SS-034

Damn*t, The Define!

Julie Ann Hood and Sarah Angelo, Pinnacle 21

ABSTRACT

Validation rules for define.xml have been evolving for over a decade, and sponsors are expected to use these rules to validate prior to submission. However, even after years of improvements to rule descriptions in order to provide further guidance, validation rules can still prove difficult to interpret. Since the define.xml is generally not completed until the end of the study, many users may not be familiar with the validation rules or sometimes even the define.xml file itself. In this paper, we will cover some of the more obscure define validation rules to help provide clarity on what the rules are actually checking for, how to investigate the issue, and the necessary steps to correct the define.xml. This guidance will hopefully empower others in the industry to feel more optimistic about tackling define validation issues and refrain from yelling, "Damn*t, the define!".

INTRODUCTION: SETTING THE STAGE

According to the FDA Study Data Technical Conformance Guide, “The data definition file describes the metadata of the submitted electronic datasets, and is considered arguably the most important part of the electronic dataset submission for regulatory review.”[8] The data definition file, more commonly referred to as the “define.xml,” contains metadata used to describe the data sets, including all variables, possible values of variables, and origin of the data, as well as the standards and dictionaries used to create them.

Employed to assist with the automated validation process for eSubmissions, the define.xml file includes functional elements, such as hyperlinks to specific sections of the document or to external files, and is also quite a poetic display of metadata. Despite the added utility of the define.xml file during nonautomated, pre-submission review, there are several workflows in the industry where this document is not drafted until the study is nearly completed. We’ve all been there: after the trepidatious and sometimes arduous task of creating the define.xml, it comes time to validate the fruits of your labors - usually when it’s down to the wire and you’re certain the apocalypse is only hours away.

You muster your courage (maybe even in liquid form), browse for your define.xml file, and click that ominous blue button to Validate. At the same time, you start praying for a clean report, counting each and every second it takes for the progress bar to fill. Instead, out pops a report containing a variety of issues that might as well be written in Klingon at this point in your sleep-deprived state. You know you need to decipher and resolve all these issues before you can finally complete your submission package, much less get some shuteye. As frustration threatens to overwhelm you, you curse at your monitor glowing mockingly back at you, “Damn*t, the Define!” What to do?

In this article, we’ll break down some of the more complex Define-XML validation issues, including some common pitfalls to avoid, best practices for researching these issues, and steps to update both the Pinnacle 21 define.xml Excel spec as well as the XML code for the define.xml itself to resolve each issue. By providing both methods to remedy these issues, we’ll add to your ever-growing arsenal against noncompliance. After we reveal one of the most prevalent issues, with tips to avoid and resolve it, we’ll send you off with words of encouragement, a sparkling clean define.xml validation report well within your grasp. So take a deep breath, count to 10, deposit that dollar into the swear jar, and prepare to dive into Define-XML validation issues!

STANDARD AND DICTIONARY NAME AND VERSION

When preparing the define.xml file, the first, most critical details needed are the standard and dictionary names, and versions used to create the data sets. These values are used when validating the define.xml alone and when the data sets are validated with the define.xml file, so it’s important to ensure these values are accurate. If an incorrect version of the standard or dictionaries is chosen for validation, the validation report output may contain issues that aren’t relevant, or it may incorrectly exclude issues that require some sort of resolution. Can you feel the pressure building already?

STANDARD NAME AND VERSION

DD0021 & DD0022

When validating using a Pinnacle 21 platform such as Community, RuleIDs DD0021 and DD0022 are specific examples of rules related to standard and dictionary names and versions. If these issues are ever present within your validation report- STOP (do not pass Go, do not collect $200)! Dig out this paper and immediately check the following:

The message for DD0021 is “Invalid Standard Name value <value>” and will appear in the validation report when validating Define-XML v1.0 or v2.0 if the def:StandardName attribute is not populated with a value specified in the corresponding Define-XML Specification.

```
<MetaDataVersion OID="MDV..SDTM-IG.3.2" Name="Study null Data Definitions"
                 def:DefineVersion="2.0.0"
                 def:StandardName="SDTMIG"
                 def:StandardVersion="3.2">
```

Figure 1. Example Showing Incorrect Value For def:StandardName In XML Code For Define-XML v2.0

