PhUSE 2006

Paper TS01

Listings and Patient Summaries in Excel (SAS and Excel, an excellent partnership)

Xavier Passera, Detour Solutions Ltd., United Kingdom

ABSTRACT

The purpose of this paper is to explain how SAS and Excel can be used together to produce listings, patient summaries and ad-hoc listings while minimising the programming in SAS and VBA (Excel). While SAS can be used to produce these outputs, it becomes quite time consuming to change SAS code when the requirements change daily.

The process uses SAS to manipulate the data and Excel to organize the layout. As such, SAS preprocesses the data by organizing and formatting it, while Excel is used to determine the layout. This paper covers the SAS preprocessing requirements. However the emphasis is that end-users determine the layout (in Excel) and this described by using Excel functions and VBA macros.

If you have listing or patient summary requirements that change regularly or are unclear, this paper should offer an innovative solution.

INTRODUCTION

Using Excel to produce listings, patient summaries or ad-hoc listings is not the tool that first comes to mind. SAS can do these tasks quite well. However when the requirements change regularly or are unclear, the time required to program the changes can be quite time-consuming. It's even more challenging when the users are in different time zones and physical locations, have different paper formats and have personal requirements.

Using SAS and Excel together is a solution that solves these issues. One SAS program per study is setup to process all the datasets and ensure they reflect the CRF. The transformed datasets are copied to an Excel template containing macros (VBA). These macros provide the user(s) a custom toolbar allowing them to organize the data, hide variables etc.

By using Excel (ieasy to use/ minimal training), end-users are given the flexibility to decide how they want their listings and PS to look. From a programmers point of view, a standard way of exporting/importing data to/from excel provides documentation and a link between the two systems.

This paper looks at the SAS programming requirements and issues when preparing data for Excel. In Excel this paper will show two methods on how to use the data, such as the use of custom toolbars to import/export ASCII files and how to create a listing and PS.

PREPROCESSING DATA IN SAS

Documenting data sent to Excel is particularly useful when you have to support the process and questions are asked regarding the source of the data. If you send data with no header part then you rely on the documentation being stored somewhere else.

Whether you're sending 1 or many datasets to Excel, certain aspects are identical. The golden rule is that documentation should be automatic and that data should reflect the layout of the CRF. This can be described as follows:

- All datasets should have a label, example: DEMOG1 = Demography Details, DEMOG2 = Randomization
- All variables should have a label and be decoded (or use the formatted value)
- Variables in all datasets should follow the order of the CRF.
- Variables with no information (all missing) should be removed from the datasets
- Datasets that have information from different CRF pages should be split up. For example, if the DEMOG dataset contains demography details, vital signs, pregnancy information and randomisation information then the dataset should be split up in 4 datasets. Basically splitting up datasets makes printing easier in Excel.
- File size may be an issue in Excel but as this is PC dependent and data/study dependent, trial-and-error is used to determine the best format. For example this may mean subsetting the data by country or centre.

(Note: The header part not only documents the data sent, it also allows to load data from Excel in SAS quite easily. A generic macro can be written to do this.)

EXAMPLE

```
%put ** DEMO **;
proc sort data=demo (keep=center pt prot subjinit page subjstat visit
    birthd age sex race hgt wgt)
    out=demo1 (label="Demographic Details");
  by crtn pt ;
run;

** Order the variables in the dataset **;
%S_ORDER (OrdData =demo1,
          OrdVar  =center pt prot subjinit page subjstat visit
    birthd age sex race hgt wgt );

** Convert variables to character and use format if found **;
%S_ADJUST (AdjLibn =WORK,
           AdjMemn =DEMO1,
           AdjForm =YES);
```

CREATE EXPORT FILE FROM A SINGLE DATASET

Sending a single dataset to Excel is done using the ASCII method and using the file extension naming convention CSV. CSV files are associated with Excel, meaning Excel loads automatically when you double-click on them.

The structure of the ASCII file is:
Row 1 : Dataset label
Row 2 : Variable Labels
Row 3 : Variable Names
Row 4 : Variable Formats
Row >5: Data

EXAMPLE

```
%S_EXCEL (ExcData  =DEMO1,
          ExcOut   =”$HOME/demo1.csv”,
          ExcHead  =Demography information);
```

The macro creates an ASCII file using the delimiter ";". A blank space is added to the data part so that Excel doesn't format the cells.

