Paper SM03

VBA for CRT, Automating the Master Spec

Matt Metherell, PHASTAR, London, UK

ABSTRACT

Producing a CRT package, and the define.xml in particular, can often end up taking a back seat during study reporting despite its prominent role in the regulatory process. A common tool for creating the define.xml is Pinnacle 21 Community® (P21C), which, amongst other things extracts metadata from XPT files to create a master specification spreadsheet which it can then convert into a define.xml. In order for this define.xml to pass validation, certain tasks will need to be performed on the master specification spreadsheet and this paper looks at a solution using Visual Basic for Applications (VBA) to automate these tasks, and how these tasks vary with different versions of P21C.

INTRODUCTION

This paper aims to give a brief overview of the process for creating a define.xml with P21C (not to be confused with P21 Enterprise®, which is not covered by this paper) and to demonstrate a number of VBA macros to speed up this process.

Once P21 Community produces a spec template from the XPT files, we must fill it in to create what we shall refer to as the 'master spec' which P21C will convert into the define.xml. This paper looks at the differences in versions of P21C and offers guidance for filling in parts of this template automatically using VBA, which allows us to program the Excel tasks we wish to perform.

The macros in this paper can be utilized by pressing Alt+F11 to open the VBA editor, clicking on ThisWorkbook, pasting in the relevant macro and pressing Run.

As of October 2019, the oldest version available for download on the P21 website (https://www.pinnacle21.com/downloads) was version 2.1.3, so this paper looks at how different versions of the tool require different fixes.

MANDATORY VALUE

SDTM

P21C sets the variable USUBJID to Mandatory=No for all domains apart from DM, whereas the SDTM Implementation Guide (IG) v3.2 says USUBJID is a required variable for all domains apart from RELREC. All other variables are handled correctly, so we can use the following VBA macro to correct this:

```
Sub SDTM_Mandatory_Yes()

    'Select variable tab and filter to USUBJID when Dataset not equal to RELREC
    Sheets("Variables").Select
    If ActiveSheet.FilterMode = True Then Range("A:P").ShowAllData
    ActiveSheet.Range("A:P").AutoFilter Field:=3, Criteria1:="USUBJID"
    ActiveSheet.Range("A:P").AutoFilter Field:=2, Criteria1:="<>RELREC"
Operator:=xlAnd

    'Select Mandatory header and move down to first non-filtered row
    Range("I1").Select
    ActiveCell.Offset(1, 0).Select
        Do Until ActiveCell.EntireRow.Hidden = False
        ActiveCell.Offset(1, 0).Select
        Loop

    'Enter Yes, fill down and remove filtering
    ActiveCell.FormulaR1C1 = "Yes"
```

```
        Range(Selection, Selection.End(xlDown)).Select
        Selection.FillDown
        ActiveSheet.Range("A:P").AutoFilter

    End Sub
```

Changing "Variables" to "ValueLevel" in the first line will perform the same action on the ValueLevel tab due to the Dataset, Variable and Mandatory columns being in the same locations in both tabs in version 2.x specs. For version 3.x you will also need to "I1" to "J1" as the Mandatory column has moved one place to the right.

