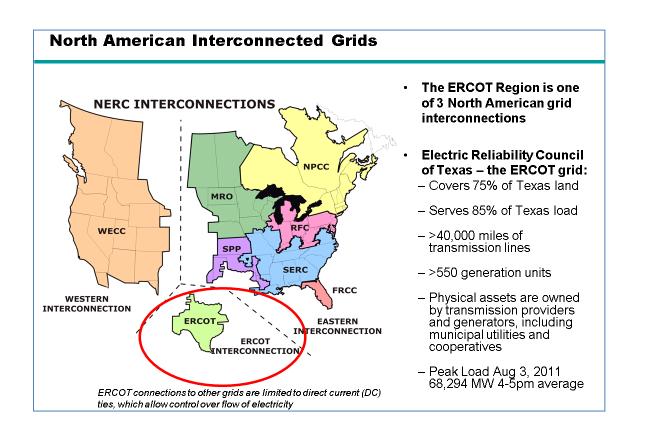
Using SAS to Read From and Write to EXCEL Workbooks Set Up as Templates Which Are Not Set Up In Columns

Carl L Raish, Electric Reliability Council of Texas (ERCOT)

BACKGROUND INDORMATION

This paper will review some sample SAS code developed at to exchange information with some entities in the ERCOT market. By way of providing some context to the code, ERCOT oversees the operations of a major portion of the electric grid in Texas. Our functions include managing reliability of the grid by procuring ancillary services to ensure a continuous balance between generation and load.



Among the services ERCOT procures is the Emergency Response Service; under the terms of this service electricity users agree to reduce their consumption in the late stages of a grid emergency. ERCOT has a \$50 million annual budget to pay those users for committing to quickly reduce their electricity consumption; and we run auctions three times a year to buy the service. The procurement is run in two stages: a resource identification phase and an offer phase.

ERCOT has established an Excel form as the mechanism for users (actually a Qualified Scheduling Entity ... QSE ... representing those users) to identify customers potentially interested in providing the service. ERCOT processes the form with SAS, and returns the form with diagnostic and other information to assist the QSE in submitting an offer. This completes the resource identification phase.

In the offer phase, the QSE fills in additional fields on the form (especially including prices) and returns the form to ERCOT.

The Excel form, pictured below, has three different types of tabs: Identification, Alt and other arbitrarily named tabs. The Identification tab is used for identifying the QSE submitting the form and a set of contact information for the QSE. The tab is set up with a named range, 'id_input_area' to facilitate reading just the required fields with SAS. Cells outside of the range contain information that the QSE should be aware of when submitting the form, but which is not needed for processing the form.

A special version number cell is set up on the tab which is formatted with a white text color to make it less obvious. The version number is changed whenever the form is modified and is used to validate that the QSE is submitting the current form. Since the workbook is being processed with SAS, this is critical to ensure that the data required from the form is pulled from the correct cells. The version number is set up on all tabs in the workbook.

C :)) - ((H + 1)	נ 🖻	D =							ERII	DS by Clie	ent Oct1
	Hom	ne	Insert	:	Page I	ayout	Fo	rmulas	D	ata	Re	view	View	Devel
Paste	* ===	Aria B	I I U		11	• A	A A			≫- (≢ 1		_	ap Text	iter •
Clipboa	ird 💿			For	nt		5			Alio	nm	ent		5
	input	area	3	- 6		fx	OSE	Name						
A		_	E	3			-				С			
9 E 10 E 11 N 12 E 13 d	us iness us iness on-Bus i RCOT wi	Hours Hours ness H Ilsend	2 are th 3 are th Iours and I an ema	ie Hou ie Hou e all ot ail ackr	rs Endi rs Endi herho 10wled	ng 1400 t ng 1700 t urs . gement i	through through ofrece	h 1600, Ma h 2000, Ma ipt ofthis	onday onday docur	rthru Fri rthru Fri nent and	day, day, day,	except B except B guestin	RCOT holid RCOT holid RCOT holid nos regardir y Contact(s	ays. ays. ngthis
14		molate	ed form	to EBS/	Perrot	com								
16	manico	mpiere	Edilorini				luling	Entity (QS	SE) In	formati	on			
	SE Nam	e			-									
I	SE DUN		oer											
	-		ting form	(if diffe	rent tha	n QSE)	+							
·~ -	ubmittal					,	+							
			ed Repr	esenta	ative									
	rinted Na						1							
	ignature:						-							
	itle:						+							{
	hone Nu	mber:					+							{
	ell Phon		oer:				-							i
	-mail Ad	dress:					1							
28	SE Pri	mary C	Contact	: 1										i
29 F	rinted Na	ame:												j
	itle:													
	'hone Nu													
	ell Phon		per:				_							
	-mail Ad													
	ISE Prin rinted Na		Contact	2 (Up	tional	,								
	rinted Na itle:	ame:												{
	itie: 'hone Nu	mbor												{
	none Nu ell Phon		her:								_			{
	-mail Ade						-							{
			Contact	3 (On	tional	1								
	rinted Na			o (op							_			
	itle:													
	'hone Nu	mber:												i
	ell Phon	e Numb	per:]
	-mail Ade	dress:												j
46			~ ~									_		
	H In	tenti	ficatio	m 🖉	≿ ∕ Al	t An	neriso	urce / I	Banct	tec 🦯	Ŷ	/		

The 'Alt' tab provides a way for the QSE, during the resource identification phase, to identify individual site participants that may subsequently be grouped together into an aggregation of sites as a resource. The name 'Alt' must be used, and, as a result, there can only be one Alt tab in the workbook. If the QSE chooses not to use the tab, it may be either deleted from the workbook or simply left blank. No named ranges are set up on the 'Alt' tab; the version number cell is set up as described above for the Identification tab, and a cell is reserved in A2, to be filled in by ERCOT as a control number to be associated with submitted participants in all subsequent processing for the procurement.

6		2 × 🗋	=) =					ERIDS I	by Client C	ct12J	an13 I.xlsx -	Microsof	t Excel						- = x
	Home	Insert	Page I	ayout.	Formulas	Data	Review	View	Developer	A	dd-Ins							0	– 🖷 X
	Cut		Arial		• 10 • A				/rap Text		Text	•				Σ Au	utoSum II *	\mathbb{Z}^1	
Pa	ste 🛷 Format	Painter	BI	<u>U</u> -	🗄 • 🔕 •	<u>A</u> •			lerge & Cen	ter 🔹	\$ - %	• • 00. • 00. • • 00. • 00.	Conditional Format Cell Formatting * as Table * Styles *	Insert De	lete Format	a ci	lear *		& Find & r ▼ Select ▼
	Clipboard	G.		Fo	nt	G		Alignment		- G	Numb	er 「	a Styles	C	ells			Editing	
	H7	-	()	f _x															¥
	A	В	С	D	E	F	G	н		J	К	L	М		N	0	Р	Q	B
1	ERID Number																		
2	ESI-ID	Sub																Private	
~		Metering	Unique Meter ID	LR at this Site	LR Participating in AS Market	Site Owner Controlling Entity	Site Name		Site Address (city)	Zip Code	TDSP Interconnection Agreement	Year of Agreement	TSP/DSP		Duits #	Servic e Area	ale Meter Point	Use	Dn-site Gen
	10443720001234567	NO		this Site NO	Participating	Controlling Entity Acme Brick	Dallas Plant	3815 Maple Blvd	Dallas	Zip Code	Interconnection	Agreement	ONCOR ELECTRIC DELIVERY COMPANY L		1039940674000	Servic e Area NO	ale Meter Point NO	Use Networ k NO	NO
5	10443720001234568	NO NO		this Site NO NO	Participating	Controlling Entity Acme Brick Acme Brick	Dallas Plant Elgin Plant	3815 Maple Blvd 1776 Oak Rd	Dallas Elgin	Zip Code	Interconnection	Agreement	ONCOR ELECTRIC DELIVERY COMPANY L ONCOR ELECTRIC DELIVERY COMPANY L	LC (TDSP)	1039940674000 1039940674000	Servic e Area NO	ale Meter Point NO NO	Use Networ k NO NO	NO NO
5		NO		this Site NO	Participating	Controlling Entity Acme Brick Acme Brick	Dallas Plant	3815 Maple Blvd 1776 Oak Rd	Dallas	Zip Code	Interconnection	Agreement	ONCOR ELECTRIC DELIVERY COMPANY L	LC (TDSP)	1039940674000	Servic e Area NO	ale Meter Point NO	Use Networ k NO	NO
5 6 7	10443720001234568	NO NO		this Site NO NO	Participating	Controlling Entity Acme Brick Acme Brick	Dallas Plant Elgin Plant	3815 Maple Blvd 1776 Oak Rd	Dallas Elgin	Zip Code	Interconnection	Agreement	ONCOR ELECTRIC DELIVERY COMPANY L ONCOR ELECTRIC DELIVERY COMPANY L	LC (TDSP)	1039940674000 1039940674000	Servic e Area NO	ale Meter Point NO NO	Use Networ k NO NO	NO NO
5 6 7 8	10443720001234568	NO NO		this Site NO NO	Participating	Controlling Entity Acme Brick Acme Brick	Dallas Plant Elgin Plant	3815 Maple Blvd 1776 Oak Rd	Dallas Elgin	Zip Code	Interconnection	Agreement	ONCOR ELECTRIC DELIVERY COMPANY L ONCOR ELECTRIC DELIVERY COMPANY L	LC (TDSP)	1039940674000 1039940674000	Servic e Area NO	ale Meter Point NO NO	Use Networ k NO NO	NO NO
5 6 7	10443720001234568	NO NO		this Site NO NO	Participating	Controlling Entity Acme Brick Acme Brick	Dallas Plant Elgin Plant	3815 Maple Blvd 1776 Oak Rd	Dallas Elgin	Zip Code	Interconnection	Agreement	ONCOR ELECTRIC DELIVERY COMPANY L ONCOR ELECTRIC DELIVERY COMPANY L	LC (TDSP)	1039940674000 1039940674000	Servic e Area NO	ale Meter Point NO NO	Use Networ k NO NO	NO NO
5 6 7 8 9	10443720001234568	NO NO		this Site NO NO	Participating	Controlling Entity Acme Brick Acme Brick	Dallas Plant Elgin Plant	3815 Maple Blvd 1776 Oak Rd	Dallas Elgin	Zip Code	Interconnection	Agreement	ONCOR ELECTRIC DELIVERY COMPANY L ONCOR ELECTRIC DELIVERY COMPANY L	LC (TDSP)	1039940674000 1039940674000	Servic e Area NO	ale Meter Point NO NO	Use Networ k NO NO	NO NO
5 6 7 8 9 10 11	10443720001234568 10443720001234569	NO NO NO	MeterID	NO NO NO	Participating in AS Market	Controlling Entity Acme Brick Acme Brick Acme Brick	Dallas Plant Elgin Plant Malakoff Plant	3815 Maple Blvd 1776 Oak Rd	Dallas Elgin	Zip Code	Interconnection	Agreement	ONCOR ELECTRIC DELIVERY COMPANY L ONCOR ELECTRIC DELIVERY COMPANY L	LC (TDSP)	1039940674000 1039940674000	Servic e Area NO	ale Meter Point NO NO	Use Networ k NO NO	NO NO
5 6 7 8 9 10 11 11	10443720001234568 10443720001234569	NO NO NO	MeterID	NO NO NO	Participating	Controlling Entity Acme Brick Acme Brick	Dallas Plant Elgin Plant Malakoff Plant	3815 Maple Blvd 1776 Oak Rd	Dallas Elgin	Zip Code	Interconnection	Agreement	ONCOR ELECTRIC DELIVERY COMPANY L ONCOR ELECTRIC DELIVERY COMPANY L	LC (TDSP)	1039940674000 1039940674000 1039940674000	Servic e Area NO	ale Meter Point NO NO NO	Use Networ k NO NO	NO NO

The other tabs, American and Bankers Inc, are filled in as resources. The 'R' tab is a pre-formatted blank tab provided to facilitate the addition of tabs by the QSE. The tab names are arbitrarily assigned by the QSE, and some of the cells are set up as drop down lists. The cells referred to for the drop down list are off to the side of the form and are formatted with a white color to make them less subject to tampering. No named ranges are set up on the these tabs; the version number cell is set up as described above for the Identification tab, and a cell is reserved in A2, to be filled in by ERCOT as a control number to be associated with the specific resource in any subsequent processing. The baseline options for the drop down list associated with cell C4 are set on the standard form posted for use at the beginning of the resource identification process and are filled in by SAS when outputting the approved form based on the baseline analysis performed by ERCOT. These cells are also formatted with a white text color to make them less obvious.

9		• 🗋 🗐 =	Layout Fo	rmulas Data	Review	ERIDS View	by Client C	oct12Jan13 I.xl	sx - Micros	oft Excel						
Pa	Clipboard	Arial		• A A		»	Wrap Text Merge & Cen	ter •	⁄₀ , *.0	.00 Cor	nditional Forma matting • as Table Styles	t Cell	Insert Delete Format	∑ AutoSum Fill ▼ Clear ▼ Ec	-	1 d &
	G8	- (0	f _x													≽
-	A	В	С	D	E	F	G	Н	I	J	K	L	М		N	-
1	ERID Number	Renewal Opt In	Resource Type	ERS Time Period	Include Time	Declared Self-Serve	Declared Injection	Capacity Offer (MV)	Price (\$/MV/hr)	Maximu m Base	MV Tied Offers					- Â
2			Load	Business Hours 1	Yes	Jen-Jerve	meetion	((********)	III Dase						
3	Resource Name		Baseline	Business Hours 2	Yes											
	American Eagle			Business Hours 3	Yes											
5				Non Business Ho												_
6	ESI-ID	Sub Metering	Jnique Meter II	LR at this Site	LR Participatin g	Site Owner Controlling Entity	Site Name	Site Address (street)	Site Address (citu)	Zip Code	TDSP Interconnection Agreement	Year of Agreement	TSPIDSP		TSP/DSP Duns #	s
7		YES	00115164	NO	2		Eagle	3101 Gaylord Parkw		75034			BRAZOS ELECTRIC POWER (CO OP INC (TDSP)	0037722904000	YES 🗏
8								<u> </u>								
9								[-11
10																
12																
13																
14																
15																_
16 17																_
18																
19																
20																
-14-4	🕨 🕨 🗌 Identific	ation 🖉 🥂 🦊	Alt America	n Eagle 🖉 Banke	ers Inc 🦯 🐮	1/							1			
Rea	dy 🛅													0% 😑 — 🕻	J(÷ .::

READING THE SUBMISSION FORM WITH SAS

The complete code ERCOT uses to read these workbooks is included in Appendix A. The remainder of this section will discuss what the code is doing step by step.

The first section of code, shown below, sets up several macro variables including version to compare against the ones on the submitted form to make sure the most recent version is being used. All forms to be processed are dropped into the folder specified by the '&offerpath' macro variable. The first data step creates 'fnames_list' by scanning the folder and selecting all Excel files found in the folder. The second data step creates a macro variable, '&numfiles', which contains the number of files to be processed.

🖏 SAS - [Read in ERID forms.sas]	
🚰 File Edit View Iools Run Solutions Window Help	- B ×
✓ ✓	
Explorer X 1	~
Contents of 'Work' 2 options symbolgen mlogic mprint linesize = 100 msglevel = i;	
Name 3	
Fnames_list 4	
Sasmacr 5 %let version = 12.3;	
6 %let contract_period = 2012 OctJan13;	
<pre>8 %let offerpath = d:\test; 9 %let reportpath = d:\test;</pre>	
9 %let reportpath = d:\test;	
12 libname perm "d:\test";	
13	_
14	
15 data fnames_list (Drop = rc fileid filecount i) ;	
16	
17 length filename \$ 300 ;	
18	
19 rc = filename("Input", "&offerpath.");	
20 fileid = dopen("Input");	
21 filecount = dnum(fileid); 22 DO i = 1 to filecount;	
24 filename = dread(fileid, i);	
<pre>25 if upcase(scan(filename,-1,'.')) in('XLS ', 'XLSM', 'XLSX') then Output;</pre>	
26	
27 end;	
28 rc = DClose(fileid);	
29	
30 run;	
31	
32 /* create a macro variable of the number of filenames in the folder */	
33 data _null_;	
34 35 set FNames List nobs = numfiles;	
35 set FNames_List nobs = numfiles; 36 call symputx("numfiles", put(numfiles,4.));	
37 stop;	
38	
39 run;	
40	~
	>
B Q Expl. Output - (Untitled)	
Z:Emergency_Interruptible_Load/Wonls datavalidation/Sas Code	29

Most of the rest of the code is nested inside a macro called 'read_work_book'. The major macro do-loop inside this macro cycles through the workbooks one at a time using the '&numfiles' macro variable created in the preceding step.

The libname statement shown below is an example of mapping to an Excel workbook using the Excel libname engine. The SAS explorer window for 'mylib' shows the tab names and named ranges for the particular Excel workbook ... the tab names end with dollar signs and named ranges do not. The tab name, wilkinson's, in the workbook shows up in the explorer window as wilkinson''s; double clicking on the table in the explorer window, however, generates a message that the table either does not exist or cannot be displayed.

🖏 SAS - [Editor - Untitled2 *]		
🕅 Eile Edit ⊻iew Tools Run Solutions Window	Help	_ 8 ×
· ~ · · · · · · · · · · · · · · · · · ·	🚰 🖬 🎒 🐧 🐰 🗈 🛍 🕫 🎁 🔯 🖈 🗙 🛈 🛷	
Explorer 🛛 🗙	1	~
Contents of 'Mylib'	<pre>2 libname mylib "d:\test\Offers - 10M ERS.xlsx" mixed = yes header = no;</pre>	
Name Size Type	3	
WILKIN&SON (2)\$' Table	4	
Identification\$ Table		
STATEFARM1\$ Table		
id_input_area Table		
wilkinson"s\$ Table		
		~
	2	>
🖬 Results 🧕 🔍	🗈 Output - (Untitled) 🖹 Log - (Untitled) 🕅 Read in ERID forms 🔀 Editor - Untitled2 *	
nterruptible_LoadWonIs datavalidation	as Code	Ln 4, Col 1

Below is a sample of the normal log output obtained when running the proc contents on the workbook; this is an example of the kind of errors issued when tab names containing special characters are processed.

🖏 SAS - [Log - (Untitled)]	
Eile Edit View Tools Solutions Window Help	- 8 ×
j ✓ j D 🛩 🖬 🖨 Q 🐰 🖻 🕮 🕫 😣 🗴 🥹 🏈	
Explorer I 603 proc contents data = myliball_ noprint; Contents of 'Work'	<u>^</u>
Name All_all_miss_id_ All_all_miss_id_ All_bad_tab_na All_bad_versior All_bad_versior	
All_miss_id_tab All_resources	~
	>
📴 Res 🔍 Explorer 🔛 Output - (Untitled) 📄 Log - (Untitled) 🛣 Read in ERID forms 🔀 Editor - Un	titled2 *
🛫 Z:\Emergency_Interruptible_Load\WonIs datavalidation\Sas Code	

The section of code shown below shows another way to generate a list of tab names by using proc datasets inside an ODS group to create a SAS dataset, 'tabnames', of the tabs and named ranges in the workbook. This method is the most successful I've found for producing a complete list of names while avoiding errors for tab names that contain special characters (', &, (,), space). A viewtable of 'tabnames' for a test workbook with many combinations of potential problem names is shown following the section of SAS code. In general, if SAS encounters a tab name with any of these

special characters other than a single quote, single quotes are added to the front and back of the tab name. If SAS encounters a single quote as part of the tab name, it turns the single quote into two single quotes.