CREATE EXPORT FILE FROM MULTIPLE DATASETS

The process described above needs to be repeated for all the datasets that need to be transferred. A program can be written to do this. Depending on the users and the requirements, the program may need to be flexible, such as splitting data by region.

Using the ASCII method would be time-consuming for multiple datasets so PROC CPORT is used.

EXAMPLE

```
filename study "study.xpt";
proc cport lib=work file=study;
run;
```

TRANSFER DATA TO PC AND EXCEL

At this point we have an ASCII file (eg. demo.csv) or a transport file (eg. study.xpt). They need to be transferred to the PC with Excel. FTP can be used for this task.

Once the file is saved in Excel, it can be sent to any user. They'll be able to use the range of functionalities provided by built-in macros (VBA) and Excel to format the outputs to their needs. The built-in macros are described below.

EXCEL: IMPORTING A SINGLE DATASET

Double-click on the ASCII file and Excel should load and display the data.
However the header part is not formatted and it looks like a poor listing. Excel should have loaded an add-in, which in this case is a custom menu (called HEADER, see add-in section below) used to format ASCII files.

EXAMPLE

A	B	C	D	E	F	G	H	I	J
List of variables found by study									
Library Member Name	**Variable Name**	**Variable Type**	**Variable Length**	**Variable Format**	**Variable Label**	**Study 1**	**Study 2**	**Study 3**	**Study 4**
MEMNAME	NAME	TYPE	LENGTH	FORMAT	LABEL	STU1	STU2	STU3	STU4
$32.	$32.	8	8	$8.	$256.	$1.	$1.	$1.	$1.
DEMOG	TRT1DC	2	9	$	First trial medication begin date	X	X	X	X
DEMOG	TRT1DT	1	8	DATETIM	First trial medication begin datetime		X		
DEMOG	TRT1GRP	2	80	$	First trial medication group	X	X	X	X
DEMOG	TRT1TC	2	8	$	First trial medication begin time		X		

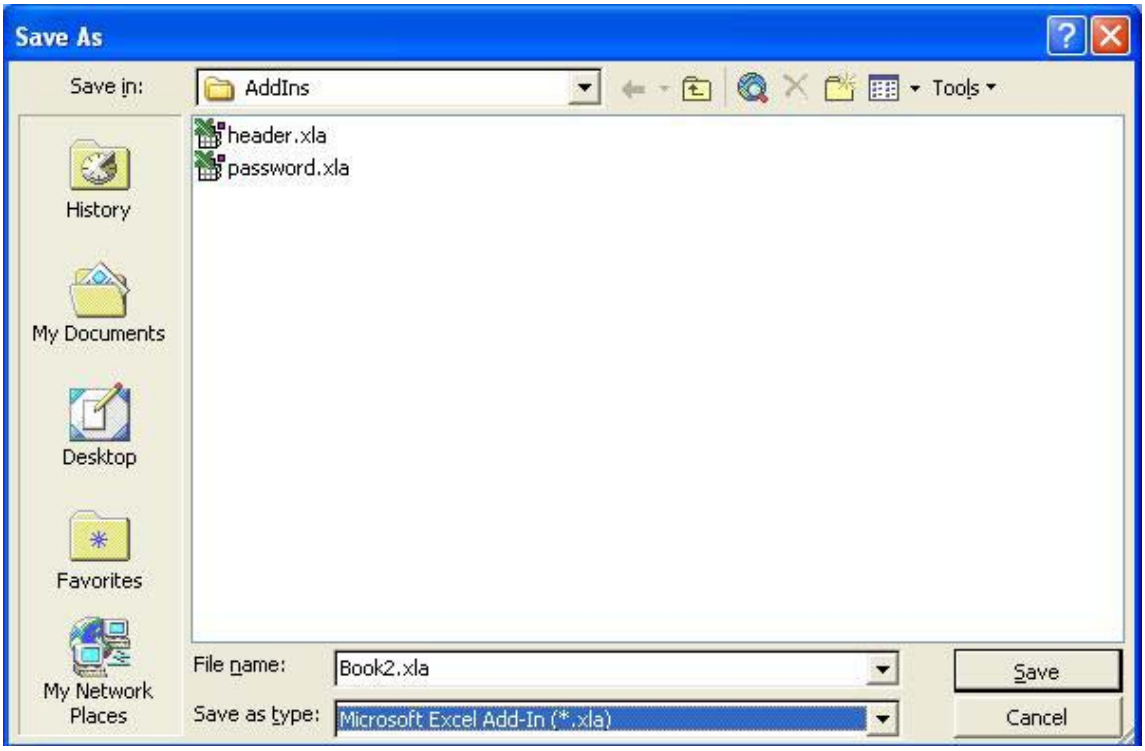
Row 1 : Dataset label or brief description of the type of data
Row 2 : Variable labels
Row 3 : Variable Name
Row 4 : Variable Format
Row 5 and beyond: data
The colouring, font size and style is done by the VBA macro. The column width is also adusted automatically, filters are added and the header part is frozen (ie. the header part is always visible when you scroll down)