ADAM

Similarly, for an ADaM spec we also need to make some amendments to the Mandatory values. The ADaM IG has fewer required variables, but there is more scope to add extra variables in line with a Sponsor's guidelines. It is for this reason that the ADaM spec has more blank values. The following macro adds 'Yes' for CDISC-required variables and sets all others to 'No'. This can easily be adapted as necessary:

```
    Sub ADaM_MandatoryYes()

        'Select variable tab and filter to CDISC-required variables
        Sheets("Variables").Select
        If ActiveSheet.FilterMode = True Then Range("A:P").ShowAllData
        ActiveSheet.Range("A:P").AutoFilter Field:=3, Criteria1:=Array( _
            "PARAM", "PARAMCD", "STUDYID", "USUBJID", "CMTRT", "MHTERM", "CMSEQ",
"ECSEQ", "DVSEQ", "EXSEQ", "HOSEQ", "MHSEQ", "MLSEQ", "PRSEQ", "SITEID", "AGE",
"AGEU", "SEX", "RACE", "ARM", "TRT01P"), Operator:=xlFilterValues

        'Select Mandatory header and move down to first non-filtered row
        Range("I1").Select
        ActiveCell.Offset(1, 0).Select
            Do Until ActiveCell.EntireRow.Hidden = False
            ActiveCell.Offset(1, 0).Select
            Loop

        'Enter Yes and fill down
        ActiveCell.FormulaR1C1 = "Yes"
        Range(Selection, Selection.End(xlDown)).Select
        Selection.FillDown

        'Remove filters from sheet
        ActiveSheet.Range("A:P").AutoFilter Field:=3
        'Filter sheet to variable with missing Mandatory value
        ActiveSheet.Range("A:P").AutoFilter Field:=9, Criteria1:="="

        'Select Mandatory header and move down to first non-filtered row
        Range("I1").Select
        ActiveCell.Offset(1, 0).Select
            Do Until ActiveCell.EntireRow.Hidden = False
            ActiveCell.Offset(1, 0).Select
            Loop

        'Enter No and fill down
        ActiveCell.FormulaR1C1 = "No"
        ActiveCell.Offset(0, -8).Select ' move left by 8 cells
        Selection.End(xlDown).Select    ' move to final filtered row
        ActiveCell.Offset(0, 8).Select  ' move right by 8 cells
```


```
        Range(Selection, Selection.End(xlUp)).Select
        Selection.FillDown

        'Remove filters from sheet
        ActiveSheet.Range("A:P").AutoFilter

    End Sub
```

PREDECESSOR IN VALUELEVEL TAB (ADAM)

In version 2.1.3 and 2.2.0 of P21 Community, if the Predecessor column was populated in the ValueLevel tab, the corresponding value would not be visible when viewing in a browser despite it being present in the XML data. The browser view is a representation of the XML, with its layout dictated by the define2-0-0.css stylesheet provided by P21C.

Variable	Where Clause	Data Type	Length	Significant Digit	Format	Mandatory	Codelist	Origin	Pages	Method	Predecessor	Value Level Cor	Join Comment
AVALC	ADLB.PARAMCD.EQ.HYS	text	200			No		Derived		ADLB.AVALC.HYS1FL			
AVALC	ADLB.PARAMCD.EQ.HYS	text	200			No		Derived		ADLB.AVALC.HYS2FL			
AVALC	ADLB.PARAMCD.NOTIN	text	200			No		Predecessor			LB.LBSTRESC		

```
<ItemDef OID="IT.ADLB.AVALC.ADLB.PARAMCD.NOTIN.HYS1FL-c6f507ff"
         Name="AVALC.ADLB.PARAMCD.NOTIN.HYS1FL-c6f507ff"
         DataType="text"
         Length="200"
         SASFieldName="AVALC">
   <def:Origin Type="Predecessor">
      <Description>
         <TranslatedText xml:lang="en">LB.LBSTRESC</TranslatedText>
      </Description>
   </def:Origin>
</ItemDef>
```

Variable	Where	Type	Length / Display Format	Controlled Terms or Format	Origin	Derivation/Comment
AVALC	PARAMCD EQ HYS1FL (AST or ALT elevation)	text	200		Derived	For subjects in safety analysis set (ADSL.SAFFL=Y), equal to Y if patient has AST or ALT elevation (PARAMCD in ('L02101U1','L02102U1') and CRIT1FL='Y') between treatment start and treatment end +30 days (ADSL.TRTSDT<=ADT<ADSL.TRTEDT+30), otherwise equal to N.
AVALC	PARAMCD EQ HYS2FL (AT or Total Bilirubin (THL) elevation)	text	200		Derived	For subjects in safety analysis set (ADSL.SAFFL=Y), equal to Y if patient has (ALT > 3X ULN or AST > 3X ULN) and (TBL >2X ULN within 14 days on or after AT elevation): PARAMCD='L02110U1' and CRIT1FL='Y' and TRTSDT<=ADT<TRTEDT+30, within 14 days of AT elevation (see HYS1FL).
AVALC	PARAMCD NOTIN ("HYS1FL" (AST or ALT elevation) , "HYS2FL" (AT or Total Bilirubin (THL) elevation))	text	200		Predecessor	

