

Use of ODS tagsets.excelxp to create Excel type files

Douglas Staddon, Cmed Ltd, Horsham, UK

ABSTRACT

Excel files are widely understood in many departments within an organization so if you can create them easily this should aid communication and data quality. This paper will show how to use ODS tagsets.excelxp to create XML files that can be opened in Excel just like a normal spreadsheet. Case study data from the CDISC pilot study has been used to demonstrate the benefits and problems encountered. An XML file was created for each patient in the study with separate tabs per raw dataset and also a file of the whole database.

KEYWORDS

Excel, Tagsets, excelxp, XML, patient profile

INTRODUCTION

Excel files can be created in these main ways in SAS:

- Dynamic data exchange (DDE) with Excel open.
- Proc Export
- Libname XLS excel "&path\filename.xls";
- Ods tagsets.excelxp ... ;

All except tagsets require SAS Access for PC file formats.

SIMPLE EXAMPLE USING TAGSETS

The SAS file SASHELP.CLASS was converted into an XML file that can be opened directly in Excel.

This can be done with this code:

Note : Macro variable &path will depend on your operating system.

```
ods tagsets.excelxp
  file="&path./sashelp.xml"
  style=sasweb ;
```

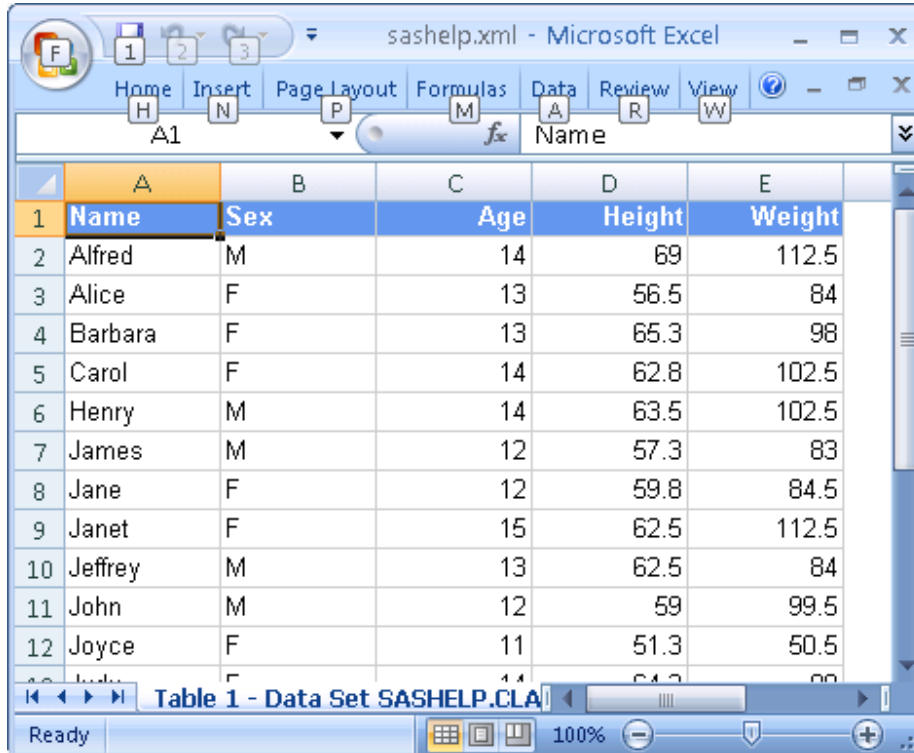
```
proc print
  data=sashelp.class
  noobs ;
run;
```

```
ods tagsets.excelxp close;
```

PhUSE 2011

Paper CC06

This creates the file sashelp.xml shown here:



	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Alfred	M	14	69	112.5
3	Alice	F	13	56.5	84
4	Barbara	F	13	65.3	98
5	Carol	F	14	62.8	102.5
6	Henry	M	14	63.5	102.5
7	James	M	12	57.3	83
8	Jane	F	12	59.8	84.5
9	Janet	F	15	62.5	112.5
10	Jeffrey	M	13	62.5	84
11	John	M	12	59	99.5
12	Joyce	F	11	51.3	50.5