HOW TO CREATE AN ADD-IN

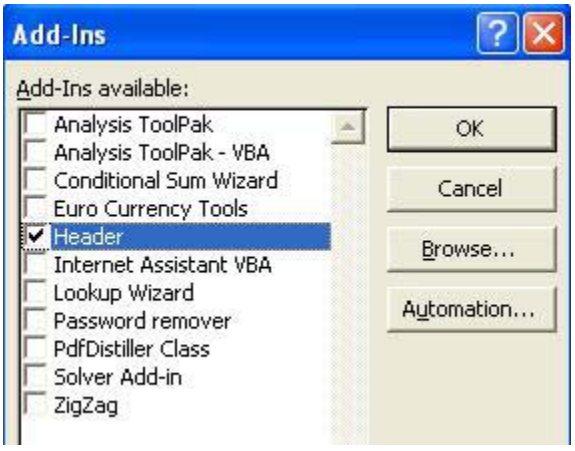
You can use the macro recorder to record tasks and then re-use the macro later on.
As the ASCII files always have the same format, a macro is created to format the header and standardize the way the data is displayed.

Use the macro recorder to record a macro. Then perform the tasks in Excel, such as cell formatting. Once you've recorded the tasks, save the macro. Press ALT-F11 to view the VBA code associated with what you've recorded. You may need to adjust the code depending on the complexity of the task.

Save the macro in the add-in folder.



Next, Excel has to be told to import it everytime you run Excel
Click on menu Tools/Add-ins



Click on browse and select the file with the VBA macro.

In the example above Excel loads an add-in called HEADER everytime it is run

EXCEL: IMPORTING MULTIPLE DATASETS

The data from the transport file called study needs to be transferred to Excel. This is done using SAS/PC with a small AF application which reads the transport file and dumps the data of each dataset in separate sheets. The application also formats the header part of the sheets (freeze panes, adds filters, adjust automatically the column width etc). Some system sheets are also created to help the end-user

This could also be done with regular SAS code. For example the following code runs an excel macro which creates a sheet called VARIABLES. It then formats the header part, sets the column width, adds a filter etc.