The subsequent data step creates two datasets, 'tab_names_1' and 'bad_tab_names'. Tab names written to 'tab_names_1' that had a single quote within the tab name (that SAS doubled) are translated back to having a single quote, and single quotes added at the beginning and end of the tab name. Tab names that had a single quote and an '&' within the tab name are written to 'bad_tab_names'; I have not found a way to manipulate these kind of tab names to get SAS to read them. Further down in the code the 'bad_tab_names' data set is accumulated into the 'all_ bad_tab_names' to make available a list of all Excel files with problem tab names. For the time being, the listed workbooks are modified manually by ERCOT to eliminate the special characters and reprocessed. If the 'bad_tab_names' dataset is not empty, an error message is displayed in the log, and no other tabs in the workbook are processed.

The data step also excludes several tab names with restricted use for this application: the 'Availability', 'Baseline', 'Exceptions', 'Identification', etc tabs either are reserved for subsequent use by ERCOT or are simply excluded. If the submitter has set up a print range in the workbook, that too is excluded from the 'tab_names_1' data set.

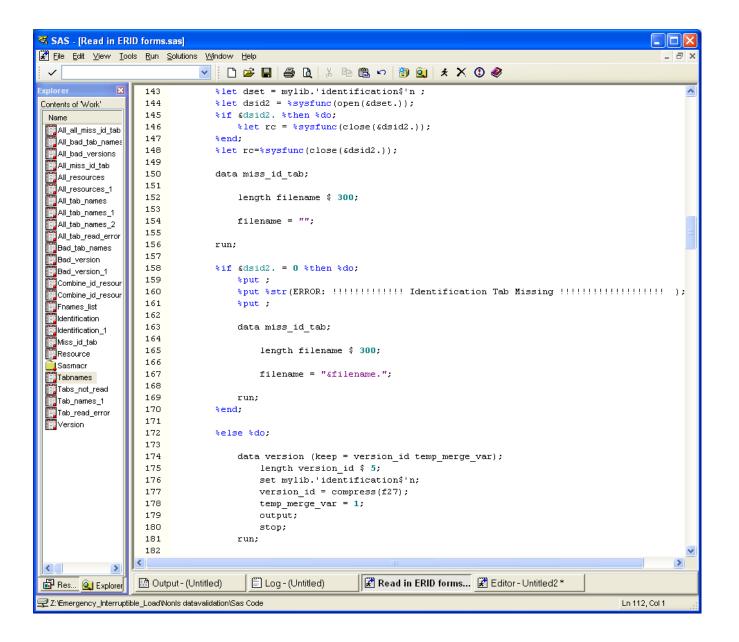
弦 SAS - [Read in ERID forms.sas] ¥ 1 🗋 🖆 📕 🎒 🐧 🐰 🖻 🛍 🗠 🎁 🔍 🖈 🗙 🛈 🥔 ^ 44 - %macro read work book; 45 46 %let first = 1; 47 %do i = 1 %to &numfiles.; 48 49 50 data _null_ ; 51 52 obsnum = input("&i.",4.); 53 set fnames List point = obsnum; 54 call symputx("filename", put(filename, \$300.)); 55 stop; 56 57 run; 58 59 libname mylib "&offerpath.\&filename." mixed = yes header = no; 60 61 ods trace on; ods output "Library Members" = tabnames; 62 63 64 proc datasets library = mylib; 65 quit; 66 67 ods output close; 68 ods listing; 70 %let numbad = 0; data tab_names_1 (keep = filename tab_name) 71 bad tab names (keep = filename tab name); 72 73 length tab name \$ 35 filename \$ 300; 74 set tabnames; 75 76 filename = "&filename.": 77 if index (name, "''') > 0 then tab name = "'' || trim(tranwrd(name, "''", "'")) || "'"; else tab_name = name; 78 79 80 if length(trim(tab name)) >= 4 then do; if index(substr(tab_name, 2, length(trim(tab_name)) = 2), "`") > 0 &81 82 index(substr(tab_name, 2, length(trim(tab_name)) - 2), "&") > 0 then do; 83 output bad_tab_names; 84 call symputx("numbad",1); 85 end; else do; 86 87 if upcase(substr(tab_name,1,4)) 88 in('AVAI', 'BASE', 'EXCE', 'ID_I', 'IDEN', 'R\$', 'SHEE') | index(tab_name,'Print_Area') > 0 then delete; 89 90 else output tab_names_1; 91 end: end; 92 else if upcase(tab name) $^{*}= {}^{*}R$; then output tab names 1; 93 94 95 run; 96 97 %if &numbad. = 1 %then %do; 98 %put 99 100 %put ; 101 end: > 🗒 Log - (Untitled) 🔀 Read in ERID forms... 💽 Results Viewer-file:/. 🔡 Output - (Untitled) 🖃 D: test Ln 85, Col 13 Autosave complete

😽 SAS -	VIEWTABLI	: Work.Tabnames (Library Members)]	
📴 Eile 🛛	<u>E</u> dit <u>∨</u> iew <u>T</u> oo	ls <u>D</u> ata <u>S</u> olutions <u>W</u> indow <u>H</u> elp		_ 8 ×
		💽 🔁 🖬 🎒		> × ↓ª ↓Z 🛅 ≣
	Variable Number	Name	Member Type	DBMS Member
1	1	'STATE FARM1\$'	DATA	TABLE
2	2	'WILKIN&SON"s\$'	DATA	TABLE
3	3	'WILKINSON(2)\$'	DATA	TABLE
4	4	'abc\$'	DATA	TABLE
5	5	'a&b(2) c\$'	DATA	TABLE
6	6	'a"b&c d\$'	DATA	TABLE
7	7	'a"b&c(2)\$'	DATA	TABLE
8	8	'a"sb&c(2) d\$'	DATA	TABLE
9	9	'a(1) b\$'	DATA	TABLE 🗧
10	10	'a(1)b(2)\$'	DATA	TABLE
11	11	'state&farm\$'	DATA	TABLE
12	12	'wilk &son\$'	DATA	TABLE
13	13	'wilk&so&n\$'	DATA	TABLE
14	14	'wilk&son(2)\$'	DATA	TABLE
15	15	'wilkin son"s\$'	DATA	TABLE
16	16	'wilkinson"s(2)\$'	DATA	TABLE
17	17	Identification\$	DATA	TABLE
18	18	a"sb"s\$	DATA	TABLE
19	19	id_input_area	DATA	TABLE
20	20	wilkinson"s\$	DATA	TABLE
<				
🔡 Outp	out - (Untitl	🖹 Log - (Untitled) 🛛 🛣 Read in ERID	🔊 Results View	🖼 VIEWTABLE
			🗐 D: itest	:

mand	VIEWTABLE: Work.Ta	b_names_1]	
📴 File I	<u>E</u> dit <u>V</u> iew <u>T</u> ools <u>D</u> ata <u>S</u> o	olutions <u>Wi</u> ndow <u>H</u> elp	- 8 ×
 Image: A second s		💽 🔽 🖾 🖉 📕 🔮	3 🖪 [
	tab_name	filename	~
1	'STATE FARM1\$'	Offers - 10M ERS.xlsx	
2	WILKINSON(2)\$'	Offers - 10M ERS.xlsx	
3	'abc\$'	Offers - 10M ERS.xlsx	
4	'a&b(2) c\$'	Offers - 10M ERS.xlsx	
5	'a(1) b\$'	Offers - 10M ERS.xlsx	
6	'a(1)b(2)\$'	Offers - 10M ERS.xlsx	
7	'state&farm\$'	Offers - 10M ERS.xlsx	
8	'wilk &son\$'	Offers - 10M ERS.xlsx	
9	'wilk&so&n\$'	Offers - 10M ERS.xlsx	
10	'wilk&son(2)\$'	Offers - 10M ERS.xlsx	
11	"wilkin son's\$"	Offers - 10M ERS.xlsx	
12	"wilkinson's(2)\$"	Offers - 10M ERS.xlsx	
13	'a'sb's\$'	Offers - 10M ERS.xlsx	
14	'wilkinson's\$'	Offers - 10M ERS.xlsx	~
<			>
🔡 Outp	out 🖺 Log - (🕅 🕅 Re	ad i	EW
		🖃 D: \test	

😽 SAS -	VIEWTABLE: Work.Bad_tab_name	esj 📃 🗖 🔀					
📴 Eile 🛛	<u>E</u> dit <u>V</u> iew <u>T</u> ools <u>D</u> ata <u>S</u> olutions <u>W</u> indo	w Help _ a ×					
 ✓ 	▼	🗅 🚅 📕 🎒 🛕 🖻 🖬 🖉 🗠 🗙					
	tab_name	filename 🔼					
1	"WILKIN&SON's\$"	Offers - 10M ERS.xlsx					
2	"a'b&c d\$"	Offers - 10M ERS.xlsx					
3	"a'b&c(2)\$"	Offers - 10M ERS.xlsx					
4	"a'sb&c(2) d\$"	Offers - 10M ERS.xlsx					
<							
🔡 Outp	🗈 Output 🖹 Log - (U 🗷 Read in 💽 Results 🖼 VIEWT 🖼 VIEWT						
NOTE: You	cannot shrink the window beyond this point.	🖃 D:'test					

The next section of code, shown below, checks whether the required 'Identification' tab is present in the workbook. If it is not present, an error message is displayed in the log, and no other tabs in the workbook are processed. The filenames of workbooks with a missing identification tab are output to the 'miss_id_tab' dataset; further down in the code the 'miss_id_tab' data set is accumulated into the 'all_miss_id_tab' to make available a list of all Excel files with missing 'Identification' tabs. If the 'Identification' tab is present, the 'version' data set is created; this data step reads just the first row of the 'Identification' tab and keeps the version number stored in cell AA1.



The next section of code inputs identification information from the 'id_input_area' named data range; the range is two columns wide, and the data needed is in the second column of some of the rows in the range (the other rows are cosmetic merged cells to make the sections of the form stand out clearly). The QSE's name is entered on the first row of the range, and the QSE's Duns number is entered on the second row of the range. The remainder of the range contains contact information for the QSE and is read into the 'id_info' array. Note that because the headers option was set to no on the libname statement, the incoming variable of interest is always 'f2'. Since each row of the workbook is treated as an observation by SAS, all needed variables are retained until the last row of the range has been read at which point the observation is output to the 'identification' dataset.

The 'identification_1' dataset is the merge of the identification information with the version number, and the final data step in this section of code creates a macro variable, '&numtabs', for the number of tabs in the workbook. This macro variable is used in the macro do loop to sequence through the tabs found in the workbook.

😽 SAS - [Read in ER			X
<u> <u> </u> </u>	ols <u>R</u> un <u>S</u> olutions <u>W</u> indow	Help _ C	r ×
· ·	D) 😂 🖬 🎒 🐧 🐇 ங 🛍 🕫 🔯 🔕 🛪 🗙 🕐 🛷	
Explorer 🔀	183 0	data identification (keep = qsename qseduns temp_merge_var id_info1 - id_info23);	. 🔨
Contents of 'Work'	184		
Name	185	length qsename qseduns id_info1 - id_info23 \$64.;	
All all miss id tab	186	array id_info(23) \$ id_info1 - id_info23;	
All_bad_tab_names	187	retain qsename qseduns id_infol - id_info23;	
All_bad_versions	188		
All_miss_id_tab	189	set mylib.'id_input_area'n ;	
All_resources	190		
All_resources_1	191	if _n = 1 then qsename = f2;	
All_tab_names	192	else if _n = 2 then qseduns = f2;	
All_tab_names_1	193	if $(\underline{n} \ge 3 \& \underline{n} \le 4) (\underline{n} \ge 6 \& \underline{n} \le 11) (\underline{n} \ge 13 \& \underline{n} \le 17)$	_
All_tab_names_2	194 195	$ (_n \ge 19 \& _n <= 23) (_n \ge 25 \& _n <= 29)$ then do; i + 1;	=
All_tab_read_error	195	id info(i) = f2;	
Bad_tab_names	197	if i = 2 then id_info2 = upcase(id_info2);	
Bad_version	197	temp merge var = 1;	
Bad_version_1	199	if $i = 23$ then output;	
Combine_id_resour	200	end:	
Combine_id_resour		run;	
Identification	202		
Identification_1	203 0	data identification 1;	
Miss_id_tab	204	merge identification version;	
Resource	205	by temp_merge_var;	
Sasmacr	206 1	run;	
Tabnames	207		
Tabs_not_read	208 0	data _null_;	
Tab_names_1	209	set tab_names_1 nobs = numtabs;	
Tab_read_error	210	put numtabs=;	
Version	211	call symputx("numtabs",put(numtabs,4.));	
	212	stop;	
	213		
	214 1	run;	~
< >			<u> </u>
📴 Res 🔍 Explorer	🔄 Output - (Untitled)	Editor - Untitled Read in ERID forms	
쿶 Z:\Emergency_Interrupt	- ible_Load\NonIs datavalidation\Sa	as Code Ln 213, Col 17	

The next section of code shows the beginning of the macro do loop; the first data step does a direct access read of the 'tab_names_1' data set, based on the iteration of the macro do loop, to create a macro variable, '&sheet', containing the 'sheet' statement to be used with the subsequent proc import step. If the modified tab name is enclosed in single quotes and also contains a single quote, the 'sheet' statement is created with double quotes to allow the proc import to read that particular tab. Proc import is the only way I've found to read these tabs. Note again that the proc import uses the following options: 'mixed = yes getnames = no and scantext = yes'. The 'mixed = yes' option tells SAS that all cells are to be treated as characters ... all conversions to numeric variables are handled in the subsequent data step. The 'getnames = no' option tells SAS that data begins on the first row of the workbook; otherwise SAS would interpret the first row as containing variable names. The 'getnames = no' option causes SAS to use automatic variable names: column A becomes F1, column B becomes F2, etc. The 'scantext = yes' option tells SAS to scan each column to determine the maximum length to use for each character variable.

```
😽 SAS - [Read in ERID forms.sas]
                                                                                                                        Eile Edit View Tools Run Solutions Window Help
                                                                                                                           æ
                              🔽 🗋 🗃 🔚 🚇 🐧 🐰 🖻 🏙 🗠 🎁 🚉 🖈 🗶 🛈 🥔
  \checkmark
             %do j = 1 %to &numtabs.;
  181
                                                                                                                             ~
  182
  183
               data tab_names_2 (keep = tab_name temp_merge_var) ;
  184
  185
                 length tab_name_quote $ 300;
  186
                 obsnum = input("&j.",4.);
  187
  188
                 set tab_names_1 point = obsnum;
  189
                 if substr(tab name, 1, 1) = "'" & substr(tab name, length(trim(tab name)), 1) = "'"
  190
                   & index(substr(tab_name, 2, length(trim(tab_name)) - 2), "'") > 0 then do;
  191
  192
                   tab name_quote = 'sheet = "' || trim(put(tab_name,$300.)) || '"';
                   call symputx("sheet", tab_name_quote);
  193
  194
                 end;
  195
                 else if substr(tab_name, 1, 1) = "'" & substr(tab_name, length(trim(tab_name)), 1) = "'" then do;
  196
  197
                   tab_name_quote = "sheet = " || trim(put(tab_name,$300.));
  198
                   call symputx("sheet", tab_name_quote);
  199
                 end;
  200
  201
                 else do;
                   tab name quote = "sheet = '" || trim(put(tab name, $300.)) || "'";
  202
  203
                   call symputx("sheet", tab_name_quote);
  204
                 end;
  205
  206
                 temp_merge_var = 1;
  207
                 output;
  208
                 stop;
  209
  210
               run;
  211
  212
  213
               proc import datafile="&eridpath.\&filename." replace out = import_resource dbms = excel;
  214
                 mixed = ves;
  215
                 getnames = no;
  216
                 scantext = ves;
  217
                 Ssheet.:
  218
               run;
  219
 220
               data import resource 1;
  221
  222
                   set import resource;
 223
  224
                   temp_merge_var = 1;
  225
 226
               run:
                                                                                                                          >
                      🗒 Log - (Untitled)
🔡 Output - (Untitled)
                                           Read in ERID forms... 💽 Results Viewer - SA.
File saved successfully
                                                                                    🖃 D: test
                                                                                                                Ln 203, Col 43
```

The next data step parses and converts the data imported from each tab in the workbook and outputs it to the 'resource' data set; the code is set up to pull data from the f-variable associated with the appropriate cell locations based on whether the tab being input is an 'Alt' tab or a standard resource tab. The _n_ automatic SAS variable in combination with the f1 – f27 variables are used as positions within the row to define each of the variables being read from the workbook ... cells with labeling information and cells which are intended to be blank for aesthetic purposes are skipped over. As mentioned above, because of the 'mixed = yes' option, all variables were read in as characters. In this step, they are assigned to variables kept in the output dataset with the lengths of character variables controlled specifically by the code for each variable and with numeric variables, if any, converted using the input function. As with the 'Identification' tab, since each row of the workbook is treated as an input observation by SAS, all needed variables are retained until the last row needed for the observation is output to the 'resource' dataset. Observations are output for each incoming row with customer data provided that the esild or unique_meter_id variables are greater than blank.

As with the 'Identification' tab, a version number for each tab is read from cell AA1 and used for subsequent validation to identify whether the QSE is using the current workbook version.