In version 3.x this is not an issue, but for 2.x we need to add a ValueLevel comment to have the browser display the information correctly.

Order	Dataset	Variable	Where Clause	Data Type	Length	Significant Dig	Format	Mandator	Codelist	Origin	Pages	Method	Predecessor	Value Level Com
	ADLB	AVALC	ADLB.PARAMCD.EQ.HY	text	200			No		Derived		ADLB.AVALC.HYS1FL		
	ADLB	AVALC	ADLB.PARAMCD.EQ.HY	text	200			No		Derived		ADLB.AVALC.HYS2FL		
	ADLB	AVALC	ADLB.PARAMCD.NOTIN	text	200			No		Predecessor			LB.LBSTRESC	ADLB.LB.LBSTRESC

```
<ItemDef OID="IT.ADLB.AVALC.ADLB.PARAMCD.NOTIN.HYS1FL-c6f507ff"
         Name="AVALC.ADLB.PARAMCD.NOTIN.HYS1FL-c6f507ff"
         DataType="text"
         Length="200"
         SASFieldName="AVALC"
         def:CommentOID="COM.ADLB.LB.LBSTRESC">
   <def:Origin Type="Predecessor">
      <Description>
         <TranslatedText xml:lang="en">LB.LBSTRESC</TranslatedText>
      </Description>
   </def:Origin>
</ItemDef>
```

Variable	Where	Type	Length / Display Format	Controlled Terms or Format	Origin	Derivation/Comment
AVALC	PARAMCD EQ HYS1FL (AST or ALT elevation)	text	200		Derived	For subjects in safety analysis set (ADSL.SAFFL=Y), equal to Y if patient has AST or ALT elevation (PARAMCD in ('L02101U1','L02102U1') and CRIT1FL='Y') between treatment start and treatment end +30 days (ADSL.TRTSDT<=ADT<ADSL.TRTEDT+30), otherwise equal to N.
AVALC	PARAMCD EQ HYS2FL (AT or Total Bilirubin (THL) elevation)	text	200		Derived	For subjects in safety analysis set (ADSL.SAFFL=Y), equal to Y if patient has (ALT > 3X ULN or AST > 3X ULN) and (TBL >2X ULN within 14 days on or after AT elevation): PARAMCD='L02110U1' and CRIT1FL='Y' and TRTSDT<=ADT<TRTEDT+30, within 14 days of AT elevation (see HYS1FL).
AVALC	PARAMCD NOTIN ("HYS1FL" (AST or ALT elevation) , "HYS2FL" (AT or Total Bilirubin (THL) elevation))	text	200		Predecessor	LB.LBSTRESC

