

Metadata-based Auto-Programming Process

Lisa Lyons, Janssen R&D US

Qian Zhao, Harry Chen, Janssen R&D China

Introduction

Introduction

- ▶ Traditionally the process for programming analysis datasets(e.g. ADaM) is cumbersome and heavily relies on manual work
- ▶ Regulatory agencies enforcing more standardized data and industry moving towards standardizing analysis requirements
 - ▶ We need to achieve more automation for analysis by reusing standard and previous study algorithms stored in metadata to generate SAS code.
- ▶ Illustrate the concept of metadata-based auto programming in ADaM dataset creation
 - ▶ **Macro Level** using ADaM planning specifications and Macro level metadata.
 - ▶ **Data Step Level** using a structured metadata approach for IF-THEN/ELSE logic

Example 1:

Macro Level Metadata

Macro Level Metadata

Example 1

- ▶ Automatically generate ADAE.sas
 - ▶ Input → ADaM planning sheet (specifications)
 - ▶ Process → Extracting macro call information from standard macro and program folders and convert to metadata

ADAE.xlsx [Compatibility Mode] - Excel

ADAE.xlsx Specification

	F	G	H	J	N	R	U	V	W
1	DATASET	VARNAME	VARVER	VARLABEL	DERVORD	CODELST	ORIGCOM	DERVCOPY	SASMCR
55	ADAE	TRTEMFL	1	Treatment Emergent Analysis Flag	09	ADYES	If ADSL.ARFSTDT=<ASTDT=< ADSL.TRTEDT + [xx] days then TRTEMFL= 'Y'. See	Derived	%ADTRTEMFL
66	ADAE	AERELN	1	Causality (N)	10	RELN	Code AE.AEREL to numeric value. Low relation should correspond to low value.	Derived	%ADREL
67	ADAE	RELGR1	1	Pooled Causality Group 1	10	RELGR	If AEREL is 'NOT RELATED' or 'DOUBTFUL' then RELGR1 is equal to 'NOT RELATED'. Otherwise, if	Derived	%ADREL
68	ADAE	RELGR1N	1	Pooled Causality Group 1 (N)	10	RELGRN	Code RELGR1 to numeric.	Derived	%ADREL

Instructions ADM Dataset ADAE ADM Codelist Release History

Macro Level Metadata

Example 1 (cont'd)

- ▶ Automatically generate ADAEfmt.sas
 - ▶ Input → ADaM planning sheet (specifications)
 - ▶ Process → output format/informat from Codelist

ADAE.xls [Compatibility Mode] - Excel

ADAE.xlsx Specification

File Home Insert Draw Page Layout Formulas Data Review View Power Pivot Tell me what you want to do

F47

	C	D	E	F	G	H	I	J
1	REFERENCE	TXAREA	DSAREA	CODELST	DATATYPE	CODEVAL	RNK	DECOD
9	ADaM	Global	General	RELN	integer	1	1	NOT RELATED
10	ADaM	Global	General	RELN	integer	2	2	DOUBTFUL
11	ADaM	Global	General	RELN	integer	3	3	POSSIBLE
12	ADaM	Global	General	RELN	integer	4	4	PROBABLE
13	ADaM	Global	General	RELN	integer	5	5	VERY LIKELY
14	ADaM	Global	General	RELN	integer	6	6	MULTIPLE
15	ADaM	Global	General	RELGR	text	NOT RELATED	1	NOT RELATED
16	ADaM	Global	General	RELGR	text	RELATED	2	RELATED
17	ADaM	Global	General	RELGRN	integer	1	1	NOT RELATED
18	ADaM	Global	General	RELGRN	integer	2	2	RELATED
28	ADaM	Global	General	ADYES	text	Y		Yes

Instructions ADM Dataset ADAE ADM Codelist Release ...