In the above code for Define-XML v2.0, the value of *SDTMIG* is present for def:StandardName, which triggers the DD0021 issue because the hyphen is not present in the value as required by the Define-XML v2.0 Specification.[2]

This seems fairly straightforward, so why are we mentioning this? If you’re a studious reader of the Define-XML Specification documents, you may have picked up on the following: the values allowable for def:StandardName change within each of the three different versions of these documents. In the specification for Define-XML v1.0, it’s noted that values of *CDISC SDTM*, *CDISC SEND*, and *CDISC ADaM* should be used for this attribute.[1] However, in the specification for Define-XML v2.0, the values of *SDTM-IG*, *SEND-IG*, and *ADaM-IG* should be used instead.[2] Then, in the specification for Define-XML v2.1, another monkey wrench is thrown into the mix, as the specified values have been updated with the removal of the hyphen (*SDTMIG*, *SENDIG*, and *ADaMIG*), values are now determined by Define-XML Controlled Terminology within the Standard Name codelist, and the attributes def:StandardName and def:StandardVersion are no longer used in v2.1 because they’ve been replaced by the new elements, def:Standards and def:Standard, to allow for the inclusion of multiple standards.[3] Is your head spinning yet?

Suffice it to say, when dealing with different versions of Define-XML, make sure that the value corresponding with the standard name is accurate per that version’s Define-XML Specification (or Define-XML CT for v2.1), and that the correct attribute to report this data is in your XML code.

Similarly, the message for DD0022 is “Invalid Standard Version value <value> for <standard>” and will fire when validating Define-XML v1.0 or v2.0 if the value for def:StandardVersion does not correspond with a valid value for the standard that is specified in def:StandardName. For this attribute, the version of the Implementation Guide, and not the model version, should be used when documenting versions of SDTMIG, SENDIG, or ADaMIG. For example, if your data sets were created using ADaMIG v1.2, the value for def:StandardVersion should be *1.2* and not *2.1*, which is the value of the corresponding ADaM model.

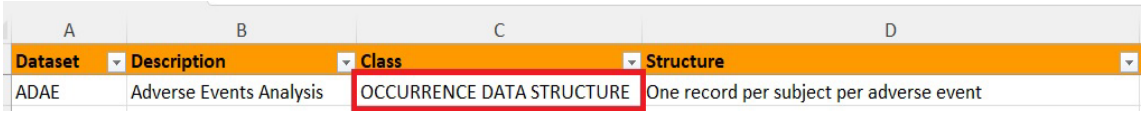
An example of an issue that has occurred related to the def:StandardVersion value is DD0055. The message for this issue states “Invalid Class value” and has appeared frequently in define.xml validation reports for ADaM data sets due to the Class differences between ADaMIG v1.0 and ADaMIG v1.1. Prior to January 1, 2022, the PMDA Data Standards Catalog did not include ADaM v1.1, which required studies that had previously been submitted to FDA using this standard to down-version to ADaM v1.0. In the define.xml, simply changing def:StandardVersion from *1.1* to *1.0* would cause this issue to appear for the ADAE dataset if the Class value remained unchanged as *OCCURRENCE DATA STRUCTURE* instead of *ADAM OTHER*.

Now that the PMDA will accept data sets using the ADaMIG v1.1 standard, this issue should occur with

much less frequency. However, in cases where down-versioning to ADaMIG v1.0 is still needed or when this issue is present in your validation report, refer to these steps to resolve this issue:

To fix this issue in the Pinnacle 21 define.xml Excel spec:

1. Navigate to the Datasets tab and change the “Class” of the specified data set to a value appropriate for the version of ADaMIG being used. In this case, change the value to “ADAM OTHER”.

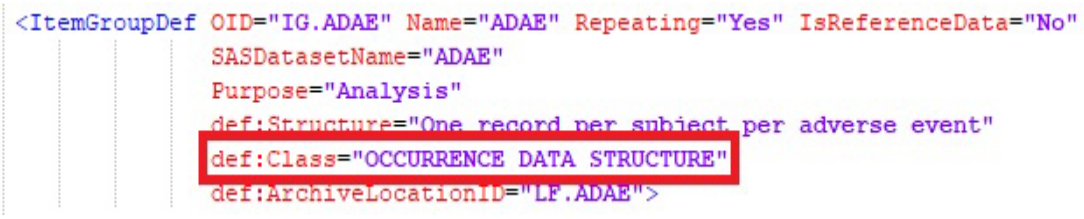


	A	B	C	D
	Dataset	Description	Class	Structure
	ADAE	Adverse Events Analysis	OCCURRENCE DATA STRUCTURE	One record per subject per adverse event