This is achieved for all Predecessors that don't currently have a comment with the following macro:

```
    Sub Predecessor_Comments()

        'Select ValueLevel tab, just predecessor vars with no comment
        Sheets("ValueLevel").Select
        ActiveSheet.Range("A:P").AutoFilter Field:=14, Criteria1:="<>"
```

```
    ActiveSheet.Range("A:P").AutoFilter Field:=15, Criteria1:="="

    'Move to first piece of data
    Range("N1").Select
    ActiveCell.Offset(1, 0).Select
        Do Until ActiveCell.EntireRow.Hidden = False
        ActiveCell.Offset(1, 0).Select
        Loop
    Range(Selection, Selection.End(xlDown)).Select 'select whole column
    Application.CutCopyMode = False
    Selection.Copy

    'Paste at bottom of Comments tab
    Sheets("Comments").Select
    Range("A1").Select
    Selection.End(xlDown).Select
    ActiveCell.Offset(1, 1).Select
    ActiveSheet.Paste

    'Create Comment references in ValueLevel tab
    Sheets("ValueLevel").Select
    Range("O1").Select
    ActiveCell.Offset(1, 0).Select
        Do Until ActiveCell.EntireRow.Hidden = False
        ActiveCell.Offset(1, 0).Select
        Loop

    'Enter formula in <Dataset>.<Predecessor> format
    ActiveCell.FormulaR1C1 = "=RC[-13]&"".""&RC[-1]"

    'Fill down and copy
    ActiveCell.Offset(0, -1).Select 'move left by 1 cell
    Selection.End(xlDown).Select
    ActiveCell.Offset(0, 1).Select 'move right by 1 cell
    Range(Selection, Selection.End(xlUp)).Select 'select column
    Selection.FillDown
    Selection.Copy

    'Paste references into Comment tab
    Sheets("Comments").Select
    Range("A1").Select
    Selection.End(xlDown).Select
    ActiveCell.Offset(1, 0).Select 'move down 1
    Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, _
    SkipBlanks:=False, Transpose:=False

    'Return to ValueLevel and paste comment references as values
    Sheets("ValueLevel").Select
    ActiveSheet.Range ("A:P").AutoFilter
    Columns("O:O").Select
    Application.CutCopyMode = False
    Selection.Copy
    Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, _
    SkipBlanks:=False, Transpose:=False

End Sub
```

WHERE CLAUSES IN P21 COMMUNITY

The ability to display ValueLevel Predecessor values may make version 3 seem appealing, but before version 3.0.2 there was an insurmountable issue that meant ValueLevel entries displayed all derivations, comments, and predecessors for all ValueLevel entries that used the same where clause. In this example, there are four entries in the final column because the where clause was repeated for each of the variables: AVALC, PARAM, PARAMREP, and DTYPE:

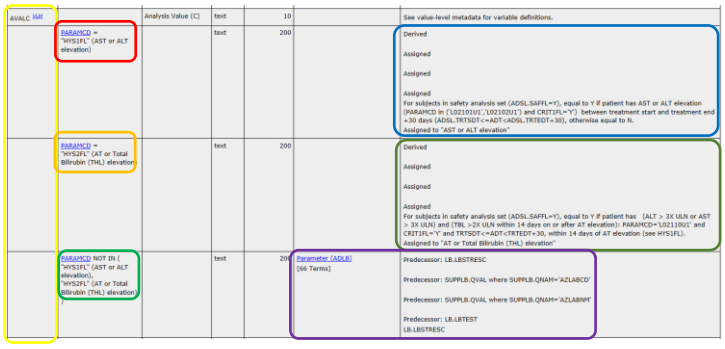
PARAM VLM		Parameter	text	Analysis Parameter	55	Parameter (ADLB) [33 Terms]	See value-level metadata for variable definitions.
	PARAMCD = "HYS1FL" (AST or ALT elevation)		text		200		Derived Assigned Assigned Assigned
	PARAMCD = "HYS2FL" (AT or Total Bilirubin (THL) elevation)		text		200		Derived Assigned Assigned Assigned
	PARAMCD NOT IN ("HYS1FL" (AST or ALT elevation), "HYS2FL" (AT or Total Bilirubin (THL) elevation))		text		200	Parameter (ADLB) [66 Terms]	Predecessor: LB.LBSTRESC Predecessor: SUPPLB.QVAL where SUPPLB.QNAM='AZLABCD' Predecessor: SUPPLB.QVAL where SUPPLB.QNAM='AZLABNM' Predecessor: LB.LBTEST

In version 3.0.1 of P21C, a ValueLevel item was defined by the ValueLevel dataset and the where clause definition, but not by the variable to which the ValueLevel applied. Specifically, the ItemOID was of the form:

IT.[VL.dataset].[WC.dataset].[WC.variable].[WC.comparator].[WC.value]

Therefore the ValueLevel item was only unique if the where clause was unique within the dataset. If multiple variables in a dataset had the same where clause, this resulted in the define.xml displaying all sources and codelists associated with the where clause. In the following example, four different variables used the same where clauses to split their definitions, and the Origins from all variables were combined in the browser view.