Macro Level Metadata

Example 1 (cont'd)

► %uacADaMCode macro call:

```
Remarks          : The ADMacrodir, studyADPrograms, and domain parameters can
                   : have multiple separated by a pipe (|).
                   : For example, &path1 | &path2 for ADMacrodir and studyADPrograms.
                   : For example, ADAE | ADVS for domain.

Macro Parameters   : ADMacrodir - paths to reference macro folders
                   : studyADPrograms - paths to reference program folders
                   : metadir - path to your metadata that will be used to create you code
                   : domain - domains you want to create
                   : outDir - output directory path write out files

*****/
%uacADaMCode(ADMacrodir = &topdrv.\1_GLOBAL\AnalysisStandards_Test Release\Macros
             ,metadir   = &opath
             ,studyADPrograms = &topdrv.\1_GLOBAL\AnalysisStandards_Test Release\Programs
             ,domain     = ADAE
             ,outDir     = &opath);
```

Macro Level Metadata

Example 1 (cont'd)

Extracting information (by Perl Regular Expression) from standard macro and program folders and converting it to macro level metadata

Macro Definition Metadata:

macName	paramn	param	equal	paramval
%MACRO ADREL	1	indsn	=	
%MACRO ADREL	2	outdsn	=	
%MACRO ADREL	3	keepvars	=	
%MACRO ADREL	4	debug	=	OFF
%MACRO ADTRTEMFL	1	indsn	=	
%MACRO ADTRTEMFL	2	invar	=	
%MACRO ADTRTEMFL	3	adrule	=	
%MACRO ADTRTEMFL	4	outdsn	=	
%MACRO ADTRTEMFL	5	debug	=	OFF

Macro Call Metadata (Parameter value):

domain	macName	paramn	param	equal	paramval
adae	%ADREL	1	indsn	=	adae
adae	%ADREL	2	outdsn	=	adae
adae	%ADREL	3	keepvars	=	AERELN RELGR1 RELGR1N
adae	%ADREL	4	debug	=	OFF
adae	%ADTRTEMFL	1	indsn	=	adae
adae	%ADTRTEMFL	2	invar	=	&dayvar
adae	%ADTRTEMFL	3	adrule	=	VERSION1
adae	%ADTRTEMFL	4	outdsn	=	adae
adae	%ADTRTEMFL	5	debug	=	OFF

Macro Level Metadata

Example 1 (cont'd)

Autogenerated ADAE.sas:

Auto-Created

```

/*****
* Short Description: Macro to create Treatment Emergent Analysis Flag
* Input : input dataset indsn,d_in.ae
* Output : input dataset in INDSN with TRTEMFL variable added to create the output dataset in OUTDSN
*****/
%ADTRTEMFL(indsn = _09ADADURN,
          invar=&dayvar,
          adrule = VERSION1,
          outdsn = _10ADTRTEMFL,
          debug=OFF);

/*****
* Short Description: Macro to create Causality & Pooled Causality Group variables
* Input: input dataset is indsn,d_in.ae
* Output: Input dataset in INDSN with AERELN, RELGR1, RELGR1N added to output dataset in OUTDSN  Depending
*****/
%ADREL(indsn = _10ADTRTEMFL,
       outdsn = _11ADREL,
       keepvars=AERELN RELGR1 RELGR1N,
       debug=OFF);

```



C:\TEMP\
ADAE.sas



ADAEfmt.sas

Macro Level Metadata

Example 1 (cont'd)

Reference multiple previous studies and the standard library

```
Remarks          : The ADMacrodir, studyADPrograms, and domain parameters can
                   : have multiple separated by a pipe (|).
                   : For example, &path1 | &path2 for ADMacrodir and studyADPrograms.
                   : For example, ADAE | ADVS for domain.
Macro Parameters  : ADMacrodir - paths to reference macro folders
                   : studyADPrograms - paths to reference program folders
                   : metadir - path to your metadata that will be used to create you code
                   : domain - domains you want to create
                   : outDir - output directory path write out files

%uacADaMCode(ADMacrodir = &study1MacroPath | &study2MacroPath| &study3MacroPath| &standardMacroPath
             ,metadir   = &opath
             ,studyADPrograms = &study1ProgramPath | &study2ProgramPath| &study3ProgramPath| &standardProgramPath
             ,domain     = ADSL| ADAE |ADVS |ADLB
             ,outDir     = &opath);
```