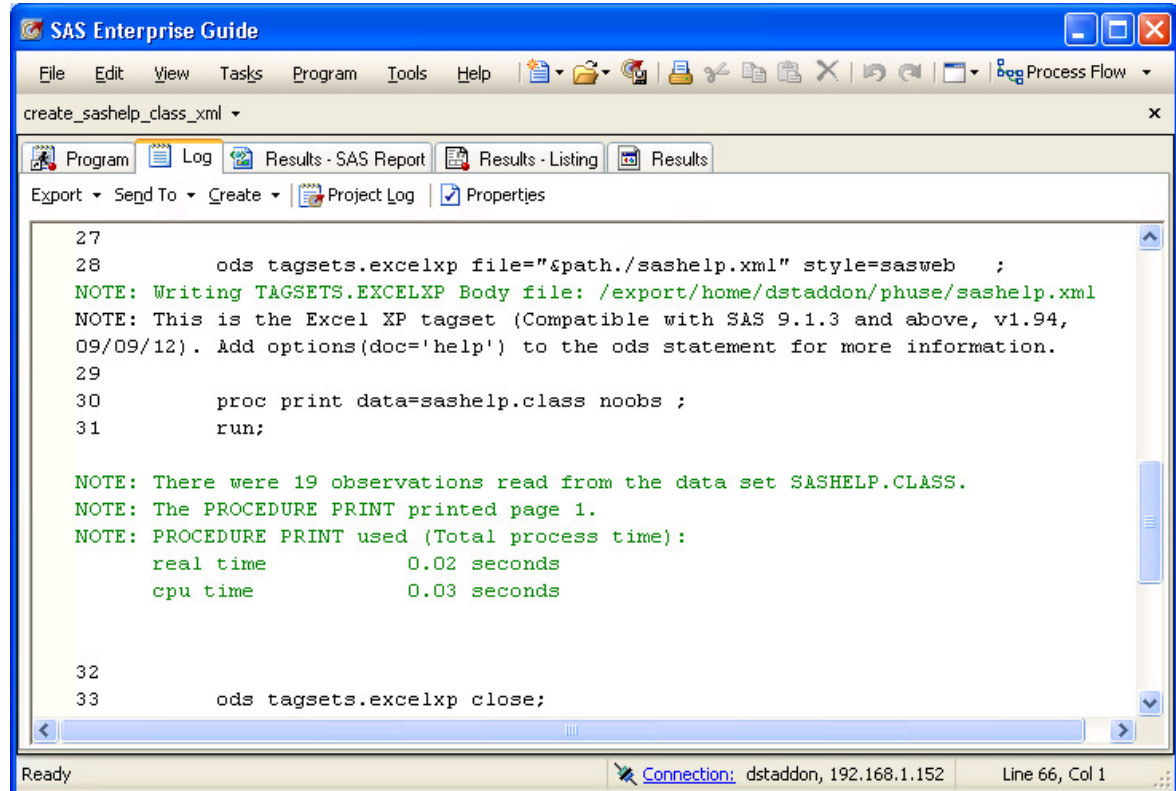
A few things to notice in the file are as follows:

- Options for PROC PRINT can be used to affect the output – so NOOBS suppresses the observation numbers. Columns could be dropped if not required using VAR statements and so on.
- SAS “styles” can be used to add colour, define size of text etc.
- Column titles are the variable names.
- Text is left justified as per Excel and numbers right justified.
- The column widths are passed to Excel by PROC PRINT so the data does not wrap.
- The sheet name has a maximum of 31 characters so it is truncated “Table 1 – Data Set SASHELP.CLAS”.

PhUSE 2011

Paper CC06

The SAS log also contains version and help information.



```
27
28     ods tagsets.excelxp file="&path./sashelp.xml" style=sasweb ;
NOTE: Writing TAGSETS.EXCELXP Body file: /export/home/dstaddon/phuse/sashelp.xml
NOTE: This is the Excel XP tagset (Compatible with SAS 9.1.3 and above, v1.94,
09/09/12). Add options(doc='help') to the ods statement for more information.
29
30     proc print data=sashelp.class noobs ;
31     run;

NOTE: There were 19 observations read from the data set SASHELP.CLASS.
NOTE: The PROCEDURE PRINT printed page 1.
NOTE: PROCEDURE PRINT used (Total process time):
      real time           0.02 seconds
      cpu time            0.03 seconds

32
33     ods tagsets.excelxp close;
```

OPTION TO PRINT ODS TAGSETS HELP

Use the SAS code below to print help information to the log:

```
ods tagsets.excelxp options (doc='help');
```

There are a lot of options and I will mention a few important ones below. Suffice it to say at the moment that these options can affect the way the Excel spreadsheet looks, the way it prints and even allow SAS data to become Excel formulas.

