **Column Name Separator String** option allows you to change that dash to whatever character or string that you would like. If you want the strings concatenated with no separator, delete the dash in the edit box. If you would like a space or spaces, put those in with the space key. Any change that you make will be immediately reflected in the preview pane.

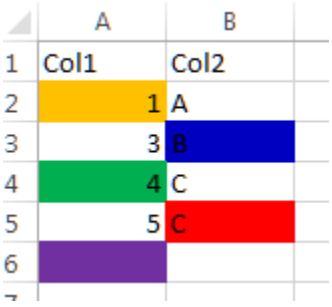
The option **Replicate headers in spanned rows** is likely to have a limited audience, but would be used when you have a worksheet with column headers that resemble the following configuration:

A	B	C	D		
			1	2	3
X	50.22	M	Joe	21	O
	24.34	F	Mary	22	P
	25.54	F	Cindy	23	P
Y	52.11	M	Mark	22	O
	43.32	F	Bill	23	P
	11.23	F	Jennifer	24	Q

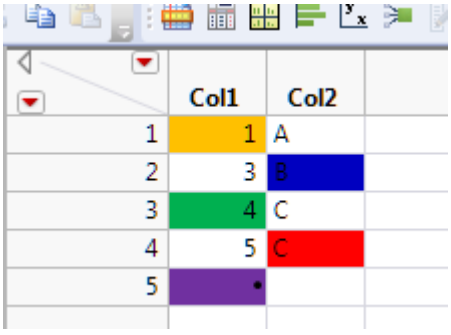
Headers with different row spans

When **Number of Rows with column headers** is set to 2, the default operation for JMP will be to import the column names as “A”, “B”, “C”, “D-1”, “D-2” and “D-3”. In some usage cases, a consistency in formatting is important. When you select the **Replicate headers in spanned rows** option, the column names will be imported as “A-A”, “B-B”, “C-C”, “D-1” and so on. The character in the spanned/merged columns is replicated as many times as necessary to match the formatting of the column names where there isn’t a span.

The **Import cell colors** option is self-explanatory. When checked, JMP will attempt to set the cell color for the imported data to match the cell color within Excel. The match may not be exact. On the Mac, cell colors are limited to primary and secondary colors due to a limitation in the library that is being used to read the data. Using this option with huge worksheets may slow the import performance.



Cell colors in Excel



Cell colors imported into JMP

The **Limit column type detection** option gives users control over how JMP determines the column data type. As mentioned earlier, when doing the preview in the Wizard, JMP examines the first 100 rows of data to determine the column data type. This is for performance reasons. When doing the actual import, JMP 13 considers all of the data in the row to avoid creating missing values when character data is encountered far down in the column. Basically any kind of character data that is detected in a column will cause the column type to become Character. If **ALL** of the data is of the same type, say numeric date data with the format M/D/Y, then the column will be typed as such. The difference in criteria can occasionally cause a difference in the appearance of the preview and the result table, as extraneous character date encountered late in the import can cause the imported column type to be set to Character. Using the previously mentioned **Show all rows** option will cause the behavior to be the same between preview and import, at the cost of performance.

So what about the opposite case? What happens when the table is so huge that the import takes forever? Much of the performance cost during import is because of the type checking. The **Limit column type detection** option offers an alternative. When checked, JMP will only examine the first 100 rows of data to determine the column data type during import. This can dramatically speed up the import results, at the risk of losing character data in an otherwise numeric column.

In the example below, the preview indicates that we will get 3 numeric columns for our automotive data.

Data Preview

	Model Yr	Mfr Name	Eng Displ	# Cyl		
94	2012	BMW	1.6	4		
95	2012	BMW	1.6	4		
96	2012	BMW	1.6	4		
97	2012	BMW	1.6	4		
98	2012	BMW	1.6	4		
99	2012	BMW	1.6	4		
100	2012	BMW	1.6	4		

However, upon import the final column is typed as Character.

