m_utl_ds2xls.sas File Reference

Utilities

Utility macro to export a SAS dataset into a XLS or XLSX file

---

Description

This program converts a SAS dataset into a Ms. Excel XLS or XLSX file. This program supports the XLSX engine for exporting to an Excel 2007, 2010, or later spreadsheet (.xlsx format), the XLS engine for exporting to an Excel 97, 2000, 2002, or 2003 spreadsheet (using file formats). Alternatively the EXCELCS engine can be used for exporting to an Excel workbook connecting remotely through SAS PC Files Server (uses the PCFILES engine).

Note

Transcoding is not supported for ENGINE=XLS. If your file has fewer than 255 columns, use ENGINE=EXCELCS as an alternative. Otherwise, save your SAS dataset or database table to an .xlsx file to support (32/64 bit) transcoding.

Autors

Paul Alexander Canals y Trocha (paul.canals@gmail.com)

Date

2022-11-14 00:00:00

Version

22.1.11

Link

https://github.com/paul-canals/toolbox

Parameters

Input	help	Parameter, if set (or ?) to print the Help information in the log. In all other cases this parameter should be left out from the macro call.
Input	base_ds	Full LIBNAME.TABLENAME name of the SAS dataset. The default value is: _NONE_.
Input	xls_file	Specifies the name of the XLS or XLSX file and path where the formatted output is to be written. If the file that you specify does not exist, then its created for you. The default value is: _NONE_.
Input	engine	Indicator [EXCELCS|XLS|XLSX] parameter to specify the engine that is used to export to an Ms. Excel file with XLS(X) extension. The XLSX engine is sufficient to export to XLSX files. For files with an XLS extension, either the XLS or EXCELCS engine can be used. For cases with a mixed 32/64 bit setup environment, either the XLSX or EXCELCS engine should be used for taking care of 32/64 bit conversion. The default value is: XLSX.
Input	newfile	Parameter to specify when exporting a SAS dataset to an existing Excel file (XLS(X) file), whether to delete the Excel file and load the data to a worksheet in a new Excel file. Y specifies that the export procedure deletes the specified Excel file, if it exists. Loads the SAS data set to a worksheet in a new Excel file. N specifies that the EXPORT procedure loads the SAS data set to a worksheet and appends it to the existing Excel file. If the specified Excel file does not exist, an Excel file is created, and the SAS data set is loaded. The default value is: N.
Input	replace	Boolean [Y|N] to decide whether the new XLS(X) output replaces the information that is currently in the specified file or if the new output is appended to the end of the existing file. The default value is: Y.
Input	worksheet	Parameter to specify the name to be exported by the macro. If the parameter value is omitted the worksheet value is equal to the BASE_DS table name.
Input	labels	Boolean [Y|N] parameter to indicate whether to use the variable labels that are defined in the the dataset as your column headings. Case where a variable does not have a SAS label, then the the name of the variable is used instead. The default value is: N.
Input	password	Specifies the that is needed to access a password-protected dataset. This argument is required if the dataset has READ or PW password. There is no need to specify this argument if the dataset has only WRITE or ALTER passwords set.
Input	varlist	Specifies the variables that are to be included in the file and the order in which they should be included. To include all of the variables in the data set, do not specify this argument. If you want to include only a subset of all the variables, then list each variable name and use single blank spaces to separate the variables. Do not use a comma in the list of variable names.
Input	where	Specifies a valid WHERE clause that selects observations from the SAS dataset. Using this argument subsets your data based on the criteria that you supply for where-expression.
Input	debug	Boolean [Y|N] parameter to provide verbose mode information. The default value is: N.

Returns

- Output XLS(X) file.

Calls

- m_utl_create_dir.sas
- m_utl_delete_file.sas
- m_utl_print_message.sas
- m_utl_print_mtrace.sas

Usage

Example 1: Show help information:

```
%m_utl_ds2xls(?)
```

Example 2: Create a new XLS file from SASHELP.class:

```
%m_utl_ds2xls(
   base_ds  = SASHELP.class
 , xls_file = %str(%sysfunc(getoption(WORK))/class.xls)
 , engine   = XLS
 , newfile  = Y
 , debug    = Y
   );
```

Example 3: Create a XLS file from SASHELP.class with variable selection:

```
%m_utl_ds2xls(
   base_ds  = SASHELP.class
 , xls_file = %str(%sysfunc(getoption(WORK))/class.xls)
 , engine   = XLS
 , varlist  = %str(Name Sex Age)
 , debug    = Y
   );
```

Example 4: Create a XLSX file with worksheet 'Boys' from SASHELP.class:

```
%m_utl_ds2xls(
   base_ds   = SASHELP.class
 , xls_file  = %str(%sysfunc(getoption(WORK))/class.xlsx)
 , engine    = XLSX
 , replace   = Y
 , labels    = Y
 , worksheet = Boys
 , where     = %str(Sex = 'M')
 , debug     = Y
   );
```

Example 5: Add worksheet 'Girls' from SASHELP.class to class.xlsx:

```
%m_utl_ds2xls(
   base_ds   = SASHELP.class
 , xls_file  = %str(%sysfunc(getoption(WORK))/class.xlsx)
 , engine    = XLSX
 , replace   = Y
 , labels    = Y
 , worksheet = Girls
 , where     = %str(Sex = 'F')
 , debug     = Y
   );
```

Example 6: Replace worksheet contents from an encrypted table to class.xlsx:

```
data WORK.classfit(encrypt=aes encryptkey=key);
   set SASHELP.classfit;
run;

%m_utl_ds2xls(
   base_ds   = WORK.classfit
 , xls_file  = %str(%sysfunc(getoption(WORK))/class.xlsx)
 , engine    = XLSX
 , replace   = Y
 , labels    = Y
 , worksheet = Boys
 , password  = key
 , where     = %str(Sex = 'M')
 , debug     = Y
   );

%m_utl_ds2xls(
   base_ds   = WORK.classfit
 , xls_file  = %str(%sysfunc(getoption(WORK))/class.xlsx)
 , engine    = XLSX
 , replace   = Y
 , labels    = Y
 , worksheet = Girls
 , password  = key
 , where     = %str(Sex = 'F')
 , debug     = Y
   );
```

Copyright