Figure 2. Pinnacle 21 define.xml Excel Spec Datasets Tab Showing Incorrect Class Value for ADaMIG v1.0

To fix in the XML code:

1. Search for the Dataset name reported in the validation report
2. In the code block starting with “<ItemGroupDef”, change the value for def:Class to a value appropriate for the version of ADaMIG being used. In this case, change the value to “ADAM OTHER”.



```
<ItemGroupDef OID="IG.ADAE" Name="ADAE" Repeating="Yes" IsReferenceData="No"
              SASDatasetName="ADAE"
              Purpose="Analysis"
              def:Structure="One record per subject per adverse event"
              def:Class="OCCURRENCE DATA STRUCTURE"
              def:ArchiveLocationID="LF.ADAE">
```

Figure 3. XML Code Showing Where to Update Class Value by Manually Editing define.xml File

DICTIONARY NAME AND VERSION

DD0025 & DD0112

The define.xml should also include any dictionaries, along with their respective versions, used when mapping data. Of particular interest is the MedDRA dictionary for two distinct agency-related reasons: The FDA’s DataFit application uses the define.xml file to obtain the version of MedDRA used for the study. An incorrect version value for this dictionary will trigger issue DD0025, reporting an “Invalid MedDRA Version <version>” and will lead to downstream validation issues. However, when this issue fires after validating using the PMDA engine, it has a Severity of Reject which is required to be fixed prior to submission.

This dictionary is so important, it has its own section in the FDA’s Technical Conformance Guide (section 6.3.1).[8] Omitting MedDRA as a Dictionary from the define.xml will score you a DD0112 error: “Missing MedDRA definition”.

Obviously, to resolve the DD0112 issue, MedDRA should be reported as a dictionary within the define.xml file. See Figure 4 for best practices on how this should appear in the Pinnacle 21 define.xml Excel spec or see Figure 5 for what XML code should be present in the define.xml file (except, be sure to use a valid version value!). To resolve or avoid a DD0025 issue, follow these steps:

In the Pinnacle 21 define.xml Excel spec:

1. The best practice is to make sure that the row reporting the MedDRA dictionary looks like the following, where “Dictionary” contains “MedDRA” and ensure the version value ends in a decimal value of 0 or 1:

ID	Name	Data Type	Dictionary	Version
MedDRA	Medical Dictionary for Regulatory Activities	text	MedDRA	23.0

Figure 4. Example of Pinnacle 21 define.xml Excel Spec Reporting Correctly Formatted Value for Version of MedDRA Dictionary

2. If your MedDRA dictionary version ends in a decimal value of 0, add an apostrophe in the cell prior to the first number in the version value so the decimal of 0 will be correctly populated in the define.xml file.

In the XML code:

1. Open the define.xml with a text editor such as Notepad++ and search for “MedDRA”.
2. Ensure the Dictionary value is “MedDRA” and the Version value within the quotations ends in a decimal value of 0 or 1.

```
<CodeList OID="CL.MedDRA" Name="Medical Dictionary for Regulatory Activities"
          DataType="text">
  <ExternalCodeList Dictionary="MedDRA" Version="24.5"/>
</CodeList>
```

Figure 5. Example Showing Incorrect MedDRA Version in XML Code

CONTROLLED TERMINOLOGY

CDISC Controlled Terminology (CT) can be another source of distress when validating the define.xml. The use of Controlled Terminology has many benefits such as harmonizing data, assisting reviewers with consistencies, aiding in automatic review, and helping with cross-study comparative analysis. There are several validation rules for Define-XML to check that codelists, codelist codes, terms, and term codes are appropriate for the chosen terminology. For Define-XML v1.0 and v2.0, the version of Controlled Terminology used does not appear within the define.xml itself.[1, 2] Instead, the CT version is reported in the Reviewer’s Guide, then selected during validation configuration so that comparisons of codelists, codes, and terms can be made against the reported CT version. However, Define-XML v2.1 allows for multiple versions of Controlled Terminology, so the CT version should be specified within the def:StandardOID attribute for each codelist used.[3] No matter which version of Define-XML is used, always ensure the correct version of CDISC CT is selected during validation or included within the define.xml for an accurate validation report.