Dataset	Variable	Where Clause	Data Type	Length	Codelist	Origin	Method	Predecessor	Comment
ADLB	AVALC	ADLB.PARAMCD.EQ.HYS1FL	text	200		Derived	ADLB.AVALC.HYS1FL		
ADLB	AVALC	ADLB.PARAMCD.EQ.HYS2FL	text	200		Derived	ADLB.AVALC.HYS2FL		
ADLB	AVALC	ADLB.PARAMCD.NOTIN.HYS1FL,HYS2FL	text	200		Predecessor		LB.LBSTRESC	ADLB.LB.LBSTRESC
ADLB	PARAMCD	ADLB.PARAMCD.EQ.HYS1FL	text	8		Assigned			ADLB.PARAMCD.HYS1FL
ADLB	PARAMCD	ADLB.PARAMCD.EQ.HYS2FL	text	8		Assigned			ADLB.PARAMCD.HYS2FL
ADLB	PARAMCD	ADLB.PARAMCD.NOTIN.HYS1FL,HYS2FL	text	8	ADLB.PARAMCD	Predecessor		SUPPLB.QVAL wh	ADLB.SUPPLB.QVAL where SUPPLB.QNAM='AZLABCD'
ADLB	PARAM	ADLB.PARAMCD.EQ.HYS1FL	text	200		Assigned			ADLB.HY1.PARAMREP
ADLB	PARAM	ADLB.PARAMCD.EQ.HYS2FL	text	200		Assigned			ADLB.HY2PARAMREP
ADLB	PARAM	ADLB.PARAMCD.NOTIN.HYS1FL,HYS2FL	text	200	ADLB.PARAM	Predecessor		SUPPLB.QVAL wh	ADLB.SUPPLB.QVAL where SUPPLB.QNAM='AZLABNM'
ADLB	PARAMREP	ADLB.PARAMCD.EQ.HYS1FL	text	40		Assigned			ADLB.HY1.PARAMREP
ADLB	PARAMREP	ADLB.PARAMCD.EQ.HYS2FL	text	40		Assigned			ADLB.HY2PARAMREP
ADLB	PARAMREP	ADLB.PARAMCD.NOTIN.HYS1FL,HYS2FL	text	40		Predecessor		LB.LBTEST	ADLB.LB.LBTEST



In version 2 of P21 Community Edition, the where clause definition was used in conjunction with the ValueLevel dataset **and variable name** to create a unique ValueLevel ID of the form:

IT.[VL.dataset].**[VL.variable]**.[WC.dataset].[WC.variable].[WC.comparator].[WC.value]

This uniqueness meant we didn't see repeated sources in version 2.x. P21 have fixed this by reintroducing the ValueLevel variable name into the ItemOID for version 3.0.2.

COMPOSITE WHERE CLAUSES

We can combine multiple where clauses with an AND operator if we give them all the same ID on the WhereClauses tab. This is useful for separating complex derivations into constituent parts, for example by isolating specific parameters at specific timepoints.

WHERE CLAUSES IN P21 COMMUNITY VERSION 2.X

To achieve this, v2.x concatenated all the where clauses to create the ItemOID and WhereClauseOID.