Macro Level Metadata

Example 1 (cont'd)

%PowerMacroExplorer

Features

- ▶ Compare macro and program metadata from multiple studies to obtain a complete overview
- ▶ Easy to find the most commonly used macros and the study-specific macros
- ▶ Learn from the metadata and pick the best one for their own needs to apply in the %uacADaMCode macro

[illegible]

Macro Level Metadata

Example 1 (cont'd)

Benefits

- ▶ Using the standards and previous studies reference save a lot time.
 - ▶ Macros have already been tested and validated
 - ▶ Don't need to 're-invent the wheel'
 - ▶ Keep the code more consistent
 - ▶ Allows for macro governance across a disease area or therapeutic areas, ensure robust macro development and consistency

MACNAME	PARAM	PARAMVAL	G	I	L	M	M	M	M	M	M	M	M	M	M	M	M	M	M
%TOXGRDV4	@ birthdt		1			1	1	1	1	1	1			1	1	1	1	1	1
	@ by	STUDYID USUBJID BASETYPE LBCAT LBTESTCD LBMETHOD LBSPEC VISITNUM LBSPID LBGRPID LBDTC LBREFID lbseq																	1
		STUDYID USUBJID LBCAT LBTESTCD LBMETHOD LBSPEC VISITNUM LBSPID LBGRPID LBDTC LBREFID											1		1				
		STUDYID USUBJID LBCAT LBTESTCD LBMETHOD LBSPEC VISITNUM LBSPID LBGRPID LBDTC LBREFID													1				
		STUDYID USUBJID LBCAT LBTESTCD LBMETHOD LBSPEC VISITNUM LBTPPTNUM LBSPID LBGRPID ADT ATM LBREFID ANL01FL ANL02FL																	1
		STUDYID USUBJID LBCAT LBTESTCD LBMETHOD LBSPEC VISITNUM LBTPPTNUM LBSPID LBGRPID LBDTC LBREFID											1				1	1	1
		STUDYID USUBJID LBCAT LBTESTCD LBMETHOD LBSPEC VISITNUM LBTPPTNUM LBSPID LBGRPID LBDTC LBREFID LBSEQ														1	1		
	@ ctcterm	ctcterm	1			1	1	1	1	1	1				1	1	1	1	1
	@ fasting	no	1										1	1	1	1			
		yes														1			1

Example 2:

IF-THEN/ELSE logic
in Data Step Level

IF-THEN/ELSE logic in Data Step Level

Example 2

- ▶ Assume we have the following data step specification for IF-THEN/ELSE logic



Microsoft Excel
Worksheet

- ▶ We can call *%ifstatement* macro against this specification to generate the SAS program



M3_demo_r1_MRM2Code_3.sas

- ▶ The above macro call generates these IF-THEN/ELSE program logic



_05_IFDEMO.sas



_04_toxicityGrade.sas



_03_BOR.sas

IF-THEN/ELSE logic in Data Step Level

Example 2 (cont'd)

Table 1 - Standardized *ifVar_outVar* metadata structure used to autogenerate complex IF-THEN/ELSE logic

groupVar	ifVar1	ifVar2	...	ifVarN	_	outVar1	outVar2	...	outVarN
xxx	xxx	xxx	xxx	xxx		xxx	xxx	xxx	xxx
xxx	xxx	xxx	xxx	xxx		xxx	xxx	xxx	xxx
xxx	xxx	xxx	xxx	xxx		xxx	xxx	xxx	xxx

Table 2 - *ifVar_outVar* metadata values populated

groupVar	ifVar1	ifVar2	_	outVar1	outVar2
1	varA	'valueA'		'valueX'	'valueY'
1	varA	'valueB'		varX	varY
2	#var1>var2	'valueC'		#expressionX	#expressionY

%ifStatement macro call



```
if ifVar1=varA then do;
  if ifVar2='valueA' then do; outVar1='valueX'; outVar2='valueY';end;
  else if ifVar2='valueB' then do; outVar1=varX; outVar2=varY;end;
end;

if var1>var2 and ifVar2='valueC' then do; expressionX; expressionY;end;
```