OPTION FOR NAMING SHEETS

To name a sheet just use the option SHEET_NAME="..." before each PROC PRINT

```
ods tagsets.excelxp options (SHEET_NAME="SASHELP.CLASS age le 12");
proc print data=sashelp.class noobs ;
  where age le 12;
  var name sex age;
run;
```

PhUSE 2011

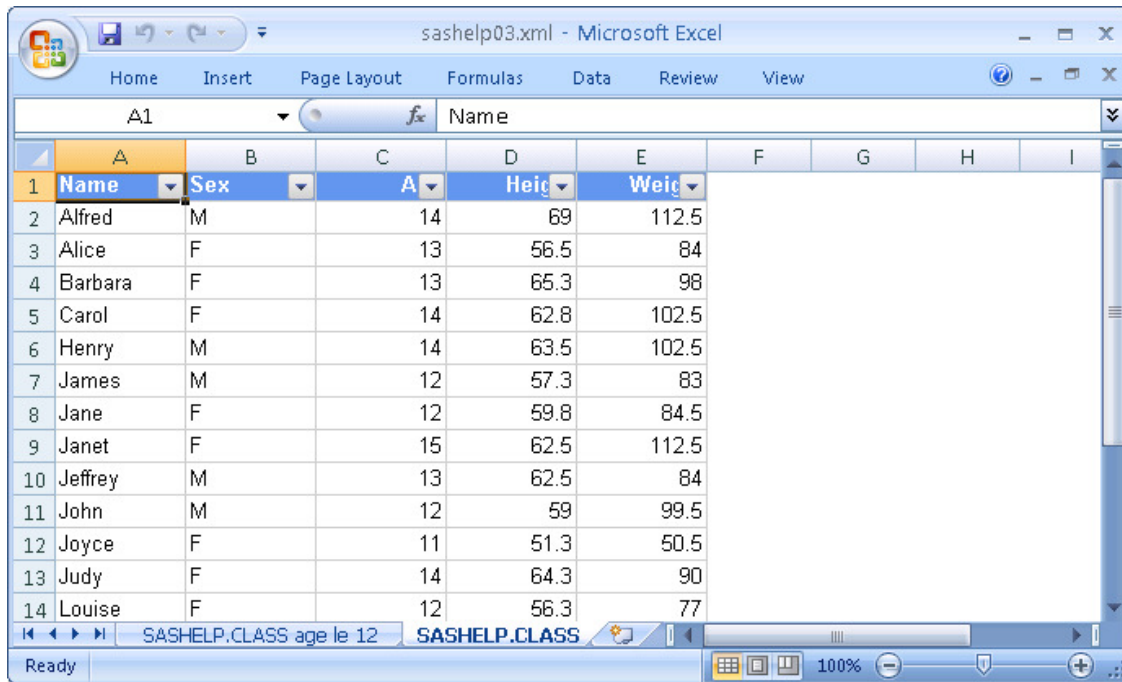
Paper CC06

OPTION FOR AUTOFILTER COLUMNS

Filtering columns can be useful to see the range of values in a column.

```
ods tagsets.excelxp options (SHEET_NAME="SASHELP.CLASS" autofilter='yes');  
proc print data=sashelp.class noobs ;  
run;
```

The filter arrow may hide the column name so these can be formatted in Excel or the column made wider.



The screenshot shows a Microsoft Excel window titled 'sashelp03.xml - Microsoft Excel'. The worksheet contains data from the SASHELP.CLASS dataset. Columns A, B, D, and E have filter arrows. The data is as follows:

	A	B	C	D	E	F	G	H	I
1	Name	Sex	A	Heig	Weig				
2	Alfred	M	14	69	112.5				
3	Alice	F	13	56.5	84				
4	Barbara	F	13	65.3	98				
5	Carol	F	14	62.8	102.5				
6	Henry	M	14	63.5	102.5				
7	James	M	12	57.3	83				
8	Jane	F	12	59.8	84.5				
9	Janet	F	15	62.5	112.5				
10	Jeffrey	M	13	62.5	84				
11	John	M	12	59	99.5				
12	Joyce	F	11	51.3	50.5				
13	Judy	F	14	64.3	90				
14	Louise	F	12	56.3	77				