ID	Dataset	Variable	Comparator	Value
ADSK.DTYPE.2	ADSK	PARAMCD	EQ	K
ADSK.DTYPE.2	ADSK	AVISITN	IN	4,7,9,10,11,12,13,14,15
ADSK.DTYPE.2	ADSK	ATPTN	EQ	PRE-DIALYSIS

```
<ItemRef ItemOID="IT.ADSK.DTYPE.ADSK.PARAMCD.EQ.K.ADSK.AVISITN.IN.4.ADSK.ATPTN.EQ.PRE-DIALYSIS-1daf034b"
         OrderNumber="2"
         Mandatory="No"
         MethodOID="MT.ADSK.DTYPE.ADJISTAT">
   <def:WhereClauseRef WhereClauseOID="WC.ADSK.PARAMCD.EQ.K.ADSK.AVISITN.IN.4.ADSK.ATPTN.EQ.PRE-DIALYSIS-1daf034b"/>
</ItemRef>
```

As the WhereClauseOID was made up purely of the dataset, variable, comparator, and values, when multiple where clauses have the same definition, they would all have the same WhereClauseOID. This meant that we could not have multiple instances of the same where clause in the master spec, otherwise they would all appear as duplicates each time one of them was referenced. For example, if both STUDYID and USUBJID used the same where clauses to split their derivations, we may see the following:

Dataset	Variable	Where Clause
ADSK	STUDYID	ADSK.STUDYID.1
ADSK	STUDYID	ADSK.STUDYID.2
ADSK	USUBJID	ADSK.USUBJID.1
ADSK	USUBJID	ADSK.USUBJID.2

dy | Datasets | Variables | **ValueLevel** | WhereClause

ID	Dataset	Variable	Comparator	Value
ADSK.STUDYID.1	ADSK	PARAMCD	EQ	DIALKC
ADSK.STUDYID.2	ADSK	PARAMCD	NE	DIALKC
ADSK.USUBJID.1	ADSK	PARAMCD	EQ	DIALKC
ADSK.USUBJID.2	ADSK	PARAMCD	NE	DIALKC

Study | Datasets | Variables | ValueLevel | **WhereClauses** | Cc ...

```
<def:WhereClauseDef OID="WC.ADSK.PARAMCD.EQ.DIALKC.02a44fce">
   <RangeCheck SoftHard="Soft" def:ItemOID="IT.ADSK.PARAMCD" Comparator="EQ">
      <CheckValue>DIALKC</CheckValue>
   </RangeCheck>
</def:WhereClauseDef>
<def:WhereClauseDef OID="WC.ADSK.PARAMCD.NE.DIALKC.cc537c8c">
   <RangeCheck SoftHard="Soft" def:ItemOID="IT.ADSK.PARAMCD" Comparator="NE">
      <CheckValue>DIALKC</CheckValue>
   </RangeCheck>
</def:WhereClauseDef>
<def:WhereClauseDef OID="WC.ADSK.PARAMCD.EQ.DIALKC.02a44fce">
   <RangeCheck SoftHard="Soft" def:ItemOID="IT.ADSK.PARAMCD" Comparator="EQ">
      <CheckValue>DIALKC</CheckValue>
   </RangeCheck>
</def:WhereClauseDef>
<def:WhereClauseDef OID="WC.ADSK.PARAMCD.NE.DIALKC.cc537c8c">
   <RangeCheck SoftHard="Soft" def:ItemOID="IT.ADSK.PARAMCD" Comparator="NE">
      <CheckValue>DIALKC</CheckValue>
   </RangeCheck>
</def:WhereClauseDef>
```

Parameter Value List - ADSK [STUDYID]

Variable	Where	Type	Length / Display Format	Controlled Terms or Format	Origin	Derivation/Comment
STUDYID	PARAMCD EQ DIALKC (Dialysate Potassium Concentration (mmol/L)) PARAMCD EQ DIALKC (Dialysate Potassium Concentration (mmol/L))	text	11		Predecessor	FA.STUDYID
STUDYID	PARAMCD NE DIALKC PARAMCD NE DIALKC	text	11		Predecessor	LB.STUDYID

Parameter Value List - ADSK [USUBJID]

Variable	Where	Type	Length / Display Format	Controlled Terms or Format	Origin	Derivation/Comment
USUBJID	PARAMCD EQ DIALKC (Dialysate Potassium Concentration (mmol/L)) PARAMCD EQ DIALKC (Dialysate Potassium Concentration (mmol/L))	text	30		Predecessor	FA.USUBJID
USUBJID	PARAMCD NE DIALKC PARAMCD NE DIALKC	text	30		Predecessor	LB.USUBJID

To combat this, we must de-duplicate the WhereClauses tab.