IF-THEN/ELSE logic in Data Step Level

Example 2 (cont'd)

Lab Toxicity Grade Derivation

1	PARAMCAT	PARAM	LBTESTCD	UNIT	AGELOD	AGEHID	g1lo	g1hi	g2lo	g2hi	g3lo	g3hi	g4lo	g4hi
2	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	1	7	130<=	<=140	120<=	<130	-9999<	<120	NA	NA
3	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	8	21	120<=	<=130	100<=	<120	-9999<	<100	NA	NA
4	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	22	35	95<=	<=105	80<=	<95	-9999<	<80	NA	NA
5	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	36	60	85<=	<=94	70<=	<84	-9999<	<70	NA	NA
6	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	61	90	90<=	<=99	70<=	<90	-9999<	<70	NA	NA
7	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	91	730	90<=	<=99	70<=	<90	-9999<	<70	NA	NA
8	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	731	5840	100<=	<=109	70<=	<100	-9999<	<70	NA	NA
9	HEMATOLOGY	Neutrophils (x10E9/L)	NEUT	x10E9/L	1	1	5<=	<=7	3<=	<5	1.5<=	<3	-9999<	<1.5
10	HEMATOLOGY	Neutrophils (x10E9/L)	NEUT	x10E9/L	2	6	1.75<=	<=2.5	1.25<=	<1.75	0.75<=	<1.25	-9999<	<0.75
11	HEMATOLOGY	Neutrophils (x10E9/L)	NEUT	x10E9/L	7	60	1.2<=	<=1.8	0.9<=	<1.2	0.5<=	<0.9	-9999<	<0.5
12	HEMATOLOGY	Neutrophils (x10E9/L)	NEUT	x10E9/L	61	90	0.75<=	<=1.2	0.4<=	<0.75	0.25<=	<0.4	-9999<	<0.25
13	HEMATOLOGY	Neutrophils (x10E9/L)	NEUT	x10E9/L	91	5840	0.75<=	<=1.2	0.4<=	<0.75	0.25<=	<0.4	-9999<	<0.25
14	HEMATOLOGY	Platelets (x10E9/L)	PLAT	x10E9/L	1	5840	NA	NA	50<=	<=75	<=25	<50	-9999<	<25
15	CHEMISTRY	Creatinine (umol/L)	CREAT	umol/L	1	6	1*88.4<=	<=1.7*88.4	1.7*88.4<	<=2.4*88.4	2.4*88.4<	<=3*88.4	3*88.4<	<9999
16	CHEMISTRY	Creatinine (umol/L)	CREAT	umol/L	7	60	0.5*88.4<=	<=0.9*88.4	0.9*88.4<	<=1.4*88.4	1.4*88.4<	<=2*88.4	2*88.4<	<9999

```

array glo{*} g1lo g2lo g3lo g4lo;
array ghi{*} g1hi g2hi g3hi g4hi;
do i = 1 to dim(glo);
    ageIf = cats('#',agelod,'<=age<=', agehid);
    avalIf = cats('#',glo[i], 'aval', ghi[i]);
    _ = '';
    toxgrade = i;
    if (glo[i] ne 'NA' or ghi[i] ne 'NA') then output;
end;

```

Intermediate Data

	PARAMCAT	PARAM	LBTESTCD	UNIT	ageIf	avalIf	_	toxgrade
1	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	#1<=age<=7	#130<=aval<=140		1
2	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	#1<=age<=7	#120<=aval<130		2
3	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	#1<=age<=7	#-9999<aval<120		3
4	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	#8<=age<=21	#120<=aval<=130		1
5	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	#8<=age<=21	#100<=aval<120		2
6	HEMATOLOGY	Hemoglobin (g/L)	HGB	g/L	#8<=age<=21	#-9999<aval<100		3