😽 SAS -	Read in ERID forms.sas]	
🛃 Eile Eo	litt ⊻iew Iools Run Solutions Window Help	_ @ ×
-	💌 📄 🗁 🖬 / 🚝 🔃 🖄 🖻 🎕 🗠 / 🕲 💁 🖈 🗙 📀 🛷	
228	data resource (keep = version_resource filename tab_name renewal_opt_in resource_type	~
229	resource_name baseline analyze_offer1 - analyze_offer4	
230	esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name	
231	street city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns	
232	noie_area wsale_meter priv_net onsite_gen gen_type load_desc	
233	temp_merge_var eridnum rundate);	
234		
235	length version_resource \$ 5 filename \$ 300 tab_name \$ 35	
236	resource_type resource_name baseline \$ 64 renewal_opt_in \$ 3	
237	esiid \$ 64 sub_metering \$ 3 unique_meter_id \$ 64 load_resource lr_in_as \$ 3	
238	site_ownr site_name \$ 64 street city $\$64$ zipcode $\$5$. tspdsp_ia $\$$ 3	
239	tspdsp_ia_year tspdsp_name tspdsp_duns \$64. noie_area wsale_meter	
240	priv_net onsite_gen \$3. gen_type \$30. load_desc \$256.	
241	analyze_offer1 - analyze_offer4 \$ 3;	
242		
243	format rundate datetime20.;	
244		
245	retain version_resource_temp_merge_var_filename_tab_name_renewal_opt_in_resource_type	
246	resource_name baseline analyze_offer1 - analyze_offer4 eridnum rundate;	
247	waves impact seconds 1 teb news 21	
248 249	merge import_resource_1 tab_names_2;	
249	by temp_merge_var;	
250	if tab name = 'Alt' then baseline = 'ALTERNATE';	
252	if $n = 1$ then do;	
252	temp merge var = 1;	
254	*tab name = substr("&sheet.", 9, 1);	_
255	tab name = translate(tab name, ", ", "§");	
256	filename = "Sfilename.";	
257	rundate = datetime();	
258	eridnum = round(100 * (rundate - "01jun2009:01:00:00"dt),1);	
259	version resource = f27;	
260	end;	
261	else if $n \ge 2 \& n \le 5 \&$ tab name ^= 'Alt' then do;	
2 6 2	if $n = 2$ then do;	
2.63	renewal opt in = '';	
264	resource_type = f3;	
265	analyze_offer1 = upcase(f5);	
266	end;	
267	else if _n_ = 3 then analyze_offer2 = upcase(f5);	
2.68	else if $n_{-} = 4$ then do;	
269	resource_name = f1;	
270	if resource_name = '' then resource_name = 'Missing Resource Name';	
271	<pre>baseline = upcase(f3);</pre>	
272	analyze_offer3 = upcase(f5);	
273	end;	
274	else if _n_ = 5 then analyze_offer4 = upcase(f5);	
275	end;	
276	else if (tab name = 'Alt' & n >= 4) (tab name ^= 'Alt' & n >= 7) then do:	<u>×</u>

File saved successfully.	D:\test Ln 221, Col	1 _;;
🔡 Output - (Untitled)	E Log - (Untitled)	
<		>
308	run;	~
307	end;	
306	<pre>put /// 'esiid and umi both blank ' filename = tab_name = /// '';</pre>	
305	gen_type > '' load_desc > '') then	
304	noie_area > '' wsale_meter > '' priv_net > '' onsite_gen > ''	
303	tspdsp_ia > '' tspdsp_ia_year > '' tspdsp_name > '' tspdsp_duns > ''	
302	site_ownr > '' site_name > '' street > '' city > '' zipcode > ''	
301	د (sub_metering > '' unique_meter_id > '' load_resource > '' lr_in_as > ''	
300	else if esiid = '' & unique meter_id = ''	
299	if \vec{s} if \vec{s}' unique meter id > '' then output;	
298	load desc = f20;	
297	gen type = f19;	
296	onsite gen = upcase(f18);	_
295	priv net = upcase(f17);	
294	wsale meter = upcase(f16);	_
293	noie area = upcase(f15);	
292	tspdsp_duns = f14;	
290	tspdsp name = f13;	
290	tspdsp ia year = f12;	
289	tspdsp ia = upcase(f11);	
288	zipcode = f10;	
287	street - 10; city = f9;	
285 286	site_name = f7; street = f8;	
284	site_ownr = f6;	
283	<pre>lr_in_as = upcase(f5);</pre>	
282	<pre>load_resource = upcase(f4); load_resource = upcase(f4);</pre>	
281	then sub_metering = 'NO';	
280	if esiid > '' & unique_meter_id = '' & sub_metering = ''	
279	unique_meter_id = f3;	
278	sub_metering = upcase(f2);	
277	<pre>esiid = compress(f1, "' ");</pre>	<u> </u>

The next section of code determines the number of observations in 'resource' dataset and if the number is greater than zero merges the identification and resource information. The 'combine_id_resource_1' dataset contains observations found with valid version numbers, and the 'bad_version_1' dataset identifies resources with invalid version numbers.

```
😽 SAS - [Read in ERID forms.sas]
File Edit ⊻iew Tools Run Solutions Window Help
                                                                                                            E.
                               🔽 🗄 🚔 🖶 🎒 🐧 🐰 🖻 🛍 🕫 🎒 🚉 🖈 🗙 🕄 🥔
  \checkmark
  307
                        %let dset = work.resource;
                                                                                                               ^
                        %let dsid3 = %sysfunc(open(&dset));
  308
  309
  310
                        %if &dsid3 %then %do;
  311
  312
                            %let numloads =%sysfunc(attrn(&dsid3,NOBS));
  313
                            %let rc = %sysfunc(close(&dsid3));
  314
  315
                        %end:
  316
  317
                        %let rc=%sysfunc(close(&dsid3));
                        %if &numloads. > 0 %then %do;
  318
  319
  320
                            data combine id resource bad version (keep = qsename eridnum sub eridnum);
  321
  322
                                merge identification_1 (in = in1) resource (in = in2);
  323
                                by temp merge var;
  324
  325
                                if ^in1 | ^in2 | compress(version_id) ^= "&version."
                                     | compress(version resource) ^= "&version." then do;
  326
  327
                                     output bad_version;
                                     call symputx ("version error", "Yes");
  328
  329
                                end:
  330
                                output combine_id_resource;
  331
                            run:
  332
  333
                            proc sort data = bad version nodupkey;
  334
                                by eridnum sub eridnum;
  335
                            run;
  336
  337
                            proc sort data = combine_id_resource;
  338
                                by eridnum sub_eridnum;
  339
                            run:
  340
  341
                            data combine id resource 1 bad version 1;
  342
                                merge combine_id_resource bad_version (in = in1);
  343
                                by eridnum sub_eridnum;
                                if ^in1 then output combine_id_resource_1;
  344
  345
                                else output bad version 1;
  346
                            run;
                                                                                                            >
                      🗒 Log - (Untitled)
                                            🗷 Read in ERID forms... 🗷 Editor - Untitled2 *
🔡 Output - (Untitled)
🛃 Z:\Emergency_Interruptible_Load\NonIs datavalidation\Sas Code
                                                                                                  Ln 340, Col 1
```

The final piece of code in the 'read_work_book' macro accumulates all the resource and error data sets across all workbooks and tabs within workbooks. Finally, the code clears the libname assigned to the Excel workbook. An additional section of code in a macro called 'check_errors' is shown in the Appendix; this code is included to display error messages at the end of the log. The messages trigger investigations into problems with any of the workbooks processed; in many cases the volume of workbooks and tabs produced is very large causing the log to be very long and earlier error messages generated to be easily overlooked.

😽 SAS - [Read in ERID forms.sas] - 0 🔀 File Edit View Tools Run Solutions Window Help _ 8 × 🔽 🗋 🗃 🖬 🖨 🐧 🐰 🖻 🛍 🗠 🎁 🖳 🖈 🗙 🛈 🥔 \checkmark 351 %if &first. = 1 %then %do; ^ 352 353 data all resources; 354 set combine_id_resource_1; 355 run; 356 357 data all bad versions; 358 set bad version 1; 359 run; 360 361 data all_tab_names; 362 set tab names 1; 363 run; 364 365 data all_bad_tab_names; 366 set bad tab names; 367 run; 368 369 data all miss id tab; 370 set miss id tab (where = (filename > '')); 371 run; 372 373 374 %let first = 0; 375 376 %end; 377 %else %do; 378 379 proc append base = all resources data = combine id resource 1; 380 run; 381 proc append base = all_bad_versions data = bad_version_1; 382 383 run; 384 proc append base = all tab names data = tab names 1; 385 386 run; 387 388 389 %end. 390 %end; 391 %end; 392 %end;

393		^
394	proc append base = all_bad_tab_names data = bad_tab_names;	
395	run;	
396		
397	proc append base = all_miss_id_tab data = miss_id_tab (where = (filename > ''));	
398	run;	
399		
400	<pre>%end;</pre>	
401		
402	&mend	_
403		-
404	<pre>% read_work_book;</pre>	
405		
406	libname mylib clear;	~
<		
🔡 Outp	out - (Untitled) 📔 Log - (Untitled) 🕅 Read in ERID forms 💽 Results Viewer - SA	
	🖃 D:\test Ln 350, Col 1	

WRITING OUT PROCESSED FORMS

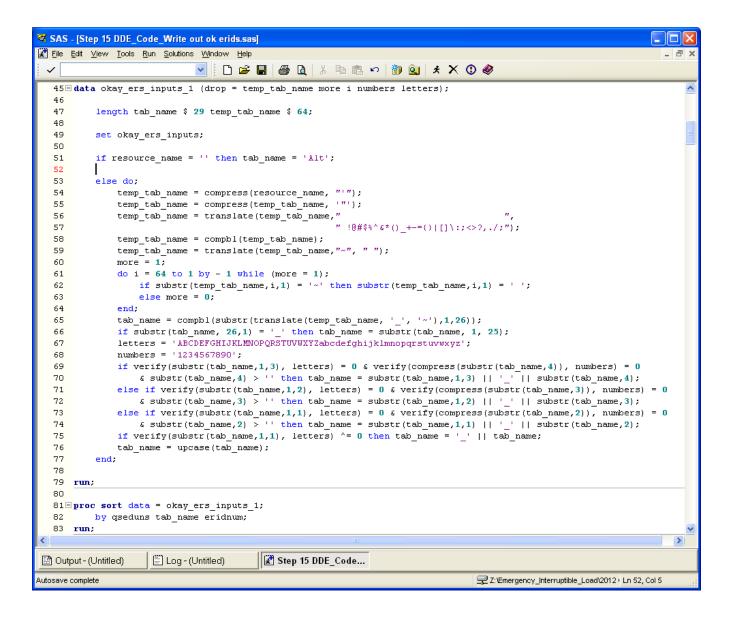
Resources that are read and output with the code described in the previous section proceed to an extensive validation step, and if successful, continue with historical usage data analysis. SAS datasets summarizing the validation and analysis are produced and stored for later use when approved forms are created to be returned to the QSEs. These forms are then completed by QSEs and returned to ERCOT in the form of an offer to provide the service.

The complete code for writing out workbooks is contained in Appendix B. The process of writing out the approved workbooks consists of using the Excel libname engine to move data from SAS datasets to Excel and using DDE commands to open, save and close workbooks, and to execute various user defined Excel macros.

To get started, the options command sets the noxwait and noxsync options; this is needed to get the DDE commands to function properly. Some macro variables are set up to establish paths to the permanent SAS datasets produced by the validation and analysis steps and a path to the Excel templates used to generate the workbooks. The data step to create 'okay_ers_inputs' is fairly straight forward; code to take note of is the creation of the 'blout1 – blout4' variables ... these are used later on to populate a drop-down list in the workbook to provide baseline options for the specific resource. When the QSE returns the form, one of these allowed options must be selected.

<pre></pre>	द्ध sas	- [Step 15 DDE_Code_Write out ok erids.sas]	
<pre>c options nowait nowayne symbolgen mlogic mprint linesize = 200 maglevel = 1;</pre>	🖌 Eile	Edit <u>V</u> iew <u>T</u> ools <u>R</u> un <u>S</u> olutions <u>W</u> indow <u>H</u> elp	- 8 ×
<pre>ilet contract_period = 2012 OctJan1; ilet contract_period = 2012 OctJan1; ilet cal_dtrive d; i enter the drive letter for the templates and output path ; i path to write x1s files and to see if they already exist for this batch */ i et cridpath = Emergency_Interruptible_Load/contract_period.\00 resource Identification\30 Templates; i have been write x1s files and to see if they already exist for this batch */ i et cridpath = Emergency_Interruptible_Load/contract_period.\00 resource Identification\30 Templates; i have been write x1s files and count of eride */ /**********************************</pre>	~	🔽 🗋 🖆 🖬 🎒 🐧 🐇 🖻 🛍 🕫 🔯 🕺 🛠 🕐 🧶	
<pre>ilet contract_period = 2012 OctJan1; ilet contract_period = 2012 OctJan1; ilet cal_dtrive d; i enter the drive letter for the templates and output path ; i path to write x1s files and to see if they already exist for this batch */ i et cridpath = Emergency_Interruptible_Load/contract_period.\00 resource Identification\30 Templates; i have been write x1s files and to see if they already exist for this batch */ i et cridpath = Emergency_Interruptible_Load/contract_period.\00 resource Identification\30 Templates; i have been write x1s files and count of eride */ /**********************************</pre>	2	options noxwait noxsync symbolgen mlogic mprint linesize = 200 msglevel = i;	~
<pre>s tiet contract_period = 2012 OctJan3; s tiet local_drive = d; s tiet local_drive = d; s enter the drive letter for the templates and output path ; s enter the drive letter for the templates and output path ; s enter the drive is the set of th</pre>			
<pre> filet local_drīve = d:;</pre>	4		=
<pre>/* /* path to write xls files and to see if they already exist for this batch */</pre>	5	<pre>\$let contract_period = 2012 OctJan13;</pre>	_
<pre>/* path to write xls files and to see if they alrendy exist for this back */ * let eridpath = Emergency_Interruptible_Load\contract_period.\00 resource Identification\D0E Templates; * lit templatepath = Emergency_Interruptible_Load\contract_period.\00 resource Identification\D0E Templates; * lithname perm *2:\Emergency_Interruptible_Load\contract_period.\00 resource Identification\D0E Templates; * oreat dataset of genames and count of erids */ * oreat dataset of genames and count of erids */ * oreat dataset of genames and count of erids */ * oreat dataset of genames and count of erids */ * oreat dataset of genames and count of erids */ * oreat dataset of genames and count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count of erids */ * oreat dataset of gename sand count * or or in a * 3 resource train as \$ a site_own site * and * ore offer + analyze offer + 3 blout - blout + blout + blout * ends */ * oreat sand pre_offer + and pre_ofde \$ 5 taspdap ia year taspdap mane sand was \$ 64 * noise area vas ends * offer * and * or or * \$ of a since * \$ a gen_type \$ 30 load_desc \$ 236 * originate * or in * * * or *</pre>		<pre>%let local_drive = d:;</pre>	
<pre>9 \$1et eridpath = Emergency_Interruptible_Load/scontract_period.\00 resource Identification\3 Outpoing erid\$\okay; 10 \$1et templatepath = Emergency_Interruptible_Load/scontract_period.\00 resource Identification\DE Templates; 11 12 1ibname perm "2:\Emergency_Interruptible_Load/scontract_period.\00 resource Identification\DE Templates; 13 14 15 /* create dataset of genames and count of erids "/ 15 /* create dataset of genames and count of erids "/ 17 data okay_ers_inputs (keep = eridnum gename gedums renewal_opt_in resource_name resource_type 10 load_type baseline analyze_offer1 - analyze_offer4 blout1 - blout4 19 load ss_gen non_ss_gen esild sub_metering unique_meter_id load_resource 10 lr_in_as site_owns site_name site_name site_ode tsydap_ia_tsydar 10 tsydap_name_tsydap_dums noi_area wsale_meter priv_net onsite_gen gen_type load_desc 11 d_infol_id_info23); 12 13 14 length eridnum & gename gedums & 64 renewal_opt_in \$ 3 resource_type resource_name baseline \$ 64 25 analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout5 - blout4 load ss_gen non_ss_gen \$ 30 26 esid \$ 64 sub_metering \$ 3 unique_meter_id \$ 64 load_resource_type resource_name baseline \$ 64 27 analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout5 - blout4 load ss_gen non_ss_gen \$ 30 28 oncie_area wsale_meter priv_net onsite_gen \$ 3 gen_type \$ 30 load_desc \$ 256 29 id_infol - id_infol \$ 64; 30 oncie_area wsale_meter priv_net onsite_gen \$ 3 gen_type \$ 30 load_desc \$ 256 30 ss_gen = 'Self-Serving Gen'; 30 if bline2 = 2 then blout3 = 'DFFAULT-HERGRESSIGN'; 31 if bline2 = 2 then blout3 = 'DFFAULT-HERGRESSIGN'; 33 if bline3 = 3 then blout3 = 'DFFAULT-HERGRESSIGN'; 34 if bline2 = 2 then blout3 = 'DFFAULT-HERGRESSIGN'; 35 if bline3 = 3 then blout3 = 'DFFAULT-HERGRESSIGN'; 36 if bline3 = 3 then blout3 = 'DFFAULT-HERGRESSIGN'; 37 if bline4 = 'AbtIERMATE'; 40 blout4 = 'LATERMATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 43 run; 44 cumput-(Untiled) Eleg-(Untiled) Eleg-(Untiled) Eleg-(Dete)</pre>			
<pre>i i tet templatepath = Emergency_Interruptible_Load\scontract_period.\00 resource Identification\bDE Templates; i libname perm "2:\Emergency_Interruptible_Load\scontract_period.\00 resource Identification\bas datasets"; /************************************</pre>			
<pre>11 libname perm "Z:\Emergency_Interruptible_Load\&contract_period.\00 resource Identification\sas datasets"; 12 libname perm "Z:\Emergency_Interruptible_Load\&contract_period.\00 resource Identification\sas datasets"; 13 //</pre>			
<pre>12 libname perm "Z:\Emergency_Interruptible_Load\&contract_period.\00 resource Identification\sas datasets"; 14 15 16 17 17 18 16 17 17 18 16 17 17 18 10 10 19 10 10 10 10 10 10 10 10 10 10 10 10 10</pre>		Teo campiacepach - Emergency_Incertapriste_isaa(aconcraco_period.)oo resource Taencificacion(bbr Tempiaces	<i>,</i>
<pre>/************************************</pre>		libname perm "Z:\Emergency Interruptible Load\&contract period.\00 resource Identification\sas datasets";	
<pre>15 /* create dataset of genames and count of erids */ 16 /************************************</pre>	13		
<pre>16 /************************************</pre>	14	/**************************************	
<pre>170 data okay_ers_inputs (keep = eridnum qsename qseduns renewal_opt_in resource_name resource_type 18</pre>	15	-	
<pre>18</pre>			
<pre>19 load ss_gen non_ss_gen esid sub_metering unique_meter_id load_resource 20 lr_in_as_site_ownr_site_name_site_name_street city zipcode tspdsp_ia_tspdsp_ia_year 21 tspdsp_name_tspdsp_duns_noie_area_wsale_meter priv_net_onsite_gen_gen_type_load_desc 22 id_info1 - id_info23); 23 length eridnum 8 gsename gseduns \$ 64 renewal_opt_in \$ 3 resource_type resource_name baseline \$ 64 26 analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen \$ 30 27 street city \$ 64 zipcode \$ 5 tspdsp_ia \$ 3 tspdsp_ia_year tspdsp_name_tspdsg_duns \$ 64 28 noie_area wsale_meter priv_net_onsite_gen \$ 3 gen_type \$ 30 load_desc \$ 256 29 id_info1 - id_info23 \$ 64; 30 31 set perm.ers_inputs_okay_final; 32 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen 'Non Self-Serving Gen'; 37 if bline2 = 2 then blout2 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout3 = 'DEFAULT-REGRESSION'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'LITENATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTENATE'; 43 run; 44 Comput-(Unwided) Elog-(Unwided) Elog-(Unwid</pre>			
<pre>20 lr_in_as site_ownr site_name site_name street city zipcode tspdsp_ia tspdsp_ia_year 21 tspdsp_name tspdsp_duns noie_area wsale_meter priv_net onsite_gen_gen_type load_desc 22 id_infol - id_info23); 23 length eridnum 8 gsename gseduns § 64 renewal_opt_in § 3 resource_type resource_name baseline § 64 25 analyze_offer1 - analyze_offer4 § 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen § 30 26 esiid § 64 sub_metering § 3 unique_meter_id § 64 load_resource lr_in_as § 3 site_ownr site_name 27 street city § 64 zipcode § 5 tspdsp_ia § 3 tspdsp_ia_year tspdsp_name tspdsp_duns § 64 29 noie_area wsale_meter priv_net onsite_gen § 3 gen_type § 30 load_desc § 256 29 id_infol - id_info23 § 64; 30 set perm.ers_inputs_okay_final; 31 set perm.ers_inputs_okay_final; 32 renewal_opt_in = ''; 33 fenewal_opt_in = ''; 34 load = 'load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gene 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout3 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-HID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-HID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-HID 8 OF 10'; 39 if bline3 = 1 then blout3 = 'DEFAULT-HID 8 OF 10'; 39 if bline3 = 1 then blout3 = 'DEFAULT-HID 8 OF 10'; 39 if bline3 = 0 then blout3 = 'DEFAULT-HID 8 OF 10'; 39 if bline3 = 0 then blout3 = 'DEFAULT-HID 8 OF 10'; 39 if bline3 = 0 then blout3 = 'DEFAULT-HID 8 OF 10'; 39 if bline3 = 0 then blout3 = 'DEFAULT-HID 8 OF 10'; 39 if bline3 = 0 then blout3 = 'DEFAULT-HID 8 OF 10'; 30 if bline3 = 0 then blout3 = 'DEFAULT-HID 8 OF 10'; 31 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 43 run; 43 run; 44 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 44 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 45 run;</pre>			
<pre>21 tspdsp_name tspdsp_duns_noie_area wsale_meter priv_net onsite_gen_gen_type load_desc 22 id_infoi - id_info23); 23 length eridnum 8 gsename gseduns \$ 64 renewal_opt_in \$ 3 resource_type resource_name baseline \$ 64 26 analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen \$ 30 27 street city \$ 64 zipcode \$ 5 tspdsp_ia \$ ds load_resource tri_n as \$ 3 site_ownr site_name 27 street city \$ 64 zipcode \$ 5 tspdsp_ia \$ 3 tspdsp_ia_year tspdsp_name tspdsp_duns \$ 64 28 noie_area wsale_meter priv_net onsite_gen \$ 3 gen_type \$ 30 load_desc \$ 256 30 set perm.ers_inputs_okay_final; 31 set perm.ers_inputs_okay_final; 32 renewal_opt_in = ''; 33 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MID 8 OF 10'; 41 if baseline = '' \$ bline1 = . \$ bline2 = . \$ bline3 = . then baseline = 'ALTERNATE'; 43 run; 44 if baseline = '' \$ bline1 = . \$ bline2 = . \$ bline3 = . then baseline = 'ALTERNATE'; 43 run; 44 if load. Elog-(Untitled) Elog-(Untitled) Elog-Code</pre>			
<pre>id_info1 - id_info23); id_info1 - analyze_offer4 \$ 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen \$ 30 id_info1 - analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen \$ 30 id_info1 - analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen \$ 30 id_info1 - analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen \$ 30 id_info1 - analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen \$ 30 id_info1 - analyze_offer1 - analyze_offer4 \$ 3 tspdsp_ia_year tspdsp_name tspdsp_duns \$ 64 note_area wsale_meter priv_net onsite_gen \$ 3 gen_type \$ 30 load_desc \$ 256 id_info1 - id_info23 \$ 64; id_info1 - id_info23 \$ 66; id_info1 - id_info23 \$ 66;</pre>			
<pre>23</pre>			
<pre>25 analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen \$ 30 26 esiid \$ 64 sub_metering \$ 3 unique_meter_id \$ 64 load resource lr_in_as \$ 3 site_ownr_site_name 27 street city \$ 64 zipcode \$ 5 tspdsp_ia \$ 3 tspdsp_ia_year_tspdsp_name_tspdsp_duns \$ 64 28 noie_area wsale_meter priv_net_onsite_gen \$ 3 gen_type \$ 30 load_desc \$ 256 29 id_info1 - id_info23 \$ 64; 30 31 set perm.ers_inputs_okay_final; 32 33 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen= 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MID 8 OF 10'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 43 run; </pre>			
<pre>25 analyze_offer1 - analyze_offer4 \$ 3 blout1 - blout4 blout1 - blout4 load ss_gen non_ss_gen \$ 30 26 esiid \$ 64 sub_metering \$ 3 unique_meter_id \$ 64 load_resource lr_in_as \$ 3 site_ownr_site_name 27 street city \$ 64 zipcode \$ 5 tspdsp_ia \$ 3 tspdsp_ia_year_tspdsp_name_tspdsp_duns \$ 64 28 noie_area wsale_meter priv_net onsite_gen \$ 3 gen_type \$ 30 load_desc \$ 256 29 id_info1 - id_info23 \$ 64; 30 31 Set perm.ers_inputs_okay_final; 32 33 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gene 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MITCHING DAY PAIR'; 41 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 43 run; </pre>	24	length eridnum 8 gsename gseduns 64 renewal opt in 3 resource type resource name baseline 564	
<pre>27 street city \$ 64 zipcode \$ 5 tspdsp_ia \$ 3 tspdsp_ia_year tspdsp_name tspdsp_duns \$ 64 28 noie_area wsale_meter priv_net onsite_gen \$ 3 gen_type \$ 30 load_desc \$ 256 29 id_infol - id_info23 \$ 64; 30 31 set perm.ers_inputs_okay_final; 32 33 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen= 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 oF 10'; 39 if bline3 = 3 then blout2 = 'DEFAULT-MICHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 43 run; ***********************************</pre>	25		
<pre>28 noie_area wsale_meter priv_net onsite_gen \$ 3 gen_type \$ 30 load_desc \$ 256 29 id_info1 - id_info23 \$ 64; 30 31 set perm.ers_inputs_okay_final; 32 33 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gene 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 43 run; </pre>	26	esiid \$ 64 sub_metering \$ 3 unique_meter_id \$ 64 load_resource lr_in_as \$ 3 site_ownr site_name	
<pre>29 id_info1 - id_info23 \$ 64; 30 31 set perm.ers_inputs_okay_final; 32 33 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gene 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; </pre>	27		
<pre>30 31 Set perm.ers_inputs_okay_final; 32 33 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen= 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; </pre>			
<pre>31 set perm.ers_inputs_okay_final; 32 33 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen= 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; Comput-(Untitled) E Log-(Untitled) Step 15 DDE_Code</pre>		id_info1 - id_info23 \$ 64;	
<pre>32 33 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen= 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; C C C C C C C C C C C C C C C C C C C</pre>			
<pre>33 renewal_opt_in = ''; 34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen= 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; Comput-(Untitled) E Log-(Untitled) & Step 15 DDE_Code</pre>		set perm.ers_inputs_okay_iinal;	
<pre>34 load = 'Load'; 35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen = 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; C C C C C C C C C C C C C C C C C C C</pre>		renewal ont in = !!.	
<pre>35 ss_gen = 'Self-Serving Gen'; 36 non_ss_gen= 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; COntput-(Untitled) Cutput-(Untitled) Step 15 DDE_Code</pre>			
<pre>36 non_ss_gen= 'Non Self-Serving Gen'; 37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; C C C C C C C C C C C C C C C C C C C</pre>			
<pre>37 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION'; 38 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10'; 39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; C C C C C C C C C C C C C</pre>		-	
39 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR'; 40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; C C C C C C C C C C C C C C C C C C C	37		
40 blout4 = 'ALTERNATE'; 41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; C C Code ► Code	38	if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10';	
41 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE'; 42 43 run; C Dutput - (Untitled) Code			
42 43 run; Contput - (Untitled) E Log - (Untitled) E Step 15 DDE_Code			
43 run; Contput - (Untitled) E Log - (Untitled) E Step 15 DDE_Code		if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE';	
< <tr> Image: Control of the second second</tr>			_
Continut - (Untitled) Code Code Code	43		<u> </u>
	<u> </u>		2
File saved successfully.	🔡 Outp	out - (Untitled) 📄 Log - (Untitled)	
	File saved	successfully. 🐙 Z:\'Emergency_Interruptible_Load\'2012 i Ln 16, Col	57