If we set the where clause ID to be equal to [dataset].[variable].[comparator].[value], then we can use the same ID on both the ValueLevel and WhereClauses tabs without having to change anything on the ValueLevel tab when de-duplicating. However, when we de-duplicate the WhereClauses tab, we need to avoid removing any of the rows we need for building a composite where clause, so must de-duplicate only when all columns are equal.

While you will most likely create your IDs manually, the following macro will auto-populate any empty ones in the Where Clauses tab with an ID made up of the dataset, variable, comparator, and value. It recognises composite where clauses by assuming that these were manually populated. It will then de-duplicate the sheet, but the user will need to manually copy these into the ValueLevel tab.

```
Sub WhereClauseIDs()

    'Select WhereClauses tab and remove any filters and filter to those without an ID
    Sheets("WhereClauses").Select
    If ActiveSheet.FilterMode = True Then Range("A:P").AutoFilter
    ActiveSheet.Range("A:P").AutoFilter Field:=1, Criteria1:="="

    'Insert formula into first empty cell
    Range("A1").Select
    ActiveCell.Offset(1, 0).Select
        Do Until ActiveCell.EntireRow.Hidden = False
        ActiveCell.Offset(1, 0).Select
        Loop
    ActiveCell.FormulaR1C1 = "=RC[1]&"".""&RC[2]&"".""&RC[3]&"".""&RC[4]"

    'Fill down whole column
    Range("B2").Select
    Selection.End(xlDown).Select
    ActiveCell.Offset(0, -1).Select 'move left by 1 cell
    Range(Selection, Selection.End(xlUp)).Select
    Selection.FillDown

    'Paste as values
    Range("A:P").AutoFilter
    Range("A2").Select
    Range(Selection, Selection.End(xlDown)).Select
    Selection.Copy
    Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
        :=False, Transpose:=False
    Application.CutCopyMode = False

    ' De-duplicate where clauses by ID
    ActiveSheet.Range("A:E").RemoveDuplicates Columns:=5, Header:=xlYes

End Sub
```

WHERE CLAUSES IN P21 COMMUNITY VERSION 3.X

In version 3.x, the ItemOID once again consists of the clauses concatenated together, but the WhereClauseOID uses the ID we define in the Where Clause tab.

ID	Dataset	Variable	Comparator	Value
ADSK.DTYPE.2	ADSK	PARAMCD	EQ	K
ADSK.DTYPE.2	ADSK	AVISITN	IN	4,7,9,10,11,12,13,14,15
ADSK.DTYPE.2	ADSK	ATPTN	EQ	PRE-DIALYSIS

```
<ItemRef ItemOID="IT.ADSK.ADSK.PARAMCD.EQ.ADSK.AVISITN.IN.ADSK.ATPTN.EQ.412ea804ba920215685288c1808435ea8abd8892"
         OrderNumber="2"
         Mandatory="No"
         MethodOID="MT.ADSK.DTYPE.ADJISTAT">
   <def:WhereClauseRef WhereClauseOID="WC.ADSK.DTYPE.2"/>
</ItemRef>
```

This means we don't need to worry about duplicates appearing if we ensure our WhereClause ID is unique.