POSSIBLE USES OF ODS TAGSETS.EXCELXP FILES

The ability to create multiple sheets very easily in the same workbook opens up a lot of useful applications. Some examples are:

1. Derived data specifications can be created by using the metadata of datasets already created, in a similar way to a CDISC DEFINE.XML. All you need to do is record the derivations in some text and merge with the metadata from a PROC CONTENTS.
2. Compare MedDRA /WHODRUG versions and create multiple sheets of new, deleted and amended records.
3. Put all the data for the study in one Excel file. This can be used to look at protocol deviations or review data for consistency.
4. Summarize data per patient having a sheet for each domain in the database. This has been done for a case study on the CDISC Pilot 01 study - see below. I call this a patient profile and it is very useful for reviewing data for a patient and fixing data errors.

PhUSE 2011

Paper CC06

CASE STUDY CDISC PILOT 01

To try out these ideas the data from the CDISC Pilot 01 study was used to see what, if any, issues cropped up. This data is not required since these techniques could be applied to any study where the SAS data is located in one LIBNAME.

CREATE XML EXCEL PATIENT PROFILE FILES

The CDISC pilot study consists of data from a real study that was made available to learn about CDISC and SDTMs etc. Knowledge of CDISC is not required. It has been used here because it is representative of a real study and data is available.

The study contains multiple domains from the usual safety but also some efficacy. Some data has already been anonymised for patient confidentiality. There were a total of 306 patients in demography (DM) .

The SAS program needs a few macro variables to be set to be able to run:

- Name of library or libref where the data is located – e.g. RAW
- Name of dataset that contains one record per patient – e.g. DM
- Name of variable that is the patient identifier – e.g. USUBJID
- Location for output files – depending on your operating system.

The SAS code has been provided in appendix 1 for your information.

The program:

- Uses output from a PROC CONTENTS to find datasets that contain the patient identifier.
- Creates a macro variable containing the patient numbers from the specified dataset.
- For each patient loops through each dataset and prints the data into an Excel XML file with presentational quality output.

RESULTS

SAS 9.2 running on a Solaris server was used with Enterprise Guide 4.2.

- 306 files were created ranging in size from 64KB to 3062 KB – the total of all files was 493 MB.
- The small files were patients who were screening failures. Only tabs that contain data are included.
- It took about 23 minutes to loop through all the patients.

The results were stored in a central location so Data Management could view the files. For ongoing studies they should not be amended and will need updating as required as more data is made available over time.

THINGS TO NOTE

1. If you amend the ODS PATH for a study you need to keep the original references to where the default template files are located. Otherwise the XML file may give errors when opened.

The default ODS PATH can be determined by running the code

```
ods path show;
```

Giving the following in the log:

Current ODS PATH list is:

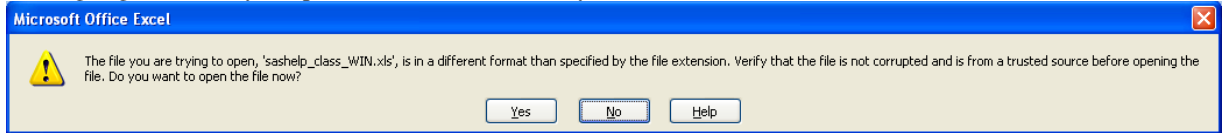
1. WORK.TEMPLAT(UPDATE)
2. SASUSER.TEMPLAT(READ)
3. SASHELP.TMPLMST(READ)

2. The XML files created are quite large. If you open the file in Excel and save as XLS the file will reduce in size quite dramatically. An XML file of the whole CDISC project is 476 MB but the file saved as XLSX is only 27MB. Alternatively you can also ZIP all the XML files together which also makes a large difference.

PhUSE 2011

Paper CC06

3. SAS 9.1.3 XML file does not open automatically in Excel. You can amend the extension to XLS but a warning is given when you open the file since it is really an XML file and click YES.



4. Avoid text that starts with an equals = sign. Excel will try to use a formula - unless this option is turned off using the following code:

```
ods tagsets.excelxp options (formulas='No');
```

CONCLUSION

ODS TAGSETS for Excel are easy to use and have a whole range of options to affect the look of the resulting spreadsheet giving great flexibility. No extra SAS modules are required like SAS ACCESS for PC FILE FORMATS and the method of creating XML should be generic for most if not all platforms.

Patient profiles are easy to create but the files may be very large. This can be solved to some extent by opening and saving to actual XLS files.