%ifStatement macro call

Auto-Created

```

if PARAMCAT='HEMATOLOGY' then do;
    if PARAM='Hemoglobin (g/L)' then do;
        if LBTESTCD='HGB' then do;
            if UNIT='g/L' then do;
                if 1<=age<=7 then do;
                    if 130<=aval<=140 then toxgrade=1;
                    else if 120<=aval<130 then toxgrade=2;
                    else if -9999<aval<120 then toxgrade=3;
                end;
            else if 8<=age<=21 then do;
                /*...*/
            end;
            /*...*/
            else if 731<=age<=5840 then do;
                if 100<=aval<=109 then toxgrade=1;
                else if 70<=aval<100 then toxgrade=2;
                else if -9999<aval<70 then toxgrade=3;
            end;
        end;
    end;
else if PARAM='Neutrophils (x10E9/L)' then do;
    if LBTESTCD='NEUT' then do;
        if UNIT='x10E9/L' then do;
            /*...*/
        end;
    end;
else if PARAM='Platelets (x10E9/L)' then do;
    /*...*/
end;
else if PARAMCAT='CHEMISTRY' then do;
    /*...*/
end;

```

IF-THEN/ELSE logic in Data Step Level

Example 2 (cont'd)

Best Overall Response (RECIST criteria)

Overall response at 1st time point (X ₁)	Overall response at 2nd time point (X ₂)	Overall response at 3rd time point (X ₃)	Best overall Response
CR	Unknown	Unknown	Final: SD if X ₁ ≥ XX otherwise, NE Interim: If no additional disease assessments are expected* then SD if X ₁ ≥ XX otherwise, NE else uCR
	CR with X ₂ - X ₁ ≥ YY	Any	CR
	CR with X ₂ - X ₁ < YY	Unknown	Final: SD if X ₂ ≥ XX otherwise, NE Interim: If no additional disease assessments are expected* then SD if X ₂ ≥ XX otherwise, NE else uCR
		PR	Flag as a data issue with note: cannot be PR after CR. SD if X ₂ ≥ XX otherwise, PD
		SD	Flag as a data issue with note: cannot be SD after CR. SD if X ₂ ≥ XX otherwise, PD
		PD	SD if X ₂ ≥ XX otherwise, PD
	PR	Any	Flag as a data issue with note: cannot be PR after CR. SD if X ₁ ≥ XX, otherwise PD
	SD	Any	Flag as a data issue with note: cannot be SD after CR. SD if X ₁ ≥ XX, otherwise PD
	PD	Any	SD if X ₁ ≥ XX, otherwise PD
PR	CR	CR	If X ₂ - X ₁ ≥ YY, CR; else it will be changed to leading (PR, CR) sequence due to data Step 10
	Unknown	Unknown	Final: SD if X ₁ ≥ XX otherwise, NE Interim: If no additional disease

Auto-Created

```

if RS1='CR' then do;
  if MISSING(RS2) and MISSING(RS3) then BOR=IFC(X1 >= XX,"SD","NE");
  else if RS2='CR' then do;
    if X2 - X1 >= YY then BOR="CR";
    else if X2 - X1 < YY then do;
      if MISSING(RS3) then BOR=IFC(X2 >= XX,"SD","NE");
      else if RS3='PR' then do; NOTE='Note: cannot be PR after CR.'; BOR=IFC(X2 >= XX,"SD","PD");end;
      else if RS3='SD' then do; NOTE='Note: cannot be SD after CR.'; BOR=IFC(X2 >= XX,"SD","PD");end;
      else if RS3='PD' then BOR=IFC(X2 >= XX,"SD","PD");
    else do; put 'ERROR: Unexpected error 1. ' USUBJID= RSDTC;BOR="";end;
  end;
end;
else if RS2='PR' then do; NOTE='Note: cannot be PR after CR.'; BOR=IFC(X1 >= XX,"SD","PD");end;
else if RS2='SD' then do; NOTE='Note: cannot be SD after CR.'; BOR=IFC(X1 >= XX,"SD","PD");end;
else if RS2='PD' then BOR=IFC(X1 >= XX,"SD","PD");
else do; put 'ERROR: Unexpected error 2. ' USUBJID= RSDTC;BOR="";end;
end;
else if RS1='PR' then do;
  if MISSING(RS2) and MISSING(RS3) then BOR=IFC(X1 >= XX,"SD","NE");
  else if RS2='CR' then do;
    if RS3='CR' then do;
      if X3 - X2 >= YY then BOR="CR";
      else do; put 'ERROR: Unexpected error 3. ' USUBJID= RSDTC;BOR="";end;
    end;
  end;
end;