DD0031

A common issue occurring with the define.xml with respect to Controlled Terminology is DD0031: Missing NCI Code for Codelist <codelist>. DD0031 fires when the Alias Name attribute for the expected Codelist associated with a variable (per the standard) is not found. For example, the variable AEACN is associated with the ACN codelist, and the validator is set to recognize this per the SDTM standard. If the Codelist Code for the ACN codelist, C66767, is not present for the ACN codelist in the Define, this issue will appear in the Pinnacle 21 validation report.

Sometimes this occurs as the result of confusion with the CDISC CT and how to add terms to extend codelists that already exist within these documents. So how do you know whether a codelist is extensible or not? For that, we turn to the CDISC Controlled Terminology files themselves. Within each of the Controlled Terminology files published by NCI, column C, labeled Codelist Extensible (Yes/No), specifies whether terms can be added.

The first row of each codelist contains light-blue shaded cells with a value in the Codelist Extensible (Yes/No) column. Per the Description in the ReadMe tab of the CDISC CT files, related to the Codelist Extensible (Yes/No) column, if the value is “Yes,” then “new terms may be added to existing codelist values…as long as the terms are not duplicates or synonyms of existing terms.”[15] To further specify, on the same tab, for the CDISC Synonym(s) column of the CT, the Description states, “This identifies the applicable synonyms for the CDISC Submission Value in Column E. These terms should not be

submitted, but are included for collection and mapping purposes."[15]

Taking both descriptions into consideration, if a term is a synonym for a CDISC Submission Value, that term should *not* be added to the codelist, regardless of whether the data was captured using the synonym value, and the CDISC Submission Value should be used in its place within the data sets and the define.xml. There can be some hesitation to change values from the raw data when mapping to SDTM, but in cases of synonyms, it's especially important to update the SDTM variable value to display the CDISC Submission Value.

A quick aside to illustrate these concepts:

Imagine you're working on a study where cortisol is being collected in the unit *mcg/day*. This data maps to the LB domain, and LBORRESU, the variable for Original Units, is associated with the Unit codelist per the SDTMIG v3.3.[6] Being the proactive programmer you are, you'll want to double-check that "mcg/day" is a valid term within the Unit codelist, and when you filter this codelist on CDISC Submission Value *mcg/day* in the 2021-12-17 version of SDTM CT, no records are returned. You check whether the Unit codelist can be extended and see that in fact, it can.

"Ah ha!" you think. *"This means that I should add mcg/day as a term into the Unit codelist!"* Suddenly, your eureka moment starts to fade as memories of this paper begin to invade your thoughts, and you remember that you also need to check the CDISC Synonym(s) values for this codelist to ensure that *mcg/day* isn't already listed there.

Upon searching that field, you see that *mcg/day* is indeed listed as a synonym for the CDISC Submission Value *ug/day*. This means you must add the term *ug/day* to the Unit codelist in your define.xml and populate this value in the LBORRESU field for the cortisol results, even though the form is collecting the results using the unit *mcg/day*. Once you've added this term to your LB coding and define.xml, you can let that sweet rush of satisfaction sweep over you, knowing you've fought the good fight for CDISC compliance and won!

Code	Codelist Code	Codelist Extensible (Yes/No)	Codelist Name	CDISC Submission Value	CDISC Synonym(s)
C71620		Yes	Unit	UNIT	Unit
C71205	C71620		Unit	ug/day	mcg/day

Figure 6. SDTM Terminology File Showing Unit Codelist Is Extensible and *ug/day* as Submission Value, Having *mcg/day* as a CDISC Synonym Value[15]

Two more details to take note of are the fields for Code and Codelist Code within the CDISC Controlled Terminology file. Although the labels in the CDISC CT are simply specified as "Code" and "Codelist Code", the "Code", within the same row that the term is located in, is the value that is applicable to the term itself and should be present in the NCI Term Code field for that term within the Pinnacle 21 define.xml Excel spec. However, the "Codelist Code" references the code for the entire codelist in which the term resides and should be present in the NCI Codelist Code field for that term within the Pinnacle 21 define.xml Excel spec.