Creating XML tagset files will show how useful they can be and highlight many applications that will improve communication between groups working on clinical studies.

Other procedures can be used instead of PROC PRINT like MEANS, REPORT and TABULATE – so just experiment!

PhUSE 2011
Paper CC06

APPENDIX

```

/*****
*   Study: CDISC Test study
*   Program name: profile_xml.sas
*   Location: ../programs
*   Description: Selects raw data per patient into XML files
*   Input files: raw.*.sas7bdat
*   Macros:      dslist, printobs, dsloop, allparts, patloop
*   Output files: *.xml
*   Assumptions: Patient identifier is character for where clause
*
*   Program History:
*   =====
*   Date       |      Name      | Reason/Comments
*   07Aug2011  | D Staddon     | Original
*****/

***Source datasets***;
libname raw "<raw library path>" access=readonly;

%let patdata=RAW.DM;    ** dataset containing one record per patient (uppercase) **;
%let patvar =USUBJID;   ** patient identifier (uppercase) **;

%let Project = %Str(CDISC Test Study);
%let patpath = <location for files>;

%macro dslist;
  /* find which datasets contain the patient variable of interest */
  proc datasets lib=work kill nolist;run;quit;

  proc contents data=raw._all_ memtype=data out=_rawcont varnum noprint;run;

  data rawcont;
    set _rawcont;
    where upcase(name)="&patvar";
  run;

  %let dslist=;
  %let dsnum=;
  proc sql noprint;
    select distinct memname      into: dslist separated by ' ' from rawcont ;
    select count (distinct memname) into: dsnum                from rawcont ;
  quit;
  %put dslist= &dslist  dsnum=&dsnum.;

%mend dslist;

```

PhUSE 2011
Paper CC06

```
%macro printobs(indata);
  /* print data for patient - if any */
  ods tagsets.excelxp options (SHEET_NAME="&indata.");

  proc print data=raw.&indata. noobs ;
    where &patvar.="&pat."; ** assumes patient id is character **;
  run;

%mend;

%macro dsloop;

  ** loop through all datasets per patient **;
  %do i=1 %to &dsnum. ;
    %let dataset=%scan(&dslist.,&i.,' ');
    %put dataset=&dataset;

    %printobs(&dataset);

  %end;
%mend;

%macro allparts(pat);

  ** create XML file with each dataset as one SHEET **;
  ods tagsets.excelxp file="&patpath./&project._profile_&pat..xml" style=sasweb
    options(FROZEN_HEADERS='1'
      AUTOFILTER='ALL'
      rowcolheadings='Yes'
    ) ;

  %dsloop;

  ods tagsets.excelxp close;

%mend;

** test for one patient **;
%*allparts(01-701-1015);

** loop through patients in patdata variable eg RAW.DM ;
%macro patloop;

  %dslist;

  ** loop through patients **;
  /* %do j=1 %to 3 ; ** limiting number of patients in this test to 3 **;*/
  %do j=1 %to &patnum. ;
    %let pat=%scan(&patlist.,&j.,' ');
    %allparts(&pat.);
  %end;
%mend;
```

PhUSE 2011

Paper CC06

```
%let patlist=;
%let patnum=;
%global pat dslist dsnum;

proc sql noprint;
    select distinct &patvar          into: patlist separated by ' ' from &patdata. ;
    select count (distinct &patvar) into: patnum          from &patdata. ;
quit;
%put patlist= &patlist patnum=&patnum.;

options orientation=landscape ps=40;
options ls=200; ** to avoid warn-ing re linesize **;

%patloop;

options ls=120; ** change back **;

** end of program **;
```

REFERENCES

1. <http://www.cdisc.org/content1037> members area for CDISC pilot study accessed on July 26, 2011

RECOMMENDED READING

1. Amadeus Course: Using SAS to create Microsoft Excel reports.
2. http://support.sas.com/rnd/base/ods/odsmarkup/excelxp_demo.html accessed on July 26, 2011

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Author Name: Douglas Staddon

Company: Cmed Group Ltd

Address: Holmwood, Broadlands Business Campus, Langhurstwood Road

City / Postcode: Horsham, West Sussex, RH12 4QP, United Kingdom

Work Phone: +44 (0) 1403 755 422

Fax: F: +44 (0)1403 755051

Email: dstaddon@cmedltd.com

Web: <http://www1.cmedresearch.com/>

Brand and product names are trademarks of their respective companies.