Introduction

Working in early phase drug development where many varied and disparate information stakeholders want immediate access to the study data, it can seem that I spend more of my time dealing with Excel rather than SAS. There have been many articles over the years dealing with transferring data from SAS® to Microsoft Excel® (DDE, ODS, ADO, SAS v9, SAS Microsoft Office Add-in, XML, text files, SASHOME, SAS/SHARE), but I wish to focus on one aspect of these different methods which perhaps hasn’t received enough attention. When left to its own devises, Excel compromises data integrity by applying its own formats.

The Problem

According to the Microsoft Knowledge Base Article ID 214233, Excel applies its own formats to the data, based on the following criteria:

1. If a number contains a slash mark (/) or hyphen (-), it may be converted to a date format.
2. If a number contains a colon (:) or is followed by a space and the letter A or P, it may be converted to a time format.
3. If a number contains the letter E (in uppercase or lowercase letters; for example, 10e5), or the number contains more characters than can be displayed based on the column width and font, the number may be converted to scientific notation, or exponential format.
4. If a number contains leading zeros, the leading zeros are dropped.

The end result is corrupt data. So how can the SAS programmer safeguard the integrity of their data? Using Excel’s “Text” format ensures the data is treated as just text and preserves the original values. Below are some examples of using the “Text” format using different import methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>via DDE</td>
<td>Pres-select the region to be populated with SAS data. Set the format to “Text” using the format-number function and “@” as the text format identifier. Row 1, column 1 to row 10, column 2 have been formatted to “Text” in readiness for the population by SAS data in the next DATA step.</td>
</tr>
<tr>
<td>via ODS</td>
<td>Use the STYLE option to set the mso-number-format to “@” as the text format identifier. Variable specific, enabling all output for var1 to be correctly formatted. Other variables can have other formats defined.</td>
</tr>
<tr>
<td>via VBA using ADO</td>
<td>Use the FIELDINFO argument is an array of two-element arrays, with each two-element array specifying the conversion options for a particular column. The first element is the column number (1-based). The second element is a constant specifying how the column is parsed. The number 2 means parse as text. If less array statements were defined than there are SAS variables, Excel still imports all the variables but the additional variables are open to data corruption issues.</td>
</tr>
<tr>
<td>via VBA and text files</td>
<td>Pre-select the region to be populated with SAS data. Cell A1 will contain the start of the data, but the end row and column depends on the number of records and variables in the SAS table. Use the obRecordset.RecordCount and obRecordset.Fields.Count properties to select the cells for formatting. Import the data into formatted cells.</td>
</tr>
</tbody>
</table>

Table 1. Genuine Examples of Corrupted Data

<table>
<thead>
<tr>
<th>Actual Intension</th>
<th>SAS Value</th>
<th>Excel Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>% positively stained cells (categorical score)</td>
<td>1-5</td>
<td>01 – May</td>
</tr>
<tr>
<td>questionnaire responses / number issued</td>
<td>60 / 90</td>
<td>2 / 3</td>
</tr>
<tr>
<td>protocol schedule time</td>
<td>01:30</td>
<td>01:30:00</td>
</tr>
</tbody>
</table>

* or more truly 38473 the Excel data serial value

Conclusion

While the knowledge of getting data from SAS to Excel is widely disseminated, its limitations are not so widely known. Hopefully this poster has not only highlighted this issue, but also has put forward some simple solutions. The programmer must be proactive in ensuring data integrity and not leave this vital step to Excel.

Abbreviations

ADO – ActiveX Data Objects  
DDE – Dynamic Data Exchange  
ICM – Integrated Object Model  
ODS – Output Delivery System  
VBA – Visual Basic for Applications  
XML – Extensible Markup Language

Recommended Reading