Code	Codelist Code	Codelist Extensible (Yes/No)	Codelist Name	CDISC Submission Value	CDISC Synonym(s)	CDISC Definition	NCI Preferred Term

Figure 7. CDISC CT Header Row Showing Code and Codelist Code Columns[15]

For more tips and detailed information on CDISC Controlled Terminology best practices, check out Pinnacle 21's very own YouTube star Sarah Angelo's webinar on how to harness the power of CT.[16] And now back to our regular validation programming…

If you determine that a new term should be added to an extensible codelist, remember to include the Codelist Code since that particular codelist already exists in the CDISC CT. However, even if a term is added to a *non*-extensible codelist, the Codelist Code still needs to be present for the new term.

For example, the variable AEACN is associated with the ACN codelist in the SDTMIG.[5, 6, 7] If the term

MULTIPLE is added to the ACN codelist, then the Codelist Code, C66767, will still need to be present for the term *MULTIPLE*. What should be missing, in this case, is the Code value for the term itself. For reference, please refer to SDTMIG v3.2, v3.3, or v3.4 in the section titled "Multiple Values for a Non-Result Qualifier Variable."

When might your study include the term *MULTIPLE* in the variable AEACN? One scenario is when multiple study treatments are given within the study. In this case, the action taken with each unique study treatment is reported in supplemental qualifiers and AEACN is mapped to *MULTIPLE* by default, which corresponds with the example given in the various SDTMIG versions.[5, 6, 7]

Another scenario hinges on the eCRF design. For example, if the eCRF contains checkbox items for the question *"What was the action taken with study treatment"*, and more than one answer is reported, this could also lead to *MULTIPLE* in AEACN with the individual responses populated in suppquals. These are both instances where it would be appropriate to add *MULTIPLE* as a term.

In other scenarios, it may be that a new sponsor-defined codelist has been created in error. Imagine the following: Your new colleague sees the variable AEACN associated with the ACN codelist but also notices that this codelist is non-extensible and starts considering how the term *MULTIPLE* could be added.

"I know!", they think. *"I'll just create a sponsor-defined codelist that's extensible and add all of the terms from the ACN codelist AND MULTIPLE! Bingo!"* So, they create a new, sponsor-defined codelist called Action Taken with Study Treatment Multi (ACNMULTI), add all the terms from the ACN codelist along with *MULTIPLE*, and associate it with AEACN.

Instead of solving the problem, an issue for DD0031 is added to the validation report, indicating they didn't add the Codelist Code for the Action Taken with Study Treatment codelist, despite associating the AEACN variable with their new codelist. In this case, the issue occurs because creating a new, sponsor-defined codelist to add a term to a non-extensible codelist was not the best solution.

What should happen in the four cases described above?

A simple solution is to add the term *MULTIPLE* to the ACN codelist. However, if some terms within the CDISC published ACN codelist are not available for use in the study, the ACN codelist should be subset and the term *MULTIPLE* should be added to this subset codelist.

A subset codelist contains some - but not all - values from the parent codelist. However, of utmost importance for this validation issue, the subset codelist also inherits the Codelist Code value from the parent codelist. Therefore, when subsetting the ACN codelist, the Codelist Code value of C66767 should be present as the Alias Name value for the new subset codelist. This will prevent the DD0031 issue from firing.

In the Pinnacle 21 validation report, for this issue, the Values cell displays the variable, the codelist associated with the variable in the define.xml, the Standard Codelist associated with the variable, and the Standard Codelist Code. In Figure 8, we can see that the value for the Define Codelist and Standard Codelist do not match.

Domain	Record	Count	Variables	Values	Pinnacle 21 ID	Message
DEFINE		1	Variable, Define Codelist, Standard Codelist, Standard Codelist Code	AEACN, Action Taken with Study Treatment Multi, Action Taken with Study Treatment, C66767	DD0031	Missing NCI Code for Codelist 'Action Taken with Study Treatment Multi'

Figure 8. Pinnacle 21 Validation Report Issue Details Displaying Values for RuleID DD0031

Because the Define Codelist and Standard Codelist values do not match, we must determine if the Codelist Code is simply missing in the define.xml for the Define Codelist specified in the validation report (which would indicate that a subset codelist was *meant* to be created but the Codelist Code was simply not added for this new codelist), or if the Standard Codelist could be used for the variable instead. To do this, you must verify what terms were available to be collected for this variable.