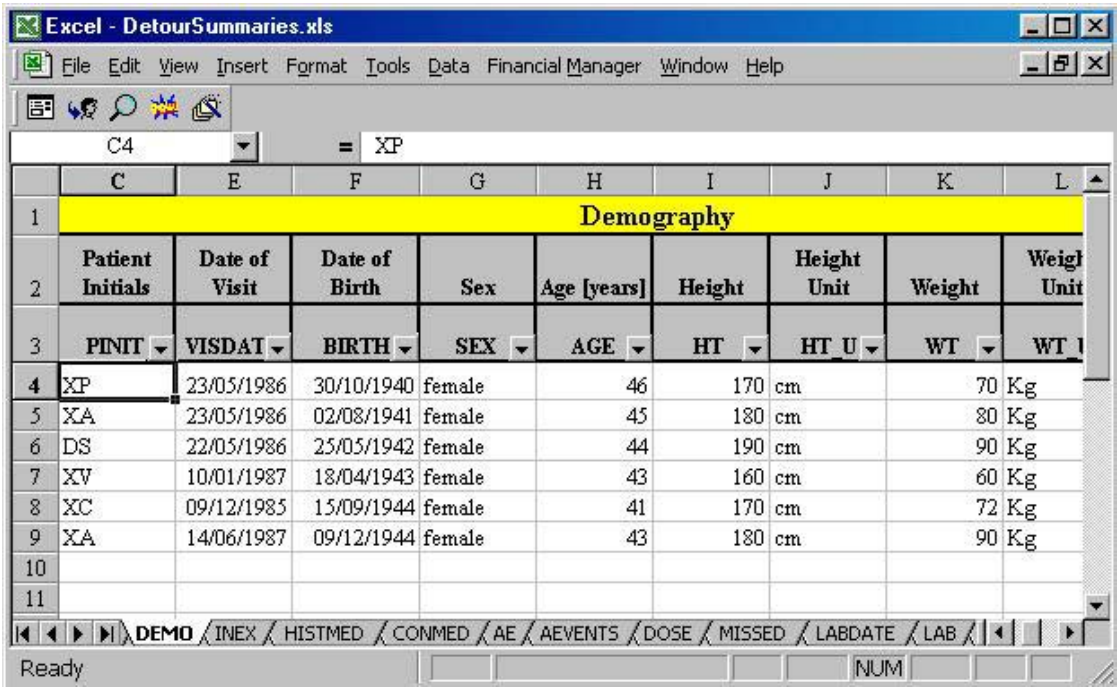
```
Filename _phcdde_ dde "Excel|System" ;
data _null_;
  file _phcdde_ ;
  put '[WORKBOOK.ACTIVATE("Macro1")]';
  ** Run Excel Macro - Insert Worksheet **;
  put '[select("r21c2:r25c2")]';
  put '[run()]';

  ** Setup Sheet **;
  put '[WORKBOOK.ACTIVATE("VARIABLES")]';
  put '[select("r1c1:r1c8")]';
  put '[format.font("Times New
Roman",9,true,false,false,false,true,true,true)]';
  put '[alignment(3,false,3,0)]';
  put '[patterns(10,15,15,true)]';
  put '[border(2,2,2)]';
  put '[row.height(24,"r1")]';
  put '[select("r1c1")]';
  put '[freeze.panes(true,0,1)]';
  put '[column.width(12,"c1")]';
  put '[column.width(26,"c2")]';
  put '[column.width(12,"c3")]';
  put '[select("r1c1:r1c8")]';
  put '[filter()]';
  put '[select("r1c1")]';
run;
```

Once all the datasets have been transferred to Excel, the file is sent to end-users.

EXCEL TO PRODUCE LISTINGS

Excel functionalities such as filters, hiding columns can be used by each user to produce a listing. They also have the full range of Windows functionalities such as copying the output to word or outlook.



Excel - DetourSummaries.xls

File Edit View Insert Format Tools Data Financial Manager Window Help

C4 = XP

	C	E	F	G	H	I	J	K	L
1	Demography								
2	Patient Initials	Date of Visit	Date of Birth	Sex	Age [years]	Height	Height Unit	Weight	Weigl Unit
3	PINIT	VISDAT	BIRTH	SEX	AGE	HT	HT_U	WT	WT_
4	XP	23/05/1986	30/10/1940	female	46	170	cm	70	Kg
5	XA	23/05/1986	02/08/1941	female	45	180	cm	80	Kg
6	DS	22/05/1986	25/05/1942	female	44	190	cm	90	Kg
7	XV	10/01/1987	18/04/1943	female	43	160	cm	60	Kg
8	XC	09/12/1985	15/09/1944	female	41	170	cm	72	Kg
9	XA	14/06/1987	09/12/1944	female	43	180	cm	90	Kg
10									
11									