	Model Yr	Mfr Name	Eng Displ	# Cyl
1	2012	aston martin	5.9	12
2	2012	aston martin	4.7	8
3	2012	aston martin	4.7	8
4	2012	aston martin	4.7	8
5	2012	aston martin	4.7	8
6	2012	Audi	4.2	8
7	2012	Audi	4.2	8

If we use **Show all rows** in the Wizard, we can examine the data to see the issue and find that somebody has entered V4 and V6 for engine sizes.

Data Preview

	Model Yr	Mfr Name	Eng Displ	# Cyl
99	2012	BMW	1.6	4
100	2012	BMW	1.6	4
101	2012	Mitsubishi ...	2.4	V4
102	2012	Mitsubishi ...	3.8	V6
103	2012	Porsche	3.8	6
104	2012	Porsche	3.8	6
105	2012	Porsche	3.8	6
106	2012	Porsche	3.8	6

In this case, we might want to preserve the data and clean or recode it within JMP. But, what if column ends with something like "Total" or an extraneous attribution to a source? This might not be worth keeping, or paying a performance penalty to detect. For worksheets that contains hundreds of thousands of rows, it is not uncommon to reduce the import time by 75% by checking **Limit column type detection**. This is a powerful option if you really understand your data. Just know that in the case above, if we had checked this option, our resulting data would look like:

	Model Yr	Mfr Name	Eng Displ	# Cyl
99	2012	BMW	1.6	4
100	2012	BMW	1.6	4
101	2012	Mitsubishi Motor...	2.4	•
102	2012	Mitsubishi Motor...	3.8	•
103	2012	Porsche	3.8	6
104	2012	Porsche	3.8	6
105	2012	Porsche	3.8	6

The final option is **Multiple series stack**, and it is easily the most complicated and difficult to explain. The name is derived from a similar, but not identical, function in the data table Stack dialog.

Occasionally, data is organized in a way where there is a hierarchy of information in the column header. In the example below, some machines on a production line contain information on the number of parts produced in a given timespan, the number of lost parts due to defects, and the amount of waste material generated.

	A	B	C	D	E	F	G	H	I	J	K
1			Machine 1			Machine 2			Machine 3		
2	Date	Run	Parts	Lost	Waste (gm)	Parts	Lost	Waste (gm)	Parts	Lost	Waste (gm)
3	1/1/2016	1	25	5	7.333361009	29	1	8.917793573	29	1	3.690662653
4	1/2/2016	2	27	3	0.243529299	28	2	6.692772428	30	0	1.686455719
5	1/3/2016	3	27	3	2.99034232	29	1	9.805004818	30	0	8.47854749
6	1/4/2016	4	26	4	5.76665524	29	1	7.609412124	30	0	0.466871018
7	1/5/2016	5	24	6	0.815343087	30	0	3.674001165	29	1	3.499085684
8	1/6/2016	6	26	4	0.591421	30	0	8.17056521	29	1	0.51250055
9	1/7/2016	7	28	2	6.110704854	29	1	7.997784959	30	0	5.461944888
10	1/8/2016	8	30	0	5.179569316	29	1	2.720275079	30	0	5.986753625
11	1/9/2016	9	30	0	3.386617807	28	2	1.632096778	29	1	0.389729003
12	1/10/2016	10	30	0	0.554209796	27	3	1.452432318	29	1	3.681520454
13	1/11/2016	11	30	0	6.537765422	25	5	1.69462562	28	2	2.646984768
14	1/12/2016	12	29	1	7.091228878	25	5	1.741778008	29	1	9.386144314
15	1/13/2016	13	29	1	3.909534895	24	6	6.849493164	29	1	4.853678708
16	1/14/2016	14	30	0	6.533786355	23	7	6.88535752	30	0	4.781785437
17	1/15/2016	15	29	1	1.341532785	23	7	0.646282748	29	1	6.369285543
18	1/16/2016	16	29	1	2.392781981	23	7	1.799571285	30	0	2.925131061
19	1/17/2016	17	30	0	4.058619803	22	8	8.698206849	30	0	6.207197229
20	1/18/2016	18	30	0	7.452887119	23	7	1.840809278	29	1	3.656127239

Original data in Excel

We can use the original Wizard feature **Treat multiple column header lines as hierarchies** to take the data in the column headers themselves and reorganize the rows to include that information:

Data Preview

	Date	Run	Column	Column 2	Data
1	01/01/2016	1	Machine 1	Parts	25
2	01/01/2016	1	Machine 1	Lost	5
3	01/01/2016	1	Machine 1	Waste (gm)	7.333361009
4	01/01/2016	1	Machine 2	Parts	29
5	01/01/2016	1	Machine 2	Lost	1
6	01/01/2016	1	Machine 2	Waste (gm)	8.917793573
7	01/01/2016	1	Machine 3	Parts	29
8	01/01/2016	1	Machine 3	Lost	1
9	01/01/2016	1	Machine 3	Waste (gm)	3.690662653
10	01/02/2016	2	Machine 1	Parts	27
11	01/02/2016	2	Machine 1	Lost	3
12	01/02/2016	2	Machine 1	Waste (gm)	0.243529299
13					

After clicking Treat multiple column header lines as hierarchies

This makes the data more useful to JMP by making the column types the same and including the Machine data with each measurement. The problem is that the lower hierarchy with Parts, Lost (amount lost) and Waste was grouped by machine. It would be easier to work with, in this case, if we could sort the top hierarchy and then introduce new columns for the data organized by the second hierarchy. Selecting the **Multiple series stack** option will produce a table that looks much like the original Excel table, but with the primary hierarchy (Machine) stacked on top of each other. Since JMP

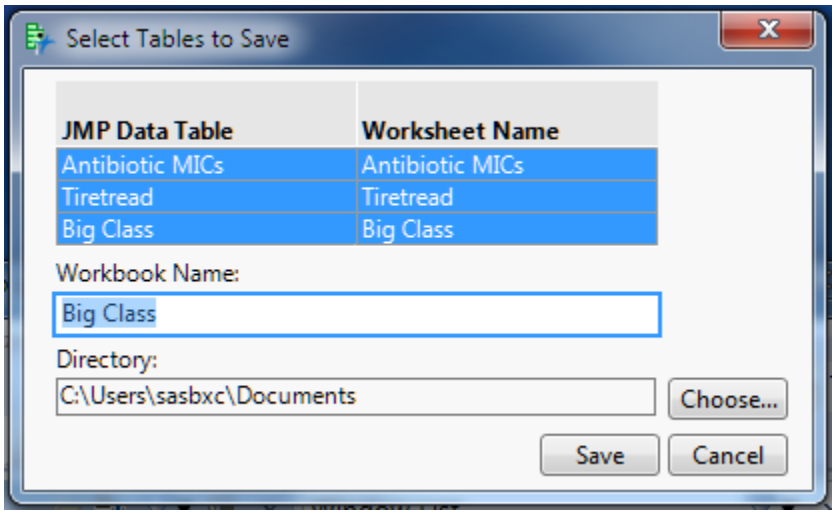
doesn't know the context of the initial header, it assigns the name "Label" to that column data. The data now looks like:

Data Preview

	Date	Run	Label	Parts	Lost	Waste (gm)
1	01/01/2016	1	Machine 1	25	5	7.333361009
2	01/02/2016	2	Machine 1	27	3	0.243529299
3	01/03/2016	3	Machine 1	27	3	2.99034232
4	01/04/2016	4	Machine 1	26	4	5.76665524
5	01/05/2016	5	Machine 1	24	6	0.815343087
6	01/06/2016	6	Machine 1	26	4	0.591421
7	01/07/2016	7	Machine 1	28	2	6.110704854
8	01/08/2016	8	Machine 1	30	0	5.179569316
9	01/09/2016	9	Machine 1	30	0	3.386617807
10	01/10/2016	10	Machine 1	30	0	0.554209796
11	01/11/2016	11	Machine 1	30	0	6.537765422
12	01/12/2016	12	Machine 1	29	1	7.091228878
13	01/13/2016	13	Machine 1	29	1	3.909534895
14						