Dataset	Variable	Where Clause
ADSK	STUDYID	ADSK.STUDYID.1
ADSK	STUDYID	ADSK.STUDYID.2
ADSK	USUBJID	ADSK.USUBJID.1
ADSK	USUBJID	ADSK.USUBJID.2

dy | Datasets | Variables | ValueLevel | WhereClause

ID	Dataset	Variable	Comparator	Value
ADSK.STUDYID.1	ADSK	PARAMCD	EQ	DIALKC
ADSK.STUDYID.2	ADSK	PARAMCD	NE	DIALKC
ADSK.USUBJID.1	ADSK	PARAMCD	EQ	DIALKC
ADSK.USUBJID.2	ADSK	PARAMCD	NE	DIALKC

Study | Datasets | Variables | ValueLevel | WhereClauses | Cc ...

```
<def:WhereClauseDef OID="WC.ADSK.STUDYID.1">
   <RangeCheck SoftHard="Soft" def:ItemOID="IT.ADSK.PARAMCD" Comparator="EQ">
      <CheckValue>DIALKC</CheckValue>
   </RangeCheck>
</def:WhereClauseDef>
<def:WhereClauseDef OID="WC.ADSK.STUDYID.2">
   <RangeCheck SoftHard="Soft" def:ItemOID="IT.ADSK.PARAMCD" Comparator="NE">
      <CheckValue>DIALKC</CheckValue>
   </RangeCheck>
</def:WhereClauseDef>
<def:WhereClauseDef OID="WC.ADSK.USUBJID.1">
   <RangeCheck SoftHard="Soft" def:ItemOID="IT.ADSK.PARAMCD" Comparator="EQ">
      <CheckValue>DIALKC</CheckValue>
   </RangeCheck>
</def:WhereClauseDef>
<def:WhereClauseDef OID="WC.ADSK.USUBJID.2">
   <RangeCheck SoftHard="Soft" def:ItemOID="IT.ADSK.PARAMCD" Comparator="NE">
      <CheckValue>DIALKC</CheckValue>
   </RangeCheck>
</def:WhereClauseDef>
```

Variable	Where Condition	Label / Description	Type	Length or Display Format	Controlled Terms or ISO Format	Origin / Source / Method / Comment
STUDYID VLM		Study Identifier	text	11		See value-level metadata for variable definitions.
	PARAMCD = "DIALKC" (Dialysate Potassium Concentration (mmol/L))		text	11		Predecessor: FA.STUDYID Predecessor: FA.USUBJID
	PARAMCD = "DIALKC" (Dialysate Potassium Concentration (mmol/L))		text	11		Predecessor: LB.STUDYID Predecessor: LB.USUBJID
USUBJID VLM		Unique Subject Identifier	text	20		See value-level metadata for variable definitions.
	PARAMCD = "DIALKC" (Dialysate Potassium Concentration (mmol/L))		text	11		Predecessor: FA.STUDYID Predecessor: FA.USUBJID
	PARAMCD = "DIALKC" (Dialysate Potassium Concentration (mmol/L))		text	11		Predecessor: LB.STUDYID Predecessor: LB.USUBJID

CONCLUSION

Versions 3.0.0 and 3.0.1 had an insurmountable issue that was fixed by version 3.0.2. Version 3 updates automatically if using the graphical user interface (GUI), so there should not be the problem of encountering this bug anymore. If using the command line interface (CLI), you can explicitly state which version to use and I would recommend avoiding 3.0.0 and 3.0.1. For version 3.0.2, the only macro required is the one that fixes the Mandatory values.

If you are required to use version 2.x, this necessitates all three macros above, as version 2.x incorrectly identifies Mandatory values, does not display ValueLevel Predecessor information, and needs the WhereClauses tab to be de-duplicated.

In either situation, a few simple macros can perform in seconds the fixes that previously took hours and allow for the simpler creation of a compliant define.xml. .

RECOMMENDED READING

The P21 forum is a great resource for info on the latest versions of P21 Community.
https://www.pinnacle21.com/forum/derived-displayed-multiple-times-origin-source-method-comment-column-definexml

CONTACT INFORMATION

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

Matt Metherell
PHASTAR
Unit 2A 2 Bollo Lane
London W4 5LE
Email: matthew.metherell@phastar.com