DEMO / INEX / HISTMED / CONMED / AE / AEVENTS / DOSE / MISSED / LABDATE / LAB

Ready NUM

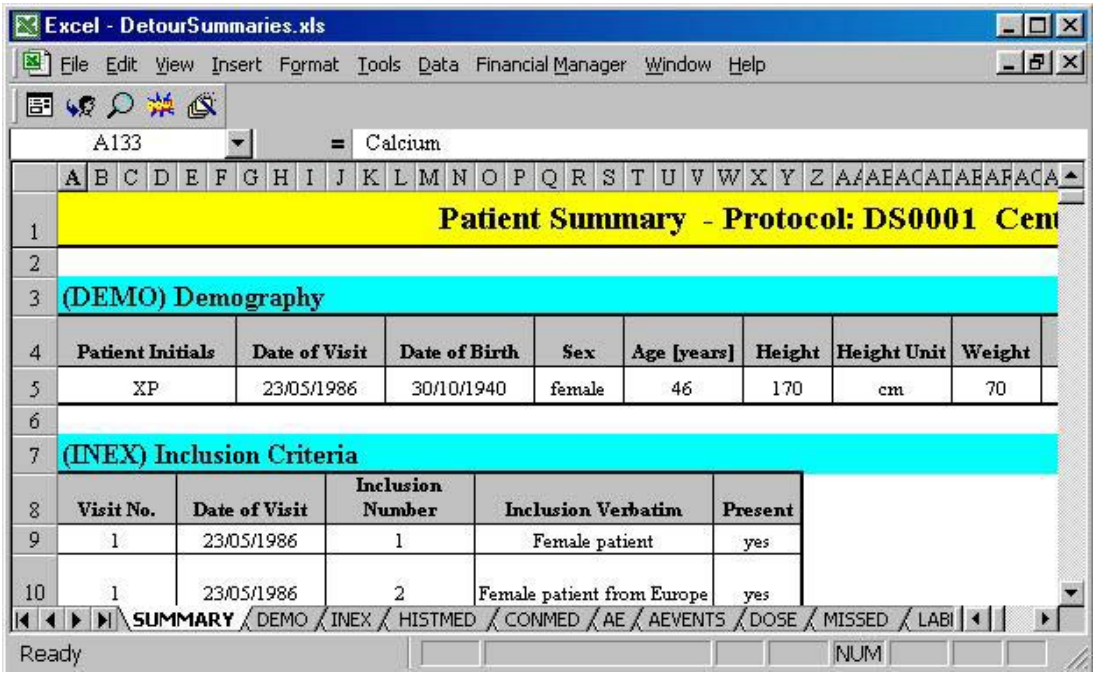
EXCEL TO PRODUCE PATIENT SUMMARIES

Another way to review data is to use Patient Summaries where all the patient data from all the sheets is summarized in one sheet.

Patient summaries can be used :

- for checking data inconsistencies
- for checking data against the CRF (or database)
- for writing narratives to authorities
- as part of a submission dossier.

This method has been used for a submission which required 10000(+) patient summaries.



If PSS are used for a submission then they probably need to be in PDF format. This is done with the batch function which converts the PSS to PDF, incorporating bookmarks for ease-of-use.

THINGS TO CONSIDER

If there are too many variables in a dataset then the PSS won't look tidy. This is dependent on the nature of the datasets and the length of the variables. In general, if a dataset has more than 12 variables you may want to consider splitting it (this is covered in the preprocessing section).

Users can have requests, such as show previous values entered in the database or highlight out-of-range laboratory values. A system dataset called PARMS with the following structure deals with this:

PARMS dataset structure:

- XMEM Dataset name (Excel sheet name)
- XOBS Record number in dataset
- XVAR Name of variable
- XCOMM Comment to insert
- XVAL Colour number

This is done in the preprocessing part.

The volume of data affects how you use the PSS. Excel has hardware (memory) and software limitations.
For example, Excel has a maximum number of rows of 65k.
You need to be aware of the hardware/software environment and structure the preprocessing accordingly.
For example, if the study has 5000 patients, you may want to create a workbook by centre.

CONCLUSION

This paper has described how SAS and Excel can be used together to produce listings and patient summaries. The programming is kept to a minimum by preprocessing the data in SAS and using Excel for the layout of the data. Using Excel for this purpose shifts the display requirements to the user thereby reducing programming time and increasing user satisfaction.

This paper also described (briefly) how to create a macro in Excel using the recorder and create an add-in. This is the first step in learning VBA and getting Excel to do what you want. As Excel is increasingly used by different groups to (re)view data or QC data, you may need to create more specific macros.

An excellent partnership indeed!

RECOMMENDED READING

The best source of information regarding help on SAS or VBA has been the internet.
Search the internet for: ***VBA code examples***.

CONTACT INFORMATION

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

Xavier Passera
Detour Solutions Ltd.
Welwyn Garden City, UK
Mobile: 07930 472 690
Email: Xavier.Passera@detoursolutions.co.uk
Web: www.detoursolutions.co.uk