The next data step begins the process of creating tab names for the workbook to be created using the QSE provided resource name as a basis. Some clean-up is done to remove any special characters that either will not be accepted by Excel as tab names or that would create problems when SAS subsequently reads the workbook back in when the submitted by the QSE as an offer. An important statement included in this part of the code is the final modification of the tab name using the upcase function; the statement is needed because, unlike SAS, Excel will treat tab names with the same letters and different cases as the same tab. By using the upcase function, SAS can recognize in the subsequent step that a duplicate tab has been generated, and modify tab names to ensure separate tabs are written.



The next section of code splits the input data into two datasets, one containing site information to be written to an Alt tab (because the QSE submitted it on an Alt tab), and one containing resource level information for sites submitted on a standard resource tab. The code then deals with the possibility of duplicate tab names ... submission of duplicate resource names is allowed and the manipulation performed in the preceding step can also result in the duplication of tab names. The code below appends an index number to any duplicate tab names found on a QSE-by-QSE basis to eliminate the duplication ... including any duplication created in the previous step by the upcase function. The

remaining steps in this section of code determine how many QSEs have resources in the batch of workbooks being processed; a single workbook of resources is produced for each QSE in the batch. The macro variable '&numqses' is used in the macro do loop to sequence through all QSEs in the batch, and the 'qse_list' dataset contains the list of QSE names and Duns numbers included in the batch. The 'qse_erid_list' contains a list of all resource tabs that need to be created across all the workbooks in the batch.

```
🐝 SAS - [Step 15 DDE Code Write out ok erids.sas]
                                                                                           File Edit View Tools Run Solutions Window Help
                                                                                           - 8 ×
                            ~
                               || 🗅 🚅 📕 🎒 🐧 🐰 🖻 🛍 🗠 🎁 🖳 🖈 🗙 🛈 🥔
 \checkmark
  85<mark>5 data</mark> okay ers rtab inputs okay ers alt inputs ;
                                                                                               ^
  86
  87
          drop count;
  88
  89
          set okay ers inputs 1;
  90
          by qseduns tab_name eridnum;
  91
          if tab name '= 'Alt' then do;
  92
  93
              if first.tab name then count = 0;
  94
              if first.eridnum then count + 1;
  95
             if count > 1 & count < 100 then
  96
                  tab_name = translate(compbl(substr(tab_name,1,26) || put(count,2.)),'_','');
  97
             output okay_ers_rtab_inputs;
  98
          end:
  99
          else output okay_ers_alt_inputs;
  100
 101 run;
 102
 103 proc sort data = okay ers rtab inputs out = qse erid list nodupkey;
         by qseduns tab name eridnum;
 104
 105 run;
 106
 107 proc freq data = okay ers inputs 1 noprint;
 108
         tables qsename * qseduns / out = qse list;
 109 run;
 110
          111
         /* create macro variable of number of qses */
 112
          113
 114
 115 %let numqses = 0;
 116
 117⊡ data null ;
 118
         set qse list nobs = numqses;
 119
          call symputx("numqses", put(numqses, 5.));
 120
          stop:
 121 run;
  100
                                                                                             >
📳 Output - (Untitled)
                    🗒 Log - (Untitled)
                                        🚰 Step 15 DDE Code...
Autosave complete
                                                              🖵 Z:VEmergency_Interruptible_Lo Ln 52, Col 5
```

The remainder of the code is contained in the 'run_qse' macro, which cycles through all QSEs in the batch and produces a workbook for each QSE. The 'fill' dataset is just a placeholder dataset to be used later in creating tabs for each of the resources being output. The 'qse_erid_list_1' dataset is a subset of the entire list of QSE's tabs specific to the QSE being processed during the iteration of the macro do loop. The last part of this section of code determines which Excel

template should be used for the QSE being processed. If an 'Alt' tab is to be created, the 'ID_Alt_Template.xlsx' template, is chosen; otherwise, the 'ID_Template' is chosen. The macro variable, '&id_template', is set up to pass the template selection on to the subsequent code.

```
SAS - [Step 15 DDE Code Write out ok erids.sas]
File Edit ⊻iew Tools Run Solutions Window Help
                                                                                                   - 8
                                                                                                       ×
                              🔽 🖹 🖆 🖪 🎒 🛕 🐰 🖻 🛍 🗠 🎁 🚉 🖈 🗙 🛈 🥔
  \checkmark
  123 ⊡ %macro run qse;
                                                                                                       ^
  124
  125
           data fill;
              a = 1; b = 1; c = 1;
  126
  127
               if a > 1;
  128
          run;
  129
  130
               /* loop through QSEs */
  131
           %if &numqses. > 0 %then %do i = 1 %to &numqses.;
  132
  133
               data _null_;
  134
                   obsnum = input("&i",4.);
                   set qse_list point = obsnum;
  135
                   call symputx("qseduns", put(qseduns, $64.));
  136
  137
                   call symputx("qseduns_fname",put(input(qseduns,13.),z16.));
  138
                   stop:
  139
               run;
  140
  141
                   /* set macro numerids to the number of r tabs for qse being processed */
  142
               %let numerids = 0;
  143
  144
  145
               /* create a file of eridnums for this qse */
  146
               data qse_erid_list_1 ;
                   set qse_erid_list (where = (qseduns = "&qseduns."));
  147
                   numerids + 1;
  148
                   call symputx("numerids", numerids);
  149
  150
               run;
  151
  152
               /* set macro variable for the template to be used (with or without and Alt tab) */
  153
               %let ID_Template = ID_Template;
  154
  155
  156
               data null;
  157
  158
                   set okay ers alt inputs (where = (qseduns = "&qseduns."));
  159
                   call symputx ("ID_Template", "ID_Alt_Template");
  160
                   stop;
  161
               run;
                                                                                                    >
                      🖺 Log - (Untitled)
                                           🛃 Step 15 DDE_Code...
 🔡 Output - (Untitled)
                                                                    Interruptible_Lo: Ln 1, Col 1
Autosave complete
```

The next data step issues system commands to create a copy of the appropriate template; this is done to prevent the original template from getting corrupted during the processing. The following macro do loop uses proc export to create tabs in the copied template workbook using the names in 'qse_erid_list_1' that was generated in the earlier step. The placeholder dataset, 'fill', is exported to each of the tabs created; this is done, because proc export does not run unless

it has a dataset to export. In earlier versions of this code, the tabs were created using DDE commands, and we've found that using proc export is a better alternative in terms of both speed and reliability.

	Step 15 DDE_Code_Write out ok erids.sas]
£ Eile Edit	: <u>View T</u> ools <u>R</u> un <u>Solutions Window H</u> elp
· ·	Image:
165	data null;
166	length command \$ 200;
167	command="&local_drive.";
168	call system(command);
169	command="cd\stemplatepath.\";
170	call system(command);
171	command="copy &ID_Templatexlsx &ID_Templatecopy.xlsx";
172	call system(command);
173	run;
174	
175	/* cycle through all tab_names for the qse being processed and proc export the tab to the */
176	/* template copy workbook note: fill is an empty dataset to export */
177	
178	<pre>%if &numerids. > 0 %then %do j = 1 %to &numerids.</pre>
179	
180	
181	/* create tabs for all the R tab_names needed for this qse */
182	data _null_;
183	obsnum = input("&j",4.);
184	set qse_erid_list_1 point = obsnum;
185	call symputx("tab_name", tab_name);
186	stop;
187	run;
188	
189	proc export data = fill outfile =
190	"&local_drive.\&templatepath.\&ID_Templatecopy.xlsx";
191	<pre>sheet="&tab_name.";</pre>
192	run;
193	
194	%end;
<u><</u>	
🔡 Output	- (Untitled) El Log - (Untitled) El Step 15 DDE_Code
Autosave com	nplete 🔤 Z: VErnergency_Interruptible_Load Ln 167, Col 37

The next set of code, which is not shown below, formats three summary reports for the QSE being processed and exports them with proc export to the workbook being created for the QSE. After those exports are complete, the code below runs; the first line uses an X command to launch Excel, and must be coded with the path to the actual excel.exe on the pc running the code. The first data step causes SAS to pause and give Excel time to open and the filename statement allows SAS to send DDE commands to Excel.

The next three data steps send DDE commands to Excel to open the workbooks needed to send data from SAS to Excel. The open command is constructed in the required format using a macro variable for the path. Double quotes are used inside the 'open' command to allow the macro variables to be resolved, the single quotes are needed by the syntax of the DDE command, and the %unquote unmasks the single quotes created by the %str function to turn the single quotes into values the SAS compiler recognizes. The sleep commands are all used to give time for the Excel command to complete before the next command is issued by SAS. The first workbook opened is named 'input data.xlsx'; SAS will later write data to this workbook that is needed to populate tabs in the QSE workbook. The second workbook opened is named 'rtab_template.xlsx'; this workbook is the template used for creating resource tabs, and it contains links to 'input data.xlsx'. The third workbook is named either 'id_template_copy.xlsx' or 'id_alt_template_copy.xlsx' depending on whether the QSE workbook being created needs and 'Alt' tab or not. This workbook contains a tab named 'Identification' and is the template for that tab; the tab has links to 'input data.xlsx'. If the' id_alt_template_copy.xlsx' is used, it has a tab named 'Alt' and is the template for that tab; the tab has links to 'input data.xlsx'.

😽 SAS	- [Step 15 DDE_Code_Write out ok erids.sas]
诺 Eile	Edit View Iools Run Solutions Window Help
	🔄 🗈 🚔 🖬 🖨 🔃 🖇 🖻 🗥 🕫 🕲 🐼
415	x "'C:\Program Files\Microsoft Office\OFFICE12\excel.exc'";
416	-
417	data null;
418	rc = sleep(2);
419	stop;
420	run;
421	
422	filename ddecmds dde 'excel system';
423	
424	/* open the input data.xlsx workbook */
425	
426	data _null_;
427	rc = sleep(1);
428	file ddecmds;
429	put %unquote(%str(%'[open("&local_drive.\&templatepath.\input data.xlsx")]%'));
430	stop;
431	run;
432	
433	/* open the original template and template copy */
434	/* the original template has links to all the input parameters on the input data.xlsx workbook */
435	
436	data _null_;
437	<pre>rc = SLEEP(1);</pre>
438	file ddecmds;
439	<pre>put %unquote(%str(%'[open("&local_drive.\&templatepath.\Rtab_Template.xlsx")]%'));</pre>
440	run;
441 442	
442	data _null_; rc = SLEEP(1);
444	file ddecmds;
445	<pre>put %unquote(%str(%'[open("&local drive.\&templatepath.\&ID Template. copy.xlsx")]%'));</pre>
446	<pre>run;</pre>
<	
Dut 🔝	tput-(Untitled)
File saved	i successfully. 💷 D: test Ln 433, Col 1

A portion of the input data.xlsx workbook is shown below; the workbook has three tabs: 'id_data', 'load_data' and 'site_data' that were set up for the three different types of data to be written to the workbook. SAS will export data needed for the' Identification' tab to the 'id_data' tab; resource level data to the 'load_data' tab; and site (customer) level data to the 'site_data' tab. The workbook is set up with a named range, 'id_inputs', which refers to cells A1 – AA2; this named range is used by SAS as the output location, when it writes to the 'id_data' tab.