If it is determined that a subset codelist is appropriate, simply add the Codelist Code to the NCI Codelist Code field for all terms within the subset codelist in the Pinnacle 21 define.xml Excel spec. Instead, if it is

determined that the Standard Codelist should be used with the addition of the term *MULTIPLE*, do the following:

To resolve this specific issue in the Pinnacle 21 define.xml Excel spec:

1. In the Variables tab, search for the variable present for this issue in the validation report. In this example, the variable is AEACN.
2. Change the value present in the Codelist field for the variable in Step 1 to correspond with the Codelist ID value representing the Standard Codelist. In this example, the Codelist ID should be ACN, which represents the Action Taken with Study Treatment codelist.
3. In the Codelists tab, search for the Define Codelist value listed in the validation report and note the ID for this codelist. In this example, search for “Action Taken with Study Treatment Multi” and note the ID of ACNMULTI.
4. In the Variables and Value Level tabs, search for the ID in the Codelist column to ensure this codelist is not associated with any other variable or value level metadata:
    a. If this codelist IS NOT associated with any other variables, return to the Codelists tab and remove all entries for the Define Codelist value listed in the validation report. In this example, the codelist ACNMULTI is not associated with any other variables in the Variables or Value Level tabs. Remove all records where Name is “Action Taken with Study Treatment Multi”.
    b. If this codelist IS associated with any other variables or value level metadata, do NOT remove any entries from the Codelists tab.
5. Add records for all necessary terms within the Standard Codelist, specified by the validation report, to the Codelists tab. Include the Standard Codelist Code as the value for NCI Codelist Code and the NCI Term Code for any terms also present within the codelist in the CDISC CT. In this example, add a record for the terms *DOSE INCREASED, DOSE NOT CHANGED, DOSE RATE REDUCED, DOSE REDUCED, DRUG INTERRUPTED, DRUG WITHDRAWN, NOT APPLICABLE,* and *UNKNOWN* with the NCI Codelist Code C66767 as well as the appropriate NCI Term Code values. Add a record for the term *MULTIPLE* with the NCI Codelist Code C66767, but leave the NCI Term Code value null since this is not a term within the ACN codelist in CDISC CT.

To resolve this issue in the XML code:

1. Search the XML code for the variable present for this issue where the code block begins with “<ItemDef OID=” in the validation report.
2. Within the ItemDef code block, find the CodeListOID and note the value. Replace the value in quotations with the Codelist ID value representing the Standard Codelist. In this example, find the ItemDef code block that displays when searching for “AEACN”, as shown in Figure 9. The value for CodeListOID was previously listed as “CL.ACNMULTI” and should be updated to “CL.ACN”.

```
<ItemDef OID="IT.AE.AEACN" Name="AEACN" DataType="text" Length="1" SASFieldName="AEACN">
   <Description>
      <TranslatedText xml:lang="en">Action Taken with Study Treatment</TranslatedText>
   </Description>
   <CodeListRef CodeListOID="CL.ACN"/>
</ItemDef>
```

Figure 9. XML Code for AEACN After CodeListOID Value Updated

3. Search the XML code for any instances of the Codelist value noted in Step 2 to ensure the Codelist was not associated with any other variables or value level metadata. If the Codelist value is found within an ItemDef code block, this means the codelist is associated with another variable or value level metadata. In this example, search for “CL.ACNMULTI”.

a. If this codelist is NOT associated with any other variables, find the codeblock starting with "<CodeList OID=" as noted in Step 2. In this example, the codelist ACNMULTI is not associated with any other variables or value level metadata. Remove all entries for the CodeList OID="CL.ACNMULTI" as noted in Figure 10.