```

	RS1	RS2	X2_X1	RS3	X3_X2	NOTE	ERROR	_JFC
1	CR	missing		missing				#BOR=IFC(X1 >= XX,"SD","NE")
2	CR	CR	#X2 - X1 >= YY					#BOR="CR"
3	CR	CR	#X2 - X1 < YY	missing				#BOR=IFC(X2 >= XX,"SD","NE")
4	CR	CR	#X2 - X1 < YY	PR		Note: cannot be ...		#BOR=IFC(X2 >= XX,"SD","PD")
5	CR	CR	#X2 - X1 < YY	SD		Note: cannot be ...		#BOR=IFC(X2 >= XX,"SD","PD")
6	CR	CR	#X2 - X1 < YY	PD				#BOR=IFC(X2 >= XX,"SD","PD")
7	CR	CR	#X2 - X1 < YY	#1			#put 'ERROR: U...	#BOR=""
8	CR	PR				Note: cannot be ...		#BOR=IFC(X1 >= XX,"SD","PD")
9	CR	SD				Note: cannot be ...		#BOR=IFC(X1 >= XX,"SD","PD")
10	CR	PD						#BOR=IFC(X1 >= XX,"SD","PD")
11	CR	#1					#put 'ERROR: U...	#BOR=""
12	PR	missing		missing				#BOR=IFC(X1 >= XX,"SD","NE")
13	PR	CR		CR	# X3 - X2 >= YY			#BOR="CR"
14	PR	CR		CR	#1		#put 'ERROR: U...	#BOR=""
15	PR	CR	#X2 - X1 >= YY	missing				#BOR="PR"
16	PR	CR	#X2 - X1 >= YY	PR		Note: cannot be ...		#BOR="PR"
17	PR	CR	#X2 - X1 >= YY	SD		Note: cannot be ...		#BOR="SD"

%ifStatement

IF-THEN/ELSE logic in Data Step Level

Example 2 (cont'd)

- ▶ In the previous examples, the macro *%ifStatement* serves as a compiler of the ifVar_outVar structure metadata specification.
- ▶ For each cell in the metadata specification, the macro distinguishes its function (value, variable, or expression, ifvar, outvar), and translates the metadata specification into the programming language.
- ▶ IF-THEN/ELSE logic is a powerful, logical construct that can cover many data derivations that may or may not be achieved using other SAS techniques. We can specify value, variable and even expressions in the metadata table.

Summary

Summary

- ▶ We should try to achieve as much automation as possible, although 100% of automation is not yet possible.
- ▶ **Benefits**
 - ▶ Reduce manual work, minimize human error which often leads to wasted time in both coding and debugging
 - ▶ Programmer focuses on design, complex analysis rather than tedious and manual coding
 - ▶ Able to compare with standard and historical algorithms by metadata comparison to enhance the standards and maintain consistency
 - ▶ Centralized control of coding instead of scattered pieces
 - ▶ Potentially require lower level of SAS skills for more routine coding
 - ▶ Further benefits with future database and interface like MDR (MetaData Repository)
- ▶ Concept can be extended to more complicated cases

Acknowledgements

We would like to thank the Janssen Autocode team Nick Masel, David Fan, Ilse Augustyns, Phyllis Wolf, Sumesh Kalappurakal, Taku Uryu, James Li, Sendil Ramabadran and Sandy Lei, for the contribution to the macros described in this paper and valuable comments.

Metadata-based Auto-Programming Process

Questions?