Note that only one row is used for data going to the 'Identification' tab. Similarly, the workbook is set up with a named range, 'Ioad_inputs', which refers to cells A1 – P2; this named range is used by SAS as the output location when it writes resource-level to the 'Ioad_data' tab; only one row is used for resource-level data. Finally, the workbook has a named

range, 'site_inputs', which refers to cells A1 – T1000, that is used by SAS as the output location, when it writes site-level data to the 'site_data' tab.

		input data	.xlsx - Micr	osoft Excel				_	= x
Home Insert Page Lay	out Formulas	Data Revie	ew View	Developer	Add-Ins			0 -	⇒ x
HS Sans Serif ▼ 10 ▼ Paste ✓ B I U ▼ □ ▼ 3			• % •	Conditiona	able *	¦ ⊷ Insert ▼ } ▲ Delete ▼ ↓ Format ▼		t & Find & er + Select +	
Clipboard 🖻 🛛 Font	🗟 🛛 Alignme	nt 🖻 Nu	umber 🛛 🖻	Sty	les	Cells	Ed	iting	
A1 •	<i>f</i> ∡ qsename								×
A B C	D	E	F	G	Н		J	K	
1 gsename gseduns id_info1	id_info2	id_info3	id_info4	id_info5	id_info6	id_info7	id_info8	id_info9	id_
2 test 123456789 submit co 3	Procurement ELID	Joe Blow Sr	signature	head cheese	999-555-4321	999-555-5432	email1	Joe Blow Jr	vici
H • • • id_data / load_data / s	ite_data 🏑 💱 🦯 👘			I 4 📃					
Ready 🔚						III III 1009	« 🖃 —		.:

8	9 9	- (°I - [• 📄			i	nput data.xlsx -	Microsoft Excel				-	. = x
	Home	Inser	t Page Layou	t Formulas	Data Revie	w View	Developer	Add-Ins				۲	- = X
Paste V		IS Sans Se B I <u>I</u>	rrif v 10 v ,			Wrap Text		▼ • • • • • • • • • • • • • • • • • • •	Conditional Form Formatting + as Tak Styles		sert Delete Fo		& Find & • Select •
	A1			eridnum									×
	A		В	С	D	E	F	G	Н	I	J K	L	-
1 ei	ridnum		renewal_opt_in	resource_type	resource_nam	e baseline	analyze_offer1	analyze_offer2	analyze_offer3	analyze_offer4	blout1 blout		blo
2			yes	load	load 123	alternate	Yes	No	Yes	No		Default - Matching Day	pair Alt
3													
4													
5													-
	→ id_d	lata 📜 lo	oad_data / site	_data 🦯 🖏 🦯									
Ready	2											IO0% ⊖ ↓	

•	. 9	- @ - D	=) =				input	data.xlsx - Mi	crosoft Ex	cel						_	= x
	Home	Insert	Page Layout	Formula	as Data	Review	View D	eveloper A	dd-Ins							🥝 –	■ X
Paste		AS Sans Serif ₿ <u>I</u> <u>U</u> ¬	• 10 • 4		= <mark>=</mark> > = = #		/rap Text Ierge & Center →	Text \$ ~ %	• • • • • • • • • • • • • • • • • • •	Condition Formatting	al Format (g * as Table * Sty	ieli Inse		Format	Σ AutoSum Fill ▼ 2 Clear ▼	Sort & I Filter * S	
Clipboa	a 🖻		Font	G.	1	Alignment	5	Numb	er 🕞	i J	Styles		Cells			Editing	
	A1	•	(• f _x	esiid													≯
		A		В		С	D	E		F	G	Н	1	J	K	L	
	siid				ring unique	e_meter_id	load_resourc	e Ir_in_as	s	ite_ownr	site_name	street	city	zipcode	tspdsp_i	tspdsp_ie	_ye f
2 12	23456789	3012345678	90123456789	No			yes	yes	lo	bad 123 inc	load 123 plan	t 1 first ave	anywhere	э	yes	2002	
3																	
4																	
5																	-
14.4.1	→I id_(data 📈 load	_datasite	data 🧷 🔁	7		,								1	1	> i
Ready	2													1009	• 🗩	-0	• .::

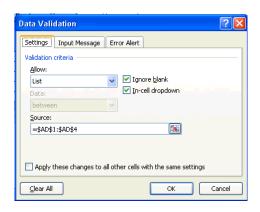
Name Manager			? 🗙
<u>N</u> ew	Edit, Delete		Eilter 🕶
Name	Value	Refers To	Scope
id_inputs ioad_inputs isite_inputs	{"qsename","qsedu {"eridnum","renewa {"esiid","sub_meteri	=id_data1\$A\$1:\$AA\$2 =load_data!\$A\$1:\$P\$2 =site_data!\$A\$1:\$T\$1000	Workbo Workbo Workbo
<	1111		>
Refers to:			

A portion of the 'rtab_template.xlsx' workbook is shown; the selected cell, A7, has a link to cell A2 on the 'site_data' tab in the 'input data.xlsx' workbook. The link is set up as an IF statement so that if the cell being linked to is populated, the value of the A2 cell is used, otherwise the cell is set to null. All cells in the 'rtab_template' in rows 7 and beyond are similar links to the appropriate cells on the 'site_data' tab of input data.xlsx.

) 🖬 🕫 - 🗎 🗐 =		F	tab_template.xlsx - Mi	crosoft Excel				- = ×
	Home Insert Page Layout	Formulas Data	Review View	Developer Add	-Ins			() _ = X
Pas	Arial 10 · A B Z U · D · A	<u>A</u> ·■≡≡\$	Wrap Text		Condition Formatting		Delete Format	So	ort & Find & ter * Select *
	A7 • (* <i>f</i> _x	~~~~~		input data.xlsx]site_d	ata'!A2,"")				*
	А	В	С	D	E	F	G	Н	
1	ERID Number	Renewal Opt In	Resource Type	ERS Time Period	Include Time Period?	Declared Self-Serve Capacity	Declared Injection Capacity	Capacity Offer (MW)	
2		yes	load	Business Hours 1	Yes				
3	Resource Name		Baseline	Business Hours 2	No				
4	load 123		alternate	Business Hours 3	Yes				
5				Non Business Hours	No				
6	ESI-ID	Sub Metering	Unique Meter ID	LR at this Site	LR Participating in AS Market	Site Owner Controlling Entity	Site Name	Site Address (street)	
	12345678901234567890123456789	①		yes	yes	load 123 inc	load 123 plant	1 first ave	anywhere
8									
9							_		
10								+	
12									
13									
14									_
14 4	🕨 🗏 R template 🤇 😓						14		▶
Read	iy 🛅						III II 100%	<u> </u>	

Cells on the template that need to be populated by SAS in rows 1 - 5 are similar links but connect to the 'load_data' tab. The baseline cell, C4, can be directly populated, but also has an associated drop-down list in cells AD1 – AD4, which also are links to the 'input data.xlsx' workbook. Using this technique, SAS can control the items showing up in the drop down list; for this application it is the way ERCOT uses to control baseline selection options that are available to the QSE for each resource.

6) 🖬 🔊 - (° - 🗋 🗐 🗧		R	Rtab_template.xlsx - Mi	crosoft Excel				- = X
	Home Insert Page Layout	Formulas Data	Review View	v Developer Add	l-Ins) – 🖻 X
	Arial • 10 • A ste • • • • • • • • • • • • • • • • • • •	▲・■■	Wrap Text		Condition Formatti	onal Format Cell Insert De ng * as Table * Styles *	elete Format	So	rt & Find & ter * Select *
	C4 • fx =IF('[input data.xlsx]load_data'!E2>0,'[input data.xlsx]load_data'!E2,"")								
	A	В	С	D	E	F	G	Н	
1	ERID Number	Renewal Opt In	Resource Type	ERS Time Period	Include Time Period?	Declared Self-Serve Capacity	Declared Injection Capacity	Capacity Offer (MW)	Î
2		yes	load	Business Hours 1	Yes				
3	Resource Name		(Business Hours 2	No				
4	load 123	(1)	alternate	Isiness Hours 3	Yes				
5									
				n Business Hours	No			01	
6	ESI-ID		Default - Matching Day Alternate			Site Owner Controlling Entity	Site Name	Site Address (street)	
7				LR at	No LR Participating		Site Name	Address	anywhere
7 8		Sub Metering		LR at this Site	No LR Participating in AS Market	Controlling Entity		Address (street)	anywhere
7 8 9		Sub Metering		LR at this Site	No LR Participating in AS Market	Controlling Entity		Address (street)	anywhere
7 8	12345678901234567890123456789	Sub Metering		LR at this Site	No LR Participating in AS Market	Controlling Entity	load 123 plant	Address (street) 1 first ave	anywhere
7 8 9 10	12345678901234567890123456789	Sub Metering		LR at this Site	No LR Participating in AS Market	Controlling Entity load 123 inc		Address (street) 1 first ave	anywhere



A portion of the 'id_alt_template.xlsx workbook' is shown; the selected cell, C23, has a link to cell A2 on the 'id_data' tab in the 'input data.xlsx' workbook. As with the 'rtab_template', the link is set up as an IF statement so that if the cell being linked to is populated, the value of the A2 cell is used, otherwise the cell is set to null. All cells containing QSE provided data in column C between rows 23 - 51 are links to the appropriate cells on the 'id_data' tab of input data.xlsx. The submittal type cell, C26, can be directly populated, but also has an associated drop-down list in cells AC1 – AC10, which also are links to the input data.xlsx workbook. Using this technique, SAS can control the items showing up in the drop down list; for this application it is the way ERCOT uses to control the allowed submission type to be used by the QSE.

Image: Connections Image: Connec	2	📕 🔊 - (° - 🗋 🗐 = 🛛 II	D_Alt_Template.xls	- Microsoft Excel 💷 🗖	х					
Get External Ali Properties Data 21 21 21 21 21 21 21 21 21 21 21 21 21 2	2	Home Insert Page Layout Formulas	Data Review	v View Developer Add-Ins 🔞 – 🗖	x					
A B C 22 Qualified Scheduling Entity (QSE) Information 23 QSE Name Itest 24 QSE DUNs Number 123456789 25 Company submitting form (if different than QSE) submit co 26 Submittal Type Procurement ELID 27 QSE Authorized Representative Procurement ELID 28 Printed Name: Joe Blow Sr 29 Signature: signature 30 Title: head cheese 31 Phone Number: 999-565-4321 32 Cell Phone Number: 999-565-5432 33 E-mail Address: email1 34 QSE Primary Contact 1 Joe Blow Jr	Exte	ernal Refresh All + @ Edit Links	& Reapply Advanced	Text to Remove What-If Analysis *						
22Qualified Scheduling Entity (QSE) Information23QSE Nametest24QSE DUNS Number12345678925Company submitting form (if different than QSE)submit co26Submittal TypeProcurement ELID27QSE Authorized Representative28Printed Name:Joe Blow Sr29Signature:signature30Title:head cheese31Phone Number:999-555-432132Cell Phone Number:999-555-543233E-mail Address:email134QSE Primary Contact 135Printed Name:Joe Blow Jr	C23 🔹 🖍 =IF('[input data.xlsx]id_data'!A2>0,'[input data.xlsx]id_data'!A2,'''')									
23 QSE Name	C B									
24QSE DUNs Number12345678925Company submitting form (if different than QSE)submit co26Submittal TypeProcurement ELID27QSE Authorized Representative28Printed Name:Joe Blow Sr29Signature:signature30Title:head cheese31Phone Number:999-555-432132Cell Phone Number:999-555-543233E-mail Address:email134QSE Primary Contact 135Printed Name:Joe Blow Jr		Qualified Scheduling Entity (QSE) Information								
24 Stor Donot National 25 Company submitting form (if different than QSE) submit co 26 Submittal Type Procurement ELID 27 QSE Authorized Representative 28 Printed Name: Joe Blow Sr 29 Signature: signature 30 Title: head cheese 31 Phone Number: 999-555-4321 32 Cell Phone Number: 999-555-5432 33 E-mail Address: email1 34 QSE Primary Contact 1 Joe Blow Jr 35 Printed Name: Joe Blow Jr		QSE Name	٩	test						
26 Submittal Type Procurement ELID 27 QSE Authorized Representative 28 Printed Name: Joe Blow Sr 29 Signature: signature 30 Title: head cheese 31 Phone Number: 999-555-4321 32 Cell Phone Number: 999-555-5432 33 E-mail Address: email1 34 QSE Primary Contact 1 35 Printed Name: Joe Blow Jr	Ī	QSE DUNs Number		123456789						
27 QSE Authorized Representative 28 Printed Name: 29 Signature: 30 Title: 31 Phone Number: 32 Cell Phone Number: 33 E-mail Address: 34 QSE Primary Contact 1 35 Printed Name:	Ī	Company submitting form (if different than QSE)		submit co						
28 Printed Name: Joe Blow Sr 29 Signature: signature 30 Title: head cheese 31 Phone Number: 999-555-4321 32 Cell Phone Number: 999-555-5432 33 E-mail Address: email1 34 QSE Primary Contact 1 35 Printed Name: Joe Blow Jr	:	Submittal Type		Procurement ELID						
28Printed Name:Joe Blow Sr29Signature:signature30Title:head cheese31Phone Number:999-555-432132Cell Phone Number:999-555-543233E-mail Address:email134QSE Primary Contact 135Printed Name:Joe Blow Jr		QSE Authorized Representative								
30 Title: head cheese 31 Phone Number: 999-555-4321 32 Cell Phone Number: 999-555-5432 33 E-mail Address: email1 34 QSE Primary Contact 1 35 Printed Name: Joe Blow Jr	Ī	Printed Name:		Joe Blow Sr						
31 Phone Number: 999-555-4321 32 Cell Phone Number: 999-555-5432 33 E-mail Address: email1 34 QSE Primary Contact 1 35 Printed Name: Joe Blow Jr		Signature:		signature	≡					
32 Cell Phone Number: 999-555-5432 33 E-mail Address: email1 34 QSE Primary Contact 1 35 Printed Name: Joe Blow Jr	L L			head cheese						
33 E-mail Address: email1 34 QSE Primary Contact 1 35 Printed Name: Joe Blow Jr		Phone Number:								
34 QSE Primary Contact 1 35 Printed Name: Joe Blow Jr	L F									
35 Printed Name: Joe Blow Jr				[email1						
	L H									
36 Title: [vice head cheese	L H									
		Title:			+					
Identification R Alt Identification Identification Ready Identification Identification Identification					I					

Using the three workbooks with appropriate links and opened on the PC running SAS, the code is now ready to move data from SAS datasets out to Excel. The libname, shown below, sets up a connection to the 'id_inputs' named range on the 'input_data.xlsx' workbook; this is the output destination used by SAS for the data. The proc datasets deletes all rows from the 'id_inputs' range ... SAS requires that the range be blank if it is being used for output. The data step outputs the required data to the named range and, since the 'id_alt_template_copy.xlsx' workbook is open and has links on its 'Identification' tab to the cells in the 'id_inputs' range, the values of those links are updated by this data step.

It should be noted that when the Excel libname engine is used to write data out to an Excel workbook, it automatically includes a title row first and then writes variable values. To work around this limitation, this code used the 'input_data.xlsx' workbook as the output destination rather than outputting directly to the final workbook.

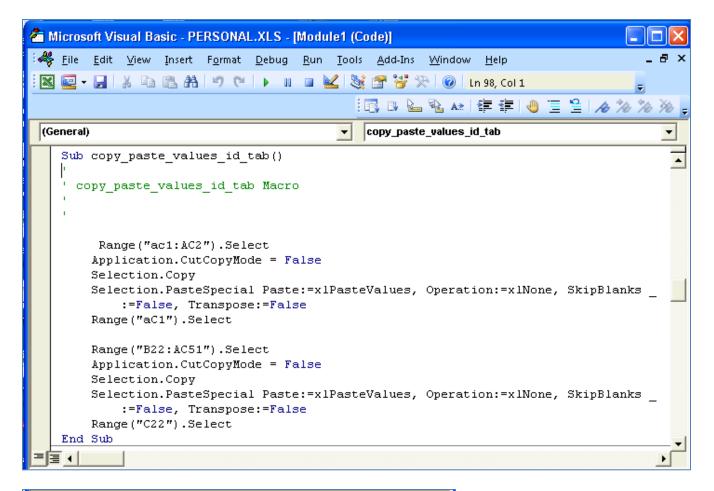
```
🐝 SAS - [Step 15 DDE Code Write out ok erids.sas]
🖹 Eile Edit View Tools Run Solutions Window Help
                                  🖹 🗋 🚔 🛃 🎒 🐧 🐰 🖻 📾 🗠  🗿 🔍 🖈 🗙 🛈 🥔
 ~
                               v
 454
               libname mylib "&local drive.\&templatepath.\input data.xlsx";
                                                                                               ~
 455
               proc datasets lib = mylib noprint;
 456
 457
                    delete id inputs;
 458
               quit;
 459
 460
               data mylib.id inputs;
 461
 462
                   keep qsename qseduns id info1 - id info23 sub type1 sub type2;
 463
 464
                   set okay ers inputs 1 (where = (qseduns = "&qseduns."));
 465
                    id info2 = '';
                   sub type1 = '10m Procurement Offer';
 466
                   sub type2 = '30m Procurement Offer';
 467
 468
                   output;
 469
                   stop;
 470
 471
               run:
🔡 Output - (Untitled)
                      🗒 Log - (Untitled)
                                            🛃 Step 15 DDE Code...
🗉 D: 'test
                                                                                  Ln 433, Col 1
```

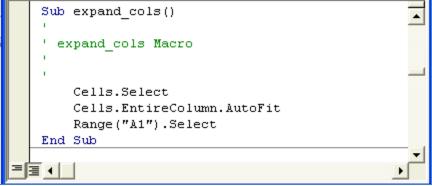
The next data step, shown below, issues DDE commands to activate various tabs in the 'id_alt_template_copy.xlsx' workbook and to run Excel macros that are stored in 'personal.xls'. Initially, SAS activates the 'Identification' tab and runs a macro called 'copy_paste_values_id_tab', which is shown below. The macro does a 'copy – paste special values' on the AC1 – AC2 range and thus removes the links in the drop down list for the submission type. Then the macro does a 'copy – paste special values' on the B22 – C51 range and thus removes the remaining links on the 'Identification' tab.

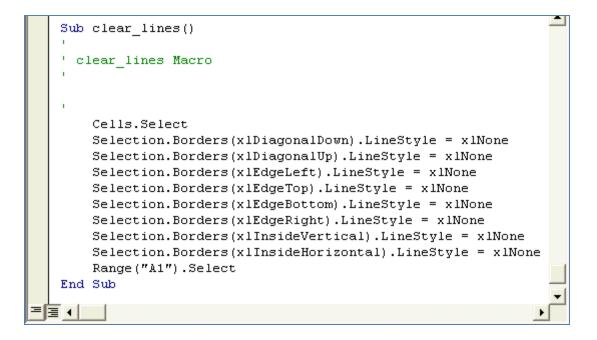
SAS then successively activates the 'Exceptions' tab, the 'Availability_Results' tab and the 'Baseline_Info' tab on the 'id_alt_template_copy.xlsx' workbook; and then runs two additional Excel macros, 'expand_cols' and 'clear_lines', on each tab. These three tabs contain summary level information on the resources submitted by the QSE on the workbook and were written by proc export as described above. The macros, shown below, 'auto-fit' all the columns on the activated tab, and clear all cell outlines on that tab. The first of these macros eliminates the annoying task of expanding columns in a workbook exported by SAS, and the second macro eliminates mysterious cell outlines that appear somewhat randomly (as far as I can tell) on these tabs.

I should point out that I am not a VBA programmer, and that all the macros described in this paper were produced by recording keystrokes that accomplished the desired tasks. In some case I have done minor editing primarily to clean up and simplify the macros.

😽 SAS - [9	Step 15 DDE_Code_Write out ok erids.sas]	X
🖹 Eile Edit	t <u>V</u> iew <u>T</u> ools <u>R</u> un <u>Solutions</u> <u>Window</u> <u>H</u> elpt	3 ×
· · ·	🖌 💽 🕲 🗠 🗋 🖾 🗠 🕲 😧 🖈	×
478	data _null;	~
479	file ddecmds;	
480	<pre>put '[workbook.activate("Identification")]';</pre>	
481	<pre>put '[run("PERSONAL.XLS!copy_paste_values_id_tab")]';</pre>	
482	<pre>put '[workbook.activate("Exceptions")]';</pre>	
483	<pre>put '[run("PERSONAL.XLS!expand_cols")]';</pre>	
484	<pre>put '[run("PERSONAL.XLS!clear_lines")]';</pre>	
485	<pre>put '[workbook.activate("Availabilty_Results")]';</pre>	
486	<pre>put '[run("PERSONAL.XLS!expand_cols")]';</pre>	
487	<pre>put '[run("PERSONAL.XLS!clear_lines")]';</pre>	_
488	<pre>put '[workbook.activate("Baseline_Info")]';</pre>	
489	<pre>put '[run("PERSONAL.XLS!expand_cols")]';</pre>	
490	<pre>put '[run("PERSONAL.XLS!clear_lines")]';</pre>	
491	stop;	
492	run;	~
<		>
🔡 Output	- (Untitled)	
🗐 D:'test	Ln 433, Col 1	







The next two sections of code deal, in a similar way, with outputting resource level and site level data; the first section outputs to the 'Alt' tab and the second outputs to standard resource tabs. Since the code is similar, the description below will focus on the standard resource tab.