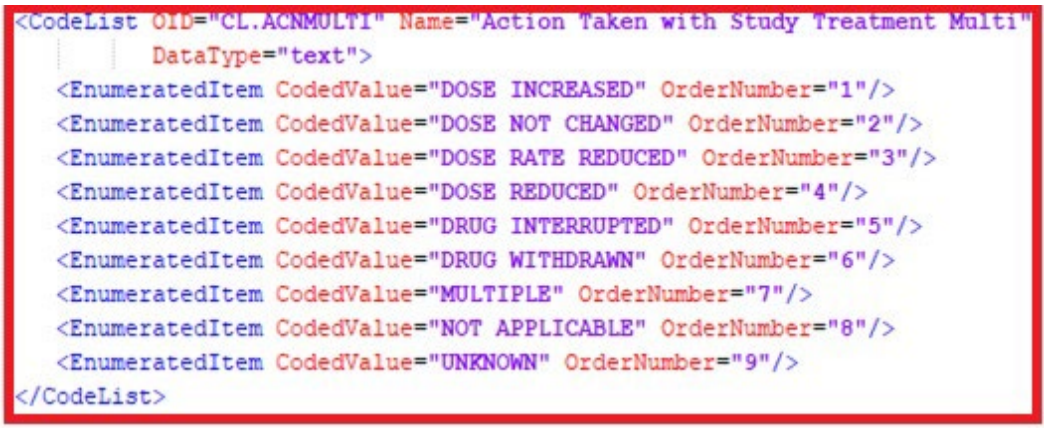


Figure 10. XML Code for Codelist=ACNMULTI Before Code Removed

b. If this codelist IS associated with any other variables or value level metadata, do NOT remove any entries from the Codelists section of the XML code.

4. Within the XML's Codelist section, listed in alphabetical order by codelist, add XML code for all necessary terms within the Standard Codelist specified by the validation report. Be sure to include the Code (for the term) as the Alias Name for any terms present within the codelist in the CDISC CT, as well as including the Standard Codelist Code as the Alias Name for the codelist.

   In this example, add XML code for the term *DOSE INCREASED, DOSE NOT CHANGED, DOSE RATE REDUCED, DOSE REDUCED, DRUG INTERRUPTED, DRUG WITHDRAWN, NOT APPLICABLE,* and *UNKNOWN* with the appropriate Alias Name values using the Code values from SDTM CT. Add XML code for the term *MULTIPLE,* added in alphabetical order, but do not include an Alias Name attribute for this term since this is not a term within the ACN codelist in CDISC CT. Add the Alias Name attribute for the Codelist Code as C66767 for the Codelist ACN. Please note that, while alphabetizing codelists is not a requirement, it is a best practice that may ease the review.

```
<CodeList OID="CL.ACN" Name="Action Taken with Study Treatment" DataType="text">
   <EnumeratedItem CodedValue="DOSE INCREASED" OrderNumber="1">
      <Alias Name="C49503" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <EnumeratedItem CodedValue="DOSE NOT CHANGED" OrderNumber="2">
      <Alias Name="C49504" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <EnumeratedItem CodedValue="DOSE RATE REDUCED" OrderNumber="3">
      <Alias Name="C150826" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <EnumeratedItem CodedValue="DOSE REDUCED" OrderNumber="4">
      <Alias Name="C49505" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <EnumeratedItem CodedValue="DRUG INTERRUPTED" OrderNumber="5">
      <Alias Name="C49501" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <EnumeratedItem CodedValue="DRUG WITHDRAWN" OrderNumber="6">
      <Alias Name="C49502" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <EnumeratedItem CodedValue="MULTIPLE" OrderNumber="7" def:ExtendedValue="Yes"/>
   <EnumeratedItem CodedValue="NOT APPLICABLE" OrderNumber="8">
      <Alias Name="C48660" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <EnumeratedItem CodedValue="UNKNOWN" OrderNumber="9">
      <Alias Name="C17998" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <Alias Name="C66767" Context="nci:ExtCodeID"/>
</CodeList>
```

Figure 11. XML Code Added for ACN Codelist, Including Extended Term *MULTIPLE*

DD0028

Another confusing rule based on Controlled Terminology is DD0028. DD0028 displays the message "Term/NCI Code mismatch in Codelist <codelist>" and occurs when the NCI Code specified for a term can be found in the codelist for the specific version of Controlled Terminology, but the term in the define.xml does not match the CDISC Submission value present in the CT.

As you might suspect, the most frequent cause of this issue is SDTMIG v3.2 PCORRESU and PCSTRESU's association with the UNIT codelist instead of the PKUNIT codelist.[5] For example, when results are reported in *ng/mL*, this term that exists within the PKUNIT codelist is often added into the UNIT codelist in the define.xml with the Code (for the term) C67306.