Lisa Lyons
Janssen R&D US
Phone: (609)-802-6283
e-mail: llyons3@its.jnj.com

Qian Zhao
Janssen R&D China.
Phone: 86-21-3337-7624
e-mail: qzhao1@its.jnj.com

Harry Chen
Janssen R&D China.
Phone: 86-21-3337-8892
e-mail: hchen104@its.jnj.com



Backup Slides

IF-THEN/ELSE logic in Data Step Level

Example 2 (cont'd)

Best Overall Response (RECIST criteria)

Overall response at 1st time point (X ₁)	Overall response at 2nd time point (X ₂)	Overall response at 3rd time point (X ₃)	Best overall Response
CR	Unknown	Unknown	Final: SD if X ₁ ≥ XX otherwise, NE Interim: If no additional disease assessments are expected* then SD if X ₁ ≥ XX otherwise, NE else uCR
	CR with X ₂ - X ₁ ≥ YY	Any	CR
	CR with X ₂ - X ₁ < YY	Unknown	Final: SD if X ₂ ≥ XX otherwise, NE Interim: If no additional disease assessments are expected* then SD if X ₂ ≥ XX otherwise, NE else uCR
		PR	Flag as a data issue with note: cannot be PR after CR. SD if X ₂ ≥ XX otherwise, PD
		SD	Flag as a data issue with note: cannot be SD after CR. SD if X ₂ ≥ XX otherwise, PD
		PD	SD if X ₂ ≥ XX otherwise, PD
	PR	Any	Flag as a data issue with note: cannot be PR after CR. SD if X ₁ ≥ XX, otherwise PD
	SD	Any	Flag as a data issue with note: cannot be SD after CR. SD if X ₁ ≥ XX, otherwise PD
	PD	Any	SD if X ₁ ≥ XX, otherwise PD
PR	CR	CR	If X ₃ - X ₂ ≥ YY, CR; else it will be changed to leading (PR, CR) sequence due to data Step 10
	Unknown	Unknown	Final: SD if X ₁ ≥ XX otherwise, NE Interim: If no additional disease

```

if RS1='CR' then do;
  if MISSING(RS2) and MISSING(RS3) then BOR=IFC(X1 >= XX,"SD","NE");
else if RS2='CR' then do;
  if X2 - X1 >= YY then BOR="CR";
else if X2 - X1 < YY then do;
  if MISSING(RS3) then BOR=IFC(X2 >= XX,"SD","NE");
  else if RS3='PR' then do; NOTE='Note: cannot be PR after CR.'; BOR=IFC(X2 >= XX,"SD","PD");end;
  else if RS3='SD' then do; NOTE='Note: cannot be SD after CR.'; BOR=IFC(X2 >= XX,"SD","PD");end;
  else if RS3='PD' then BOR=IFC(X2 >= XX,"SD","PD");
  else do; put 'ERROR: Unexpected error 1. ' USUBJID= RSDTC;BOR="";end;
end;
end;
else if RS2='PR' then do; NOTE='Note: cannot be PR after CR.'; BOR=IFC(X1 >= XX,"SD","PD");end;
else if RS2='SD' then do; NOTE='Note: cannot be SD after CR.'; BOR=IFC(X1 >= XX,"SD","PD");end;
else if RS2='PD' then BOR=IFC(X1 >= XX,"SD","PD");
else do; put 'ERROR: Unexpected error 2. ' USUBJID= RSDTC;BOR="";end;
end;
else if RS1='PR' then do;
  if MISSING(RS2) and MISSING(RS3) then BOR=IFC(X1 >= XX,"SD","NE");
else if RS2='CR' then do;
  if RS3='CR' then do;
    if X3 - X2 >= YY then BOR="CR";
    else do; put 'ERROR: Unexpected error 3. ' USUBJID= RSDTC;BOR="";end;
  end;
end;