The code to write out standard resource tabs is, again, embedded in a macro do loop to cycle through all the tabs that needed to be written for the QSE. The first data step creates macro variables to specify the tab being output; as above, the proc datasets clears the Excel range, 'load_inputs', prior to the output, and the second data step outputs resource level to the range. The steps are repeated for the site level data; an important difference for site data is the output will consist of one or more rows based on the number of sites associated with the resource.

<pre> the full version of the second ver</pre>	🕏 SAS	- [Step 15 DDE_Code_Write out ok erids.sas *]	
<pre>\$\$ %if funmerids. > 0 %then %do j = 1 %to funmerids; \$\$ data _null; \$\$ obmum = input("si",4.); \$\$ est esse rid_list_loott = obsnum; \$\$ call symputx("rubnum", put(eridnum,20.)); \$\$ call symputx("rubnum", put(eridnum,20.)); \$\$ atop; \$\$ proc datasets lib = mylb noprint; \$\$ delete load_inputs; \$\$ data mylb.load_inputs; \$\$ data mylb.load_inputs; \$\$ analyze_offer1 = analyze_offer4 blout1 = blout4 load ss_gen non_ss_gen; \$\$ atop; \$\$ output; \$\$ atop; \$\$ atop; \$\$ output; \$\$ atop; \$\$ output; \$\$ atop; \$\$ output; \$\$ atop; \$\$ output; \$\$ delete site_inputs; \$\$ delete inputs; \$\$ delete site_inputs; \$\$ delete site_inputs; \$\$ delete site_inputs; \$\$ delete site_inputs; \$\$ delete site_inputs; \$\$ delete inputs; \$\$ delete site_inputs; \$\$ delete site_inputs; \$\$ delete site_inputs; \$\$ delete site_inputs; \$\$ delete inputs; \$\$ delete site_inputs; \$\$ delete inputs; \$\$ delete input</pre>	👬 Eile	Edit <u>V</u> iew <u>I</u> ools <u>R</u> un <u>S</u> olutions <u>W</u> indow <u>H</u> elp	_ 8 >
<pre>def data_null; obsnum = input("s)",4.); set qeg_erid_list_l point = obsnum; call symputx("tab_name", tab_name); stop; call symputx("tab_name", tab_name); stop; run; run; detet load_inputs; ford datasets lib = mylib noprint; detet load_inputs; run; detet load_inputs; run; set qeg_erid_list_l (where = (eridnum = feridnum.)); output; stop; run; proc datasets lib = mylib noprint; detet site_inputs; run; proc datasets lib = mylib noprint; detet site_inputs; ford datasets lib = mylib noprint; detet site_inputs; set okay_ers_tab_inputs (where = (eridnum = feridnum.)); nobs + 1; call symputx("nobs", nobs); run; run;</pre>	~	💌 🗦 🗅 🚅 🖬 🚭 🐧 ½ 🗈 🛍 🕫 🕫 🚺 🛃 🛪 X 🛈 🛷	
<pre>665 data_null; 666 obmum = input("(s)",4.); 570 set qse_erid_list_1 point = obmum; 589 call symputx("eridnum,put(eridnum,20.)); 570 stop; 571 run; 573 proc datasets lb = mylib noprint; 574 delete load_inputs; 575 quit; 576 data mylib.load_inputs; 577 data mylib.load_inputs; 578 set qse_erid_list_1 (where = (veridnum = beridnum.)); 581 set qse_erid_list_1 (where = (veridnum = beridnum.)); 582 stop; 583 run; 584 suppl; 585 stop; 585 stop; 586 stop; 586 stop; 586 stop; 586 stop; 586 stop; 587 run; 589 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 591 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_dnum sois_area wsale_meter 592 priv_net omsite_gen gen_type load_desc; 593 sto kay_ers_ttab_inputs (where = (eridnum = feridnum.)); 594 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_dnum sois_area wsale_meter 595 priv_net omsite_gen gen_type load_desc; 595 stokay_ers_ttab_inputs (where = (eridnum = feridnum.)); 596 nobs + 1; 597 cuns</pre>	563	if snumerids. > 0 ithen ido j = 1 ito snumerids;	^
<pre>dotamm = input("si",4.); set gse_erid_list_1 point = obsnum; call symputx("ctahum",put(eridnum,20.)); call symputx("tab_name", tab_name); stop; run; run; delete load_inputs; gquit; delete load_inputs; delete load_inputs; data mylib.load_inputs; keep eridnum renewal_opt_in resource_type resource_name baseline analyte_offer1 - analyte_offer4 blout1 - blout4 load ss_gen non_ss_gen; set gse_erid_list_1 (where = (eridnum = &eridnum.)); delete site_inputs; gen; delete site_inputs; delete site_inputs; delete site_inputs; set gse_erid_list_1 (where = (eridnum = &eridnum.)); delete site_inputs; set gse_erid_list_inputs; set gse_erid_list_inputs; set gse_erid_list_inputs; set gse_erid_list_inputs; set gse_erid_list_inputs; set gse_erid_list_inputs; delete site_inputs; set gse_erid_inputs; set gse_ind_inputs (where = (eridnum = &eridnum.)); onbs + 1; coll symputx("nobs", nobs); coll run;</pre>	564		
<pre>set gse_eria_lis_1 point = obsuum; see call symputx("eridnum,put(eridnum,20.)); stop; stop; stop; stop; guit; see call symputx("tab_name", tab_name); set gse_erid_list_1 (where = (eridnum = seridnum.)); data mylib.load_inputs; keep eridnum renewal_opt_in resource_type resource_name baseline analyze_offer1 = analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; analyze_offer1 = analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; set gse_erid_list_1 (where = (eridnum = seridnum.)); delter site_inputs; est gse_erid_list_1 (where = (eridnum = seridnum.)); delter site_inputs; delter site_inputs; set gse_erid_site_inputs; set gse_erid_list_inputs; set gse_erid_list_inputs (where = (eridnum = seridnum.)); nobs + 1; call symputx("nobs", nobs); set of run;</pre>	565	data _null_;	
<pre>sed call symputx["eridnum",put(eridnum,20.)); sed call symputx("tab_name", tab_name); sed call symputx("tab_name second symputx(symputx(symputx symputx</pre>		obsnum = input("sj",4.);	
<pre>569 call symputx("tab_name", tab_name); 570 stop; 571 run; 573 proc datasets lib = mylib noprint; 574 delete load_inputs; 575 quit; 576 577 data mylib.load_inputs; 578 579 keep eridnum renewal_opt_in resource_type resource_name baseline 580 analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; 581 582 set qse_erid_list_1 (where = (eridnum = &eridnum.)); 584 output; 585 stop; 586 587 run; 590 delete site_inputs; 591 quit; 592 data mylib.site_inputs; 591 quit; 592 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 594 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 595 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 500 call symputx("nobs", nobs); 600 run; 501 502 503 503 503 504 505 505 505 505 505 505 505 505 505</pre>			
<pre>stop; stop; stop; stop; stop; stop; stop; stop; stop; stop; stop; stop; stop; delet load_inputs; stop; delet stop eridnum renewal_opt_in resource_type resource_name baseline analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; stop; s</pre>			
<pre>571 run; 572 573 proc datasets lib = mylib noprint; 574 delete load_inputs; 575 quit; 577 data mylib.load_inputs; 578 keep eridhum renewal_opt_in resource_type resource_name baseline 580 analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; 581 set qse_erid_list_1 (where = (eridhum = &eridhum.)); 584 output; 585 stop; 586 proc datasets lib = mylib noprint; 589 proc datasets lib = mylib noprint; 580 delete site_inputs; 591 quit; 592 stop; 593 data mylib.site_inputs; 594 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 595 city sipcode tspdsp_ia_tspdsp_name tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 set okay_ers_rtab_inputs (where = (eridhum.)); 600 nobs + 1; 601 call symputx("hobs", nobs); 603 run;</pre>			
<pre>572 573 proc datasets lib = mylib noprint; 574 delete load_inputs; 575 quit; 576 577 data mylib.load_inputs; 578 579 keep eridnum renewal_opt_in resource_type resource_name baseline 580 analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; 581 582 set qse_erid_list_1 (where = (eridnum = &eridnum.)); 583 output; 583 stop; 586 587 run; 589 proc datasets lib = mylib noprint; 589 delete site_inputs; 591 quit; 593 data mylib.site_inputs; 594 594 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 595 city sipcode tspdsp_ia tspdsp_name tspdsp_name tspdsp_duns noie_area wsale_meter 596 priv_met onsite_gen gen_type load_desc; 597 priv_met onsite_gen gen_type load_desc; 598 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 603 run;</pre>			
<pre>\$73 proc datasets lib = mylib noprint; 574 delete load_inputs; 575 quit; 577 data mylib.load_inputs; 579 keep eridnum renewal_opt_in resource_type resource_name baseline 580 analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; 581 set qse_erid_list_1 (where = (eridnum = &eridnum.)); 583 output; 584 output; 585 stop; 586 grun; 589 proc datasets lib = mylib noprint; 580 delete site_inputs; 591 quit; 592 data mylib.site_inputs; 593 data mylib.site_inputs; 594 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 595 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 596 priv_net onsite_gen_type load_desc; 597 priv_net onsite_site_inputs (where = (eridnum = &eridnum.)); 598 ast okay_ers_tab_inputs (where = (eridnum = &eridnum.)); 599 ast okay_ers_tab_inputs (where = (eridnum = &eridnum.)); 590 nobs + 1; 591 call symputx("nobs", nobs); 592 dia run;</pre>		run;	
<pre>574 delete load_inputs; 575 quit; 576 577 data mylib.load_inputs; 578 578 579 keep eridnum renewal_opt_in resource_type resource_name baseline 580 analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; 581 582 set qse_erid_list_1 (where = (eridnum = &eridnum.)); 583 584 output; 585 stop; 586 587 run; 589 proc datasets lib = mylib noprint; 589 delete site_inputs; 591 quit; 593 data mylib.site_inputs; 594 deta mylib.site_inputs; 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 set okay_ers_ttab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 603 run; 603 run;</pre>			
<pre>guit; guit; data mylib.load_inputs; data mylib.load_inputs; keep eridnum renewal_opt_in resource_type resource_name baseline analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; set gse_erid_list_1 (where = (eridnum = &eridnum.)); dutput; set output; set output; guit; guit; guit; guit; guit; guit; guit; guit; guit; guid data mylib.site_inputs; data mylib.site_inputs; data mylib.site_inputs; set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); nobs + 1; cull symputx("nobs", nobs); dutputs("nobs", nobs); dutputs("nobs"</pre>			
<pre>576 577 data mylib.load_inputs; 578 579 keep eridnum renewal_opt_in resource_type resource_name baseline 580 analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; 581 582 set qse_erid_list_1 (where = (eridnum = &eridnum.)); 583 584 output; 585 stop; 586 589 proc datasets lib = mylib noprint; 589 data mylib.site_inputs; 591 quit; 592 data mylib.site_inputs; 593 data mylib.site_inputs; 594 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen_gen_type load_desc; 598 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 603 run;</pre>	- · -	-	
<pre>577 data mylib.load_inputs; 578 keep eridnum renewal_opt_in resource_type resource_name baseline 579 analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; 581 set qse_erid_list_1 (where = (eridnum = &eridnum.)); 583 output; 584 output; 585 stop; 586 stop; 586 generation of the stop is a stop</pre>		quit;	
<pre>578</pre>		data walib load inputat	
<pre>579 keep eridnum renewal_opt_in resource_type resource_name baseline 580 analyze_offer1 - analyze_offer4 blout1 - blout4 load ss_gen non_ss_gen; 581 582 set qse_erid_list_1 (where = (eridnum = &eridnum.)); 583 output; 584 output; 585 stop; 586 587 run; 589 proc datasets lib = mylib noprint; 590 delete site_inputs; 591 quit; 593 data mylib.site_inputs; 594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_dees; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run; 505 run; 506 run;</pre>		data myin. Ibau_inputs;	
<pre>set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); set okay_er</pre>		keen eridnum renewel opt in recourse two recourse news beseline	
<pre>set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); dot nobs + 1; call symputx("nobs", nobs); end function</pre>			
<pre>set qse_erid_list_1 (where = (eridnum = &eridnum.)); set qut; set qut; set qse_erid_list_1 (where = (eridnum = &eridnum.)); set qut; set qut;</pre>		analyze_offeri - analyze_offeri > broadt - broadt ss_gen non_ss_gen;	
<pre>583</pre>		set ase erid list 1 (where = (eridnum = &eridnum.)):	
<pre>584 output; 585 stop; 586 587 run; 588 589 proc datasets lib = mylib noprint; 590 delete site_inputs; 591 quit; 592 593 data mylib.site_inputs; 594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia_tspdsp_ia_year tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run;</pre>			
<pre>stop; s</pre>		output:	
<pre>586 587 run; 588 589 proc datasets lib = mylib noprint; 590 delete site_inputs; 591 quit; 592 593 data mylib.site_inputs; 594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 595 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen_gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 603 run;</pre>			
<pre>588 589 proc datasets lib = mylib noprint; 590 delete site_inputs; 591 quit; 592 593 data mylib.site_inputs; 594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run; </pre>	586		
<pre>589 proc datasets lib = mylib noprint; 590 delete site_inputs; 591 quit; 592 593 data mylib.site_inputs; 594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run;</pre>	587	run;	
<pre>590 delete site_inputs; 591 quit; 592 593 data mylib.site_inputs; 594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run;</pre>	588		
<pre>591 quit; 592 593 data mylib.site_inputs; 594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia_tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run;</pre>	589	proc datasets lib = mylib noprint;	
<pre>592 593 data mylib.site_inputs; 594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run; </pre>	590	delete site_inputs;	
<pre>593 data mylib.site_inputs; 594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run; </pre>	591	quit;	
<pre>594 595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run; </pre>	592		
<pre>595 keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street 596 city zipcode tspdsp_ia tspdsp_ia_year tspdsp_name tspdsp_duns noie_area wsale_meter 597 priv_net onsite_gen gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run; </pre>	593	data mylib.site_inputs;	
<pre>596</pre>	594		
<pre>597 priv_net onsite_gen_gen_type load_desc; 598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run;</pre>	595	keep esiid sub_metering unique_meter_id load_resource lr_in_as site_ownr site_name street	
<pre>598 599 set okay_ers_rtab_inputs (where = (eridnum = &eridnum.)); 600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run; </pre>	596		
<pre>599</pre>		priv_net onsite_gen gen_type load_desc;	
600 nobs + 1; 601 call symputx("nobs", nobs); 602 603 run;			
601 call symputx("nobs", nobs); 602 603 run;			
602 603 run;			
603 run;		Call symputx("nobs", nobs);	
Image: Step 15 DDE_Code	603	run;	•
🗟 Output - (Untitled) 🖹 Log - (Untitled) 🕅 🕅 Step 15 DDE_Code	6		>
	🔡 Outp	out - (Untitled)	
ouble click to change current folder 🔤 Otheraish/My Documents/sas user conf Ln 589, Col 1	ouble clic	k to change current folder 🤤 O:\craish\Wy Documents\sas user conf Ln 58	9, Col 1

The next data step, shown below, issues DDE commands to activate the tab being processed in either the 'id_template_copy.xlsx' or the 'id_alt_template_copy.xlsx' workbooks and to run Excel macros that also are stored in 'personal.xls'. Initially, SAS activates the tab and runs a macro called 'copy_paste_rtab_template_to_id_template' or 'copy_paste_rtab_template_to_id_alt_template', which is shown below. The macro does a 'copy – paste' from 'rtab_template.xlsx' (only one tab in this workbook) to the appropriate tab (which was previously exported by SAS using the 'fill' data set) on either the 'id_template_copy.xlsx' or the 'id_alt_template_copy.xlsx' workbook. This copies all the fixed information and all the links to the destination workbook.

SAS then runs the Excel macro, 'copy_paste_values_alt_tab' shown below to eliminate all the links on the destination tab. The final set of macros deletes unneeded formats and rows from the destination tab and runs the 'expand_cols' macro described earlier. We have also installed the Excel add-in called 'clear excess formats' and frequently run it manually on the final workbooks ... many of ours become quite large, and the add-in reduces them size by large amounts.

🕏 SAS - [Ste	ep 15 DDE_Code_Write out ok erids.sas]
🛃 <u>F</u> ile <u>E</u> dit	View Tools Run Solutions Window Help
I	I I I I I I I I I I I I I I I I I I I
631	data _null_;
632	file ddecmds;
633	<pre>put '[workbook.activate("'"&tab_name."'")]';</pre>
634	if "ID_Alt_Template" = "&ID_Template."
635	<pre>then put '[run("PERSONAL.XLS!copy_paste_rtab_template_to_id_alt_template")]';</pre>
636	<pre>else put '[run("PERSONAL.XLS!copy_paste_rtab_template_to_id_template")]';</pre>
637	<pre>put '[run("PERSONAL.XLS!copy_paste_values_alt_tab")]';</pre>
638	<pre>put '[run("PERSONAL.XLS!delete_stuff")]';</pre>
639	if &nobs. < 40 then put '[run("PERSONAL.XLS!delete_51")]';
640	else if &nobs. < 75 then put '[run("PERSONAL.XLS!delete_101")]';
641	else if &nobs. < 250 then put '[run("PERSONAL.XLS!delete_301")]';
642	else if &nobs. < 400 then put '[run("PERSONAL.XLS!delete_501")]';
643	<pre>put '[run("PERSONAL.XLS!expand_cols")]';</pre>
644	stop;
645	
646	run;
<u><</u>	
🔡 Output - (Untitled) 🖹 Log - (Untitled) 🕅 Step 15 DDE_Code
🖃 D:\test	Ln 645, Col 17

```
Sub copy_paste_rtab_template_to_id_alt_template()
'
copy_paste_rtab_template_to_id_alt_template macro
'
'
Windows("Rtab_Template.xlsx").Activate
Cells.Select
Selection.Copy
Windows("ID_Alt_Template_copy.xlsx").Activate
ActiveSheet.Paste
Range("A1").Select
End Sub
```

```
Sub copy_paste_values_alt_tab()
' copy_paste_values_alt_tab Macro
'
Range("A1:AF1500").Select
Selection.Copy
Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks
:=False, Transpose:=False
Application.CutCopyMode = False
Selection.NumberFormat = "0"
Range("A2").Select
End Sub
```

🚰 Microsoft Visual Basic - PERSONAL.XLS - [Module1 (Code)]					
🦂 Eile Edit View Insert Format Debug Run Iools Add-Ins Window Help 🛛 Type a question for help 🔽 🗖	×				
ें 🔀 🔤 - 🛃 🕺 🖻 🛍 🗚 🔊 (*) 🕨 💷 🕍 💸 🚰 😚 🎌 🞯 🛯 Ln 98, Col 1 💦 🖕					
(General) Copy_paste_values_id_tab					
	•				
Sub delete_501()					
' delete 501 Macro					
' Macro recorded 1/28/2011 by ERCOT					
Rows("501:1500").Select					
Selection.ClearContents Range("A1").Select					
End Sub					
Sub delete_stuff()					
' delete_stuff Macro					
' Macro recorded 1/28/2011 by ERCOT					
Cells.Replace What:="[Rtab_Template.xlsx]", Replacement:="",					
LookAt:=xlPart, SearchOrder:=xlByRows, MatchCase:=False, SearchFormat:= False, ReplaceFormat:=False					
End Sub					
	-				

The final section of code issues DDE commands to Excel to save and close the three open workbooks. The currently active workbook at the beginning of this code is either 'id_template_copy.xlsx' or 'id_alt_template_copy.xlsx', so the first 'save as' command applies to this workbook. The 'save as' command is constructed in the required format using macro variables for the path and filename. Double quotes are used inside the 'save as' command to allow the macro variables to be resolved, the single quotes are needed by the syntax of the DDE command, and the %unquote unmasks the single quotes created by the %str function to turn the single quotes into values the SAS compiler recognizes. The 'sleep' commands are all used to give time for the Excel command to complete before the next command is issued by SAS. SAS then issues a DDE close command to 'close' the currently active workbook.