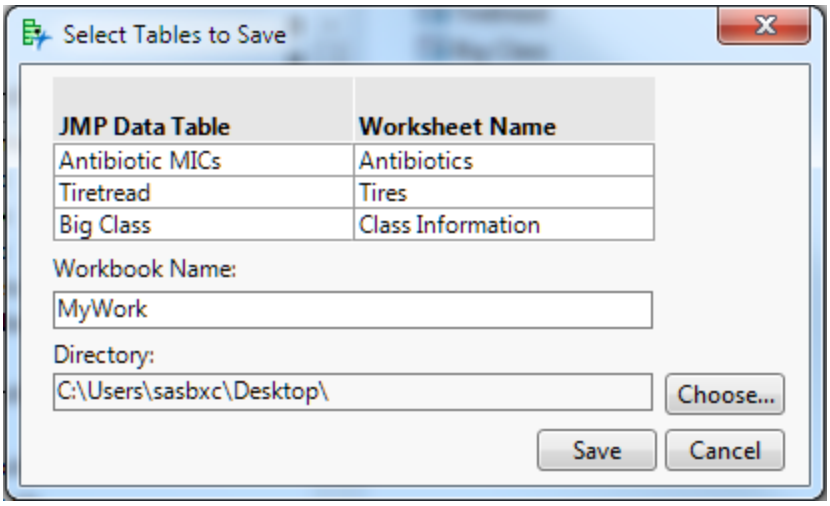
Please note that it is essential that the option for **Treat Multiple Column Names as Hierarchies** be selected before selecting **Multiple Series Stack.** If you don't do this, the feature will not do anything.

That covers the additions to the Excel Wizard. JMP 13 will also add a new facility to aggregate multiple JMP tables into a single Excel workbook, with each JMP table becoming an Excel worksheet. This has been a common request. Under the View menu, you will now find an item called **Create Excel Workbook**. This menu item will produce a dialog that shows all of the open, visible JMP data tables. You can then select the ones that you wish to aggregate into an Excel workbook.



Initial Create Excel Workbook dialog

Within this dialog, you can edit the names of the worksheets, and give the workbook a new name and location for the save. Here is an example where everything has been customized:



Worksheet and workbook names modified

The resulting Excel workbook will have sheets for each of your JMP tables. Here is a snippet:

18	ELIZABETH	14	F	62	91
19	LESLIE	14	F	65	142
20	CAROL	14	F	63	84
21	PATTY	14	F	62	85
22	FREDERICK	14	M	63	93
23	ALFRED	14	M	64	99
24	HENRY	14	M	65	119
25	LEWIS	14	M	64	92
26	EDWARD	14	M	68	112
27	CHRIS	14	M	64	99
28	JEFFREY	14	M	69	113
29	MARY	15	F	62	92
30	AMY	15	F	64	112
31	ROBERT	15	M	67	128
32	WILLIAM	15	M	65	111
33	CLAY	15	M	66	105
34	MARK	15	M	62	104
35	DANNY	15	M	66	106
36	MARTHA	16	F	65	112
37	MARION	16	F	60	115

Antibiotics | Tires | **Class Information**

Resulting Excel workbook

There are two important points to consider when using this feature. First, the saved file will be the newer Excel (.xlsx) format. There is no option for .xls format. Second, the JMP tables must conform to the maximum Excel table limits of 1 million rows and 16,535 columns or they will be truncated.

There is also a JSL interface for this feature. The example below generates the same table that we created above. The tables to save must already be open. In this example, I have created the lists of tables and sheet names prior to the **Create** call, but this is not required.

```
tableList = {"Antibiotic MICs", "Tiretread", "Big Class"};
sheetList = {"Antibiotics", "Tires", "Class Information"};

Create Excel Workbook("$DESKTOP\MyWork.xlsx", tableList, sheetList);
```

The table list contains the names of the JMP data tables. The sheet list contains the names that you would like to give to each sheet within the new workbook. There is a one-to-one mapping of table and sheet names assumed. If you don't want to modify the sheet names, you can just leave off the third parameter:

```
Create Excel Workbook("$DESKTOP\MyWork.xlsx", tableList);
```

Or if you want JMP to use all the open tables with default names, you can just supply the first parameter.

```
Create Excel Workbook("$DESKTOP\MyWork.xlsx");
```

Hopefully in JMP 13 we have struck the right balance between adding new features and keeping the interface intuitive for the majority of users. As always, we welcome your input.