However, the term associated with Code C67306 in the UNIT codelist is *ug/L*, not *ng/mL*. If you filter the SDTM Terminology on NCI Code C67306, you'll find that this *one* code is associated with *two different CDISC Submission values within two separate codelists.* Luckily, this issue was fixed in SDTMIG v3.3 so that PCORRESU and PCSTRESU are now associated with the PKUNIT codelist.[6] Phew!

Issues for DD0028 also occur as a result of CDISC Submission Values changing between versions of the CDISC CT. For example, in the 2021-03-26 version of the SDTM Terminology, Code C177957 was associated with the CDISC Submission Value *2MCA*.[12] However, in the 2021-06-25 version of SDTM Terminology, the CDISC Submission Value was changed from *2MCA* to *MCA2*.[13] If the Define-XML v1.0 or v2.0 file includes the term *2MCA* with NCI Code C177957 and the SDTM CT is set to 2021-03-26 for the validation, then no DD0028 issue will be displayed. However, if the SDTM CT version is chosen as 2021-06-25, then DD0028 will fire for this issue.

In September of 2020, SDTM Terminology was released where the value for Vital Signs Test Name *Temperature* was overwritten as *TEMP* in the CDISC Submission Value.[10] Since VSTESTCD=TEMP and VSTEST=Temperature have been listed in the SDTM Terminology as paired values since April 2007 -

and because this wasn't a planned change - there continue to be a high number of unexpected DD0028 issues appearing in define.xml validation reports when the September 2020 SDTM CT is used.

Side note: An updated version of SDTM Terminology was released shortly thereafter in November 2020 to remedy this and two other issues found in the September 2020 release.[11]

DD0028 operates like a super-strict Language Arts teacher in that it checks spelling, capitalization, and character use. Remember that not all terms are created equal, even if they may relay the same message. For instance, *PH* is not the same as *pH* in terms of standard compliance.

When this issue appears in your report, verify which version of SDTM CT was used for the study and which version was used in the validation. Also, always ensure the correct version is reported in the corresponding Reviewer's Guide. If the same CT version was used in both the study and validation, look at the term by the Code and Codelist Code to ensure the CDISC Submission Value exactly matches the value in your define.xml file.

To research and resolve this issue, first verify which version of CT was used for the study and which version was used in the validation. If the CT version isn't the issue, note the Code for which this issue is firing, open the CDISC Controlled Terminology file, and search for the Define Term Code value from your validation report. Then, complete the following:

In the Pinnacle 21 define.xml Excel spec:

1. In the Codelists tab, filter the NCI Term Code column on the Define Term Code value from your validation report. In this example, search for C177957.
2. Compare the Define Term value from the validation report against the CDISC Submission Value reported in the CT file, checking for spelling, capitalization, and any characters, including carriage returns and spaces. In this example, note that the Define Term in the validation report is *2MCA*, but the CDISC Submission value is *MCA2*.
3. Update the Term value in the Pinnacle 21 define.xml Excel spec. In this example, change the Term *2MCA* to *MCA2*.

	A	B	C	D	E	F	G	H
	ID	Name	NCI Codelist Code	Data Type	Order	Term	NCI Term Code	Decoded Value
1	LBTESTCD	Laboratory Test Code	C65047	text	1	2MCA	C177957	

Figure 12. Pinnacle 21 define.xml Excel Spec Codelists Tab Showing Term Before Updated to *MCA2*

In the XML code:

1. Search for the Define Term Code value specified in the validation report for the code block containing the term reported for this issue in the validation report within quotation marks for the CodedValue, as in Figure 13.
2. Compare the value of the CodedValue against the CDISC Submission Value in the CT file, checking for spelling, capitalization, and any characters, including carriage returns and spaces.
3. Update the CodedValue within quotation marks to match the CDISC Submission Value in the CT file.

```
<EnumeratedItem CodedValue="2MCA" OrderNumber="1">
   <Alias Name="C177957" Context="nci:ExtCodeID"/>
</EnumeratedItem>
```

Figure 13. XML Code for LBTESTCD Codelist Showing Term Before Updated to *MCA2*

DD0029

The issue for DD0029 is also related to Controlled Terminology, but the message for this issue, "Required attribute def:ExtendedValue is missing or empty", is less