The next two 'save as' and 'close' commands are issued to close the remaining two open workbooks, and, when the second one is closed, a 'quit' command is issued to close down Excel. These commands are issued to allow SAS to deal directly with closing Excel with no user intervention required inside Excel; a simple close command will make Excel open a dialog box asking the user whether the workbook should be saved. Since ERCOT frequently runs large numbers of QSEs in a single batch, this smoothes out the process considerably.

The final data step issues system commands to delete the two dummy workbooks that created by the previous step as well as either the 'id_template_copy.xlsx' or 'id_alt_template_copy.xlsx' workbooks. This is done as a clean-up to delete workbooks that are no longer needed and to remove potential conflicts with subsequent iterations inside the 'run_qse' macro.

	- [Step 15 DDE_Code_Write out ok erids.sas]
Eile Eile	Edit View Iools Run Solutions Window Help
~	N 🕞 🖬 🚔 🖬 🚔 🕰 🕹 🖿 👘 🛍 🕫 🎽 🛠 🕚 🥔
655	data _null_;
656	file ddecmds;
657	
658	<pre>put %unquote(%str(%'[SAVE.AS("&local_drive.\&eridpath.\RPT.00013012.&qseduns_fnameerids_okay.xlsx")]%'));</pre>
659	rc = sleep(2);
660	put '[close]';
661 662	rc = sleep(2);
662 663	<pre>put %unquote(%str(%'[SAVE.AS("&local drive.\&templatepath.\dummy1.xlsx")]%'));</pre>
664	<pre>rc = sleep(2);</pre>
665	<pre>put '[close]';</pre>
666	rc = sleep(2);
667	10 - Sleep(2),
668	<pre>put %unquote(%str(%'[SAVE.AS("&local drive.\&templatepath.\dummy2.xlsx")]%'));</pre>
669	rc = sleep(2);
670	put '[close]';
671	rc = sleep(2);
672	put '[quit()]';
673	stop;
674	run;
675	
676	/* delete template copy and two dummy files between gses */
677	
678	data null;
679	length command \$ 200;
680	command="\$local_drive.";
681	call system(command);
682	command="cd\stemplatepath.\";
683	call system(command);
684	command="del &ID_Templatecopy.xlsx";
685	call system(command);
686	command="del dummy1.xlsx";
687	call system(command);
688	command="del dummy2.xlsx";
689	call system(command);
690	
691	run;
🔄 Out	put - (Untitled)
D:\tes	Ln 645, Col 17

Some Additional Tips

When we first started running this code all inputs and outputs to Excel were run using DDE commands; use of the Excel libname engine greatly speeds up and improves the accuracy and reliability of the processes. Our latest iteration of the code, as described, eliminates as much as possible use of DDE commands.

From the beginning we have always run a verify step after writing to Excel workbooks with SAS. Though communication errors seem to have been eliminated with this version of the code, we still, from time to time, find other 'logic' errors when we read the workbook back in with code similar to that shown in the beginning of this paper and running proc compare to compare the resulting SAS dataset to the one used for the output code.

The code is fun to watch because both SAS and Excel are actively doing things on your desktop; it's helpful to keep an eye on the Excel, if the code appears to be hung up. In many cases it's because Excel has encountered a condition, usually an error, requiring user input via a dialog box. In these cases, you need to cancel submitted code in SAS and also shut down Excel manually.

Not surprisingly, diagnosing errors, particularly distinguishing between SAS errors and Excel macro errors is a bit mysterious ... the error messages are usually very uninformative.

APPENDIX A

Complete Read-in Code

```
options symbolgen mlogic mprint linesize = 100 msglevel = i;
%global version error;
%let version = 12.3;
%let contract period = 2012 OctJan13;
%let version error = No;
%let eridpath = d:\test;
%let offerpath = d:\test;
%let reportpath = d:\test;
libname perm "d:\test";
data fnames_list (Drop = rc fileid filecount i) ;
  length filename $ 300 ;
  rc = filename("Input", "&offerpath.");
     fileid = dopen("Input");
     filecount = dnum(fileid);
  DO i = 1 to filecount;
    filename = dread(fileid,i);
    if upcase(scan(filename,-1,'.')) in('XLS ', 'XLSM', 'XLSX') then Output;
  end;
     rc = DClose(fileid);
run;
/* create a macro variable of the number of filenames in the folder */
data null ;
  set FNames_List nobs = numfiles;
  call symputx("numfiles", put(numfiles, 4.));
  stop;
run;
%let i = 2;
  %do i = 1 %to &numfiles.;
/* read each workbook and all R-tabs and accumulate in dataset */
%macro read work book;
  %let first = 1;
  %do i = 2 %to 2;
    data _null_ ;
```

```
obsnum = input("&i.",4.);
     set fnames List point = obsnum;
     call symputx("filename", put(filename, $300.));
     stop;
   run;
   libname mylib "&offerpath.\&filename." mixed = yes header = no;
     ods trace on;
     ods output "Library Members" = tabnames;
     proc datasets library = mylib;
     quit;
     ods output close;
     ods listing;
   %let numbad = 0;
     data tab names 1 (keep = filename tab name) bad tab names (keep = filename
tab name);
     length tab name $ 35 filename $ 300;
     set tabnames;
     filename = "&filename.";
     if index(name, "''") > 0 then tab name = "'" || trim(tranwrd(name, "''", "'")) ||
"'";
     else tab name = name;
     if length(trim(tab name)) >= 4 then do;
       if index(substr(tab name, 2, length(trim(tab name)) - 2), "'") > 0 &
         index(substr(tab name, 2, length(trim(tab name)) - 2), "\&") > 0 then do;
         output bad tab names;
             call symputx("numbad",1);
       end;
         else do;
         if upcase(substr(tab name, 1, 4))
           in('AVAI', 'BASE', 'EXCE', 'ID I', 'IDEN', 'R$', 'SHEE')
           index(tab name, 'Print Area') > 0 then delete;
         else output tab_names_1;
       end;
     end;
       else if upcase(tab name) ^= 'R$' then output tab names 1;
     run;
   %if &numbad. = 1 %then %do;
     %put ;
     %put %str(ERROR: !!!!!!!!!! Tab Name with Invalid Characters in Workbook
%put ;
   %end;
   %let dset = mylib.'identification$'n ;
   %let dsid2 = %sysfunc(open(&dset.));
   %if &dsid2. %then %do;
     %let rc = %sysfunc(close(&dsid2.));
   %end;
   %let rc=%sysfunc(close(&dsid2.));
   data miss id tab;
```

```
length filename $ 300;
 filename = "";
run;
%if &dsid2. = 0 %then %do;
  %put ;
 %put ;
 data miss id tab;
   length filename $ 300;
   filename = "&filename.";
 run;
%end;
%else %do;
 data version (keep = version id temp merge var);
   length version id $ 5;
   set mylib.'identification$'n;
   version id = compress(f27);
   temp merge var = 1;
   output;
   stop;
  run;
  data identification (keep = qsename qseduns temp merge var id info1 - id info23);
    length qsename qseduns id info1 - id info23 $64.;
    array id_info{23} $ id info1 - id info23;
    retain qsename qseduns id info1 - id info23;
    set mylib.'id input area'n ;
    if _n_ = 1 then qsename = f2;
    else if _n_ = 2 then qseduns = f2;
if (_n_ >= 3 & _n_ <= 4) | (_n_ >= 6 & _n_ <= 11) | (_n_ >= 13 & _n_ <= 17) | (_n_ >= 19 & _n_ <= 23) | (_n_ >= 25 & _n_ <= 29) then do;
     i + 1;
     id info{i} = f2;
     if i = 2 then id info2 = upcase(id info2);
     temp merge var = 1;
      if i = 23 then output;
    end;
  run;
  data identification 1;
   merge identification version;
   by temp merge var;
  run;
  data _null_;
   set tab_names_1 nobs = numtabs;
   put numtabs=;
   call symputx("numtabs", put(numtabs, 4.));
   stop;
```

run;

```
do j = 1  to anumtabs.;
        data tab names 2 (keep = tab name temp merge var) ;
          length tab name quote $ 300;
          obsnum = input("&j.",4.);
          set tab names 1 point = obsnum;
              if substr(tab name, 1, 1) = "'" & substr(tab name, length(trim(tab name)),
1) = "''"
            & index(substr(tab_name, 2, length(trim(tab name)) - 2), "'") > 0 then do;
               tab name quote = 'sheet = "' || trim(put(tab name,$300.)) || '"';
            call symputx("sheet", tab name quote);
          end;
              else if substr(tab name, 1, 1) = "'" & substr(tab name,
length(trim(tab name)), 1) = "'" then do;
                tab name quote = "sheet = " || trim(put(tab name,$300.));
            call symputx("sheet", tab name quote);
          end;
              else do;
                tab name quote = "sheet = '" || trim(put(tab name, $300.)) || "'";
                  call symputx("sheet", tab name quote);
          end:
              temp merge var = 1;
              output;
          stop;
        run;
            proc import datafile="&eridpath.\&filename." replace out = import resource
dbms = excel;
         mixed = yes;
          getnames = no;
          scantext = yes;
          &sheet.;
        run;
            data import resource 1;
                  set import resource;
                  temp merge var = 1;
            run;
        data resource (keep = version resource filename tab name renewal opt in
resource type
            resource name baseline analyze offer1 - analyze offer4
            esiid sub metering unique meter id load resource lr in as site ownr site name
            street city zipcode tspdsp ia tspdsp ia year tspdsp name tspdsp duns
            noie area wsale meter priv net onsite gen gen type load desc
            temp merge var eridnum rundate);
          length version resource $ 5 filename $ 300 tab name $ 35
            resource_type resource_name baseline $ 64 renewal opt in $ 3
            esiid $ 64 sub metering $ 3 unique meter id $ 64 load resource lr in as $ 3
            site ownr site name $ 64 street city $64 zipcode $5. tspdsp ia $ 3
```

```
tspdsp_ia_year tspdsp_name tspdsp_duns $64. noie_area wsale meter
            priv net onsite gen $3. gen type $30. load desc $256.
            analyze offer1 - analyze offer4 $ 3;
          format rundate datetime20.;
          retain version resource temp merge var filename tab name renewal opt in
resource type
            resource name baseline analyze offer1 - analyze offer4 eridnum rundate;
          merge import resource 1 tab names 2;
              by temp_merge_var;
              if tab name = 'Alt' then baseline = 'ALTERNATE';
              if n = 1 then do;
                temp merge var = 1;
                        *tab name = substr("&sheet.", 9, 1);
                tab name = translate(tab name, " ", "$");
                filename = "&filename.";
                rundate = datetime();
                eridnum = round(100 * (rundate - "01jun2009:01:00:00"dt),1);
                version resource = f27;
              end;
              else if _n >= 2 & _n <= 5 & tab_name ^= 'Alt' then do;
                if n = 2 then do;
                 renewal opt in = '';
                 resource type = f3;
                  analyze offer1 = upcase(f5);
                end;
                else if _n_ = 3 then analyze_offer2 = upcase(f5);
                else if n_{-} = 4 then do;
                 resource name = f1;
                 if resource name = '' then resource name = 'Missing Resource Name';
                 baseline = upcase(f3);
                  analyze offer3 = upcase(f5);
                end;
                else if n = 5 then analyze offer4 = upcase(f5);
              end;
              else if (tab name = 'Alt' & n >= 4) | (tab name ^{=} 'Alt' & n >= 7) then
do;
                esiid = compress(f1, "' ");
                sub metering = upcase(f2);
                unique meter id = f3;
                if esiid > '' & unique meter id = '' & sub metering = ''
                 then sub metering = 'NO';
                load resource = upcase(f4);
                lr in as = upcase(f5);
                site_ownr = f6;
                site name = f7;
                street = f8;
                city = f9;
                zipcode = f10;
                tspdsp ia = upcase(f11);
                tspdsp ia year = f12;
                tspdsp name = f13;
                tspdsp duns = f14;
                noie_area = upcase(f15);
                wsale meter = upcase(f16);
                priv net = upcase(f17);
                onsite gen = upcase(f18);
                gen type = f19;
                load desc = f20;
                if esiid > '' | unique meter id > '' then output;
```

```
else if esiid = '' & unique meter id = ''
                  & (sub_metering > '' | unique_meter_id > '' | load_resource > '' |
lr_in as > ''
                  | site ownr > '' | site name > '' | street > '' | city > '' | zipcode >
1.1
                  | tspdsp ia > '' | tspdsp ia year > '' | tspdsp name > '' | tspdsp duns
> ''
                  | noie area > '' | wsale meter > '' | priv net > '' | onsite gen > ''
                  | gen type > '' | load desc > '') then
                  put /// 'esiid and umi both blank ' filename = tab name = /// '';
              end:
            run;
        %let dset = work.resource;
        %let dsid3 = %sysfunc(open(&dset));
        %if &dsid3 %then %do;
          %let numloads =%sysfunc(attrn(&dsid3,NOBS));
          %let rc = %sysfunc(close(&dsid3));
        %end;
        %let rc=%sysfunc(close(&dsid3));
        %if &numloads. > 0 %then %do;
          data combine id resource bad version (keep = qsename eridnum);
            merge identification 1 (in = in1) resource (in = in2);
            by temp merge var;
            if ^in1 | ^in2 | compress(version id) ^= "&version."
              | compress(version resource) ^= "&version." then do;
             output bad version;
              call symputx("version error", "Yes");
            end;
            output combine id resource;
          run;
          proc sort data = bad version nodupkey;
           by eridnum;
          run;
          proc sort data = combine id resource;
           by eridnum;
          run;
          data combine id resource 1 bad version 1 (keep = qsename filename tab name);
            merge combine id resource bad version (in = in1);
            by eridnum;
            if ^in1 then output combine id resource 1;
            else if first.eridnum then output bad version 1;
          run;
          %if &first. = 1 %then %do;
            data all_resources;
             set combine id resource 1;
            run;
            data all bad versions;
              set bad version 1;
            run;
```

```
data all_tab_names;
          set tab_names_1;
        run;
        data all bad tab names;
          set bad tab names;
        run;
        data all miss id tab;
         set miss id tab (where = (filename > ''));
        run;
        %let first = 0;
      %end;
      %else %do;
        proc append base = all resources data = combine id resource 1;
        run;
        proc append base = all_bad_versions data = bad_version_1;
        run;
        proc append base = all tab names data = tab names 1;
        run;
      %end;
    %end;
  %end;
%end;
proc append base = all bad tab names data = bad tab names;
run;
proc append base = all miss id tab data = miss id tab (where = (filename > ''));
run;
```

```
%end;
```

%mend;

%read_work_book;

libname mylib clear;

%macro check errors;

%let tabs_not_read = No;

```
data all_tab_names_1;
```

```
set all_tab_names;
```

tab_name = translate(tab_name,' ', '\$');

```
proc sort data = all_tab_names_1 out = all_tab_names 2 nodupkey;
   by filename tab name;
 run;
 proc sort data = all resources out = all resources 1;
   by filename tab name;
 run;
 data tabs not read (keep = filename tab name);
   merge all tab names 2 (in = in1) all resources 1 (in = in2);
   by filename tab name;
   if in1 & ^in2;
   call symputx("tabs not read", "Yes");
 run;
 %if &tabs not read. = Yes %then %do;
   %put;
   %put %str(ERROR: !!!!!!!!!!!!!!!! Tabs Not Read Error - Check 'tabs_not_read' sas
data set !!!!!!!!!!!!!!!!!!!!!!!!!!!!;
   %put;
 %end;
 %let all bad tab names = No;
 data null ;
   set all bad_tab_names ;
   if filename > '';
   call symputx("all bad tab names", "Yes");
 run;
 %if &all bad tab names. = Yes %then %do;
   %put ;
   %put %str(ERROR: !!!!!!!!!!!!!! Tab Name(s) with Invalid Characters found in Workbook
- Check 'all bad tab names' sas data set !!!!!!!!!!!!!!!! );
   %put ;
 %end;
 %if &version error. = Yes %then %do;
   %put;
   %put;
 %end;
```

```
%let all_miss_id_tab = No;
data _null_;
  set all_miss_id_tab ;
  if filename > '';
  call symputx("all_miss_id_tab", "Yes");
```

run;

%mend;

%check_errors;

APPENDIX B

Complete Write-out Code

```
options noxwait noxsync symbolgen mlogic mprint linesize = 200 msglevel = i;
%let contract period = 2012 OctJan13;
path ;
/* path to write xls files and to see if they already exist for this batch */
%let eridpath = Emergency Interruptible Load\&contract period.\00 resource
Identification\3 Outgoing erids\okay;
%let templatepath = Emergency_Interruptible_Load\&contract period.\00 resource
Identification\DDE Templates;
libname perm "Z:\Emergency Interruptible Load\&contract period.\00 resource
Identification\sas datasets";
  /* create dataset of gsenames and count of erids */
 data okay ers inputs (keep = eridnum qsename qseduns renewal opt in resource name
   resource type load type baseline analyze offer1 - analyze offer4 blout1 - blout4
   load ss gen non ss gen esiid sub metering unique meter id load resource
   lr in as site ownr site name site name street city zipcode tspdsp ia tspdsp ia year
   tspdsp_name tspdsp_duns noie_area wsale_meter priv_net onsite_gen_gen_type load_desc
   id info1 - id info23);
 length eridnum 8 qsename qseduns $ 64 renewal_opt_in $ 3 resource_type resource_name
   baseline $ 64 analyze offer1 - analyze offer4 $ 3 blout1 - blout4 blout1 - blout4
load
   ss gen non ss gen $ 30 esiid $ 64 sub metering $ 3 unique meter id $ 64 load resource
   lr in as $ 3 site ownr site name street city $ 64 zipcode $ 5 tspdsp ia $ 3
tspdsp ia year
   tspdsp name tspdsp duns $ 64 noie area wsale meter priv net onsite gen $ 3 gen type $
30
   load desc $ 256 id info1 - id info23 $ 64;
 set perm.ers inputs okay final;
 renewal opt in = '';
 load = 'Load';
 ss gen = 'Self-Serving Gen';
 non ss gen= 'Non Self-Serving Gen';
 if bline1 = 1 then blout1 = 'DEFAULT-REGRESSION';
 if bline2 = 2 then blout2 = 'DEFAULT-MID 8 OF 10';
 if bline3 = 3 then blout3 = 'DEFAULT-MATCHING DAY PAIR';
 blout4 = 'ALTERNATE';
 if baseline = '' & bline1 = . & bline2 = . & bline3 = . then baseline = 'ALTERNATE';
run;
data okay ers inputs 1 (drop = temp tab name more i numbers letters);
 length tab name $ 29 temp tab name $ 64;
```