```

Auto-Created

%ifStatement

Intermediate Data

	RS1	RS2	X2_X1	RS3	X3_X2	IFC	BOR1	BOR2	NOTE	ERROR
1	CR	Unknown		Unknown		#X1 >= XX	SD	NE		
2	CR	CR	#X2 - X1 >= YY				CR			
3	CR	CR	#X2 - X1 < YY	Unknown		#X2 >= XX	SD	NE		
4	CR	CR	#X2 - X1 < YY	PR		#X2 >= XX	SD	PD	Note: cannot be PR after CR.	
5	CR	CR	#X2 - X1 < YY	SD		#X2 >= XX	SD	PD	Note: cannot be SD after CR.	
6	CR	CR	#X2 - X1 < YY	PD		#X2 >= XX	SD	PD		
7	CR	CR	#X2 - X1 < YY	#other					#put 'ERROR: Unexpected error 1. ' USUBJID= RSDTC;	
8	CR	PR				#X1 >= XX	SD	PD	Note: cannot be PR after CR.	
9	CR	SD				#X1 >= XX	SD	PD	Note: cannot be SD after CR.	
10	CR	PD				#X1 >= XX	SD	PD		
11	CR	#other							#put 'ERROR: Unexpected error 2. ' USUBJID= RSDTC;	
12	PR	Unknown		Unknown		#X1 >= XX	SD	NE		
13	PR	CR	#X3 - X2 >= YY				CR			
14	PR	CR							#put 'ERROR: Unexpected error 3. ' USUBJID= RSDTC;	

	RS1	RS2	X2_X1	RS3	X3_X2	NOTE	ERROR	IFC
1	CR	missing		missing				#BOR=IFC(X1 >= XX,"SD","NE")
2	CR	CR	#X2 - X1 >= YY					#BOR="CR"
3	CR	CR	#X2 - X1 < YY	missing				#BOR=IFC(X2 >= XX,"SD","NE")
4	CR	CR	#X2 - X1 < YY	PR		Note: cannot be ...		#BOR=IFC(X2 >= XX,"SD","PD")
5	CR	CR	#X2 - X1 < YY	SD		Note: cannot be ...		#BOR=IFC(X2 >= XX,"SD","PD")
6	CR	CR	#X2 - X1 < YY	PD				#BOR=IFC(X2 >= XX,"SD","PD")
7	CR	CR	#X2 - X1 < YY	#1			#put 'ERROR: U...	#BOR=""
8	CR	PR				Note: cannot be ...		#BOR=IFC(X1 >= XX,"SD","PD")
9	CR	SD				Note: cannot be ...		#BOR=IFC(X1 >= XX,"SD","PD")
10	CR	PD						#BOR=IFC(X1 >= XX,"SD","PD")
11	CR	#1					#put 'ERROR: U...	#BOR=""
12	PR	missing		missing				#BOR=IFC(X1 >= XX,"SD","NE")
13	PR	CR		CR	#X3 - X2 >= YY			#BOR="CR"
14	PR	CR		CR	#1		#put 'ERROR: U...	#BOR=""
15	PR	CR	#X2 - X1 >= YY	missing				#BOR="PR"
16	PR	CR	#X2 - X1 >= YY	PR		Note: cannot be ...		#BOR="PR"
17	PR	CR	#X2 - X1 >= YY	PD				#BOR="PD"

Code to convert spec to the ifVar_outVar metadata structure for use with %ifStatement macro call

```

_BOR1 = quote(strip(BOR1));
_BOR2 = quote(strip(BOR2));
if IFC = '' then _IFC = cats('BOR=',_BOR1);
else _IFC = cats(tranwrd(IFC,'#',_BOR=IFC()),',',_BOR1,',',_BOR2,',');

```

Example 2

Benefits

- ▶ IF-THEN/ELSE logic is a powerful, logical construct that can cover many data derivations that may or may not be achieved using other SAS techniques. We can specify value, variable and even expressions in the metadata table.
- ▶ Maintain consistency between the specification and the program; therefore, changes need to be made only in one place. The code generated is easy to review and understand.
- ▶ Allow making changes in a central location and obtain better traceability from the source variable to the mapped variable.
- ▶ Enable a higher-level standardization due to reusability of the mapping specification template and the macro.
- ▶ The metadata may be saved in a general structured database to store most of the variable mapping algorithms to build a library