set okay_ers_inputs;

```
if resource name = '' then tab name = 'Alt';
  else do;
    temp tab name = compress(resource name, "'");
    temp tab name = compress(temp tab name, '"');
                                                                             ",
    temp tab name = translate(temp tab name,"
                        "!@#$%^&*()_+-={}|[]\:;<>?,./;");
    temp tab name = compbl(temp tab name);
    temp tab name = translate(temp tab name,"~", " ");
    more = 1;
    do i = 64 to 1 by - 1 while (more = 1);
      if substr(temp tab name, i, 1) = '~' then substr(temp tab name, i, 1) = ' ';
      else more = 0;
    end;
    tab name = compbl(substr(translate(temp tab name, ' ', '~'),1,26));
    if substr(tab name, 26,1) = ' ' then tab name = substr(tab_name, 1, 25);
    letters = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz';
    numbers = '1234567890';
    if verify(substr(tab name, 1, 3), letters) = 0
      & verify(compress(substr(tab name, 4)), numbers) = 0
      & substr(tab_name,4) > ''
      then tab name = substr(tab name,1,3) || ' ' || substr(tab name,4);
    else if verify(substr(tab name,1,2), letters) = 0
      & verify(compress(substr(tab name, 3)), numbers) = 0
      & substr(tab name, 3) > ''
      then tab_name = substr(tab_name,1,2) || '_' || substr(tab_name,3);
    else if verify(substr(tab name,1,1), letters) = 0
      & verify(compress(substr(tab name,2)), numbers) = 0
      & substr(tab name, 2) > ''
      then tab name = substr(tab name,1,1) || ' ' || substr(tab name,2);
    if verify(substr(tab name,1,1), letters) ^= 0 then tab name = ' ' || tab name;
    tab name = upcase(tab name);
  end;
run;
proc sort data = okay ers inputs 1;
  by qseduns tab name eridnum;
run;
data okay_ers_rtab_inputs okay_ers_alt_inputs ;
  drop count;
  set okay ers inputs 1;
  by gseduns tab name eridnum;
  if tab name ^= 'Alt' then do;
    if first.tab name then count = 0;
    if first.eridnum then count + 1;
    if count > 1 & count < 100 then
      tab name = translate(compbl(substr(tab name,1,26) || put(count,2.)),' ',' ');
    output okay ers rtab inputs;
  end;
  else output okay ers alt inputs;
run;
proc sort data = okay ers rtab inputs out = qse erid list nodupkey;
```

```
run;
```

by qseduns tab name eridnum;

```
proc freq data = okay ers inputs 1 noprint;
```

```
tables qsename * qseduns / out = qse list;
run:
  /* create macro variable of number of gses */
  %let numqses = 0;
data null ;
 set qse list nobs = numqses;
 call symputx("numqses", put(numqses, 5.));
 stop;
run;
%macro run qse;
 data fill;
   a = 1; b = 1; c = 1;
   if a > 1;
 run;
   /* loop through QSEs */
 %if &numqses. > 0 %then %do i = 1 %to &numqses.;
   data null ;
     obsnum = input("&i",4.);
     set qse list point = obsnum;
     call symputx("qseduns", put(qseduns, $64.));
     call symputx("qseduns fname", put(input(qseduns, 13.), z16.));
     stop;
   run;
     /* set macro numerids to the number of r tabs for qse being processed */
   %let numerids = 0;
   /* create a file of eridnums for this qse */
   data qse erid list 1 ;
     set qse erid list (where = (qseduns = "&qseduns."));
     numerids + 1;
     call symputx("numerids", numerids);
   run;
   /* set macro variable for the template to be used (with or without and Alt tab) */
   %let ID Template = ID Template;
   data null ;
     set okay ers alt inputs (where = (qseduns = "&qseduns."));
     call symputx("ID Template", "ID Alt Template");
     stop;
   run;
     /* copy ID Template.xlsx into ID Template copy.xlsx */
   data null ;
     length command $ 200;
     command="&local drive.";
         call system(command);
         command="cd\&templatepath.\";
         call system(command);
```

```
command="copy &ID Template..xlsx &ID Template. copy.xlsx";
     call system(command);
run;
%if &numerids. > 0 %then %do j = 1 %to &numerids.;
/* create tabs for all the R tab names needed for this qse */
  data _null_;
   obsnum = input("&j",4.);
   set qse erid list 1 point = obsnum;
   call symputx("tab name", tab name);
   stop;
  run;
  proc export data = fill outfile =
   "&local drive.\&templatepath.\&ID Template. copy.xlsx";
   sheet="&tab name.";
  run;
%end;
data messages erids okay ;
 set perm.messages erids okay;
 where qseduns = "&qseduns." ;
run;
proc sort data = messages erids okay ;
 by qsename resource_name tab_name eridnum esiid unique meter id;
run;
data rtabs with messages (keep = gsename resource name tab name eridnum)
  alttabs with messages
    (keep = qsename resource name tab name eridnum esiid unique meter id);
  set messages erids okay ;
  if message1 ^= '' | message2 ^= '' | message3 ^= '' | message4 ^= ''
   message5 ^= '' | message6 ^= '' | message n ^= ''
   message s ^= '' | message u ^= '';
  if tab name = 'Alt' then output alttabs_with_messages;
  else output rtabs with messages;
run;
proc sort data = rtabs with messages nodupkey;
 by gsename resource name tab name eridnum;
run;
proc sort data = alttabs with messages nodupkey;
 by qsename resource name tab name eridnum esiid unique meter id;
run;
data messages rtabs okay;
  length qsename qseduns $ 64 renewal opt in $ 3 resource name resource type
    load type baseline esiid $ 64 sub metering $ 3 unique meter id $ 64 load resource
    lr in as $ 3 street city $ 64 zipcode $ 5 message1 - message6 message n message s
   message_u $ 200 site_ownr site_name $ 64 tspdsp_ia $ 3 tspdsp_ia_year tspdsp_name
    tspdsp duns $ 64 analyze offer1 - analyze offer4 noie area wsale meter priv net
    onsite gen $ 3 gen type $ 30 load desc $ 256 filename $ 300 tab name $ 30;
  merge messages erids okay alttabs with messages (in = in1);
  by gsename resource name tab name eridnum esiid unique meter id;
```

```
if in1;
  renewal opt in = '';
run;
data messages alts okay;
  length qsename qseduns $ 64 renewal opt in $ 3 resource name resource type
    load type baseline esiid  64 sub metering  3 unique meter id  64 load resource
    lr in as $ 3 street city $ 64 zipcode $ 5 message1 - message6 message n message s
   message u $ 200 site ownr site name $ 64 tspdsp ia $ 3 tspdsp ia year tspdsp name
    tspdsp duns $ 64 analyze offer1 - analyze offer4 noie area wsale meter priv net
    onsite_gen $ 3 gen_type $ 30 load desc $ 256 filename $ 300 tab name $ 30;
 merge messages erids okay rtabs with messages (in = in1);
 by gsename resource name tab name eridnum;
  if in1;
  renewal opt in = '';
run;
data messages erids okay 1;
  length qsename qseduns $ 64 renewal opt in $ 3 resource name resource type
    load type baseline esiid $ 64 sub metering $ 3 unique meter id $ 64 load resource
    lr_in_as $ 3 street city $ 64 zipcode $ 5 message1 - message6 message n message s
   message u $ 200 site ownr site name $ 64 tspdsp ia $ 3 tspdsp ia year tspdsp name
    tspdsp duns $ 64 analyze offer1 - analyze offer4 noie area wsale meter priv net
    onsite gen $ 3 gen type $ 30 load desc $ 256 filename $ 300 tab name $ 30;
  set messages rtabs okay messages alts okay;
  by qsename resource name tab name eridnum esiid unique meter id;
  if eridnum > .;
  /* output resource level messages */
  if first.resource name | tab name = 'Alt' then output;
  /* blanks out load level info on the esiid rows */
  else do;
    qsename = ''; qseduns = ''; resource name = ''; baseline = '';
   message1 = '';
   output;
  end;
  /* insert blank line after last load */
  if last.eridnum & eridnum > . then do;
    qsename = ''; qseduns = ''; renewal opt in = ''; resource name = '';
    resource type = ''; load type = ''; baseline = ''; esiid = ''; sub metering = '';
    unique_meter_id = ''; load_resource = ''; lr_in_as = ''; street = ''; city = '';
    zipcode = ''; message1 = ''; message2 = ''; message3 = ''; message4 = '';
   message5 = ''; message6 = ''; message n = ''; message u = '';
    site ownr = ''; site name = ''; tspdsp ia = ''; tspdsp ia year = '';
    tspdsp name = ''; tspdsp duns = ''; analyze offer1 = ''; analyze offer2 = '';
    analyze offer3 = ''; analyze offer4 = ''; noie area = ''; wsale meter = '';
    priv net = ''; onsite gen = ''; gen type = ''; load desc = ''; eridnum = .;
    filename = ''; tab_name = '';
    output;
  end;
run;
proc export data = messages erids okay 1
  outfile = "&local drive.\&templatepath.\&ID Template. copy.xlsx" replace;
```

```
Sheet = "Exceptions";
run;
/* writes out availability results to workbooks */
data erid avail results;
 length resource name site name $ 64;
 set perm.erid avail results;
 where qseduns = "&qseduns." ;
run;
proc sort data = erid avail results ;
 by qsename resource name eridnum esiid unique meter id contract period
   time period;
run;
data erid avail results
  (keep = qsename resource name esiid unique meter id site name
   min esiid ct contract period time period min mw mw 1st pctile mw 95th pctile
    avg mw eridnum);
  length qsename resource_name site_name esiid unique_meter_id
    contract_period $ 64 time_period $12 offer_mw min_mw mw_lst_pctile
    mw_95th_pctile avg_mw min_esiid_ct eridnum 8 ;
  label qsename = "QSE Name" resource name = "Resource Name" esiid = "ESI ID"
    unique meter id = "Unique Meter Id" site name = "Site Name"
    contract period = "Analysis Months" time period = "Time Period"
    offer mw = "Offer MW" min mw = "Minimum Base Load MW"
   min esiid ct = "Number of Loads Analyzed"
   min mw = "Minimum MW"
   mw 1st pctile = "1st Percentile Load MW"
   mw 95th pctile = "95th Percentile Load MW" avg mw = "Average Load MW";
  set erid avail results;
  by gsename resource name eridnum esiid unique meter id contract period
    time period;
  output;
  /* make a blank row between each contract period */
  if last.contract period then do;
   qsename = '' ; resource name = ' '; esiid= ' '; unique meter id = '';
    site name = ' '; contract_period = ' '; time_period = ' '; offer_mw = .;
   min esiid ct = .; mw 95th pctile = .; avg mw = .; min mw = .; mw 1st pctile = .;
   eridnum = .; load resource = ''; lr in as = '';
   output;
  end:
  /* make a blank row between each contract period */
  if last.resource name then do;
   qsename = ''; resource name = ' '; esiid= ' '; unique meter id = '';
    site name = ' '; contract period = ' '; time_period = ' '; offer_mw = .;
   min esiid ct = .; mw_95th_pctile = .; avg_mw = .; min_mw = .; mw_1st_pctile = .;
    eridnum = .; load resource = ''; lr in as = '';
    output;
  end;
run;
proc export data = erid avail results
 outfile = "&local drive.\&templatepath.\&ID Template. copy.xlsx" replace;
  Sheet = "Availabilty Results";
run;
```

```
/* writes out baseline results to workbooks */
 data compare ranking report;
   length resource name site name $ 64;
   set perm.compare ranking report;
   where qseduns = "&qseduns." ;
 run;
 proc sort data = compare ranking report;
   by gsename resource name eridnum esiid unique meter id contract period;
 run;
 data export baseline info
    (keep = qsename resource name contract period esiid unique meter id
      site name RSquare MAPE P90 kW Confidence P95 kW Confidence
      P99 kW Confidence Baseline eridnum);
   length qsename resource name site name esiid unique meter id contract period
      Baseline $ 64 RSquare MAPE P90 kW Confidence P95 kW Confidence
      P99 kW Confidence eridnum 8. ;
   set compare ranking report;
   by qsename resource name eridnum esiid unique meter id site name contract period;
   where qseduns = "&qseduns." ;
    /* inserts blank row after load level messages*/
   output;
   if last.contract period then do;
     RSquare = .;
     MAPE = .;
     P90 kW Confidence = .;
     P95 kW Confidence = .;
     P99 kW Confidence = .;
     Baseline = ' ';
     resource name = ' ';
     qsename = '';
      contract_period = ' ';
      esiid = '';
     unique_meter_id = ' ';
      site_name = ' ';
     eridnum = .;
     qsenum = .;
     output;
   end;
 run;
 proc export data = export baseline info
   outfile = "&local drive.\&templatepath.\&ID Template. copy.xlsx" replace;
   Sheet = "Baseline Info";
 run;
 x "'C:\Program Files\Microsoft Office\OFFICE12\excel.exe'";
 data null ;
   rc = sleep(2);
   stop;
 run;
 filename ddecmds dde 'excel|system';
   /* open the input data.xlsx workbook */
```

```
data _null_;
      rc = sleep(1);
      file ddecmds;
     put %unquote(%str(%'[open("&local drive.\&templatepath.\input data.xlsx")]%'));
      stop;
    run;
    data _null_;
      rc = SLEEP(1);
      file ddecmds;
     put %unquote(%str(%'[open("&local drive.\&templatepath.\Rtab Template.xlsx")]%'));
    run;
    data null ;
      rc = SLEEP(1);
      file ddecmds;
     put
%unquote(%str(%'[open("&local drive.\&templatepath.\&ID Template. copy.xlsx")]%'));
    run;
    /* identification information for the QSE being processed */
    /* pull the input data needed from the permanent sas dataset */
    /* and output to the 'input data.xlsx' workbook in the 'id inputs' range */
    /* the 'ID Alt Template copy.xlsx' workbook which ends up being the saved qse
workbook */
    /* has links to the 'input data.xlsx' workbook */
    libname mylib "&local drive.\&templatepath.\input data.xlsx";
    proc datasets lib = mylib noprint;
      delete id inputs;
    quit;
    data mylib.id inputs;
      keep qsename qseduns id info1 - id info23 sub type1 sub type2;
      set okay ers inputs 1 (where = (qseduns = "&qseduns."));
      id info2 = '';
      sub type1 = '10m Procurement Offer';
      sub_type2 = '30m Procurement Offer';
      output;
      stop;
    run;
    data _null_;
      file ddecmds;
      put '[workbook.activate("Identification")]';
      put '[run("PERSONAL.XLS!copy paste values id tab")]';
      put '[workbook.activate("Exceptions")]';
      put '[run("PERSONAL.XLS!expand cols")]';
      put '[run("PERSONAL.XLS!clear lines")]';
      put '[workbook.activate("Availabilty Results")]';
      put '[run("PERSONAL.XLS!expand cols")]';
      put '[run("PERSONAL.XLS!clear lines")]';
      put '[workbook.activate("Baseline Info")]';
      put '[run("PERSONAL.XLS!expand cols")]';
      put '[run("PERSONAL.XLS!clear lines")]';
      stop;
```

```
%if &ID Template. = ID Alt Template %then %do;
      proc datasets lib = mylib noprint;
        delete load inputs;
      quit;
      data mylib.load inputs;
        keep eridnum renewal opt in resource type resource name baseline
          analyze offer1 - analyze offer4 blout1 - blout4 load ss gen non ss gen;
        set okay ers alt inputs (where = (qseduns = "&qseduns."));
        output;
        stop;
      run;
      proc datasets lib = mylib noprint;
        delete site inputs;
      quit;
      data mylib.site inputs;
        keep esiid sub metering unique meter id load resource lr in as site ownr
site name
          street city zipcode tspdsp ia tspdsp ia year tspdsp name tspdsp duns noie area
          wsale meter priv net onsite gen gen type load desc;
        set okay ers alt inputs (where = (qseduns = "&qseduns."));
        output;
        nobs + 1;
        call symputx("nobs", nobs);
      run;
      data _null_;
       file ddecmds;
        put '[workbook.activate("Alt")]';
        put '[run("PERSONAL.XLS!copy paste values alt tab")]';
        if &nobs. < 40 then put '[run("PERSONAL.XLS!delete 51")]';
        else if &nobs. < 75 then put '[run("PERSONAL.XLS!delete 101")]';
        else if &nobs. < 250 then put '[run("PERSONAL.XLS!delete 301")]';
       else if &nobs. < 400 then put '[run("PERSONAL.XLS!delete 501")]';
        put '[run("PERSONAL.XLS!expand cols")]';
        stop;
      run;
      proc datasets lib = mylib noprint;
        delete site inputs;
      quit;
    %end;
    %if &numerids. > 0 %then %do j = 1 %to &numerids;
      data null ;
        obsnum = input("&j", 4.);
        set qse erid list 1 point = obsnum;
        call symputx("eridnum", put(eridnum, 20.));
```

run;

```
call symputx("tab name", tab name);
        stop;
     run;
      /* delete contents of cells in the load inputs range ... excel engine cannot */
     proc datasets lib = mylib noprint;
        delete load inputs;
     quit;
     data mylib.load inputs;
        keep eridnum renewal opt in resource type resource name baseline
          analyze offer1 - analyze offer4 blout1 - blout4 load ss gen non ss gen;
        set qse erid list 1 (where = (eridnum = &eridnum.));
        output;
        stop;
     run;
      /* delete contents of cells in the load inputs range ... excel engine cannot
replace values */
     proc datasets lib = mylib noprint;
        delete site inputs;
     quit;
     data mylib.site inputs;
        keep esiid sub metering unique meter id load resource lr in as site ownr
site name
          street city zipcode tspdsp ia tspdsp ia year tspdsp name tspdsp duns noie area
          wsale meter priv net onsite gen gen type load desc;
        set okay ers rtab inputs (where = (eridnum = &eridnum.));
        nobs + 1;
        call symputx("nobs", nobs);
     run;
     data null ;
       file ddecmds;
       put '[workbook.activate("'"&tab name."'")]';
        if "ID Alt Template" = "&ID Template."
         then put '[run("PERSONAL.XLS!copy paste rtab template to id alt template")]';
        else put '[run("PERSONAL.XLS!copy paste rtab template to id template")]';
        put '[run("PERSONAL.XLS!copy paste values alt tab")]';
        put '[run("PERSONAL.XLS!delete stuff")]';
        if &nobs. < 40 then put '[run("PERSONAL.XLS!delete 51")]';
        else if &nobs. < 75 then put '[run("PERSONAL.XLS!delete 101")]';
        else if &nobs. < 250 then put '[run("PERSONAL.XLS!delete 301")]';
        else if &nobs. < 400 then put '[run("PERSONAL.XLS!delete 501")]';
       put '[run("PERSONAL.XLS!expand cols")]';
        stop;
     run;
   %end;
   libname mylib clear;
      /* save and close the template copy workbook with an MIS compatible file name */
```

```
/* and save and close the two dummy files for the other two open workbooks ^{\prime}
      /* (this avoids having to respond in excel to okay losing the changes) */
    data null ;
      file ddecmds;
      put
%unquote(%str(%'[SAVE.AS("&local drive.\&eridpath.\RPT.00013012.&qseduns fname..erids oka
y.xlsx")]%'));
      rc = sleep(2);
      put '[close]';
      rc = sleep(2);
      put %unquote(%str(%'[SAVE.AS("&local drive.\&templatepath.\dummy1.xlsx")]%'));
      rc = sleep(2);
      put '[close]';
      rc = sleep(2);
      put %unquote(%str(%'[SAVE.AS("&local drive.\&templatepath.\dummy2.xlsx")]%'));
      rc = sleep(2);
      put '[close]';
      rc = sleep(2);
     put '[quit()]';
      stop;
    run;
      /* delete template copy and two dummy files between gses */
    data null ;
      length command $ 200;
      command="&local drive.";
          call system(command);
          command="cd\&templatepath.\";
          call system(command);
      command="del &ID Template. copy.xlsx";
          call system(command);
      command="del dummy1.xlsx";
          call system(command);
      command="del dummy2.xlsx";
          call system(command);
    run;
```

%end;
%mend;

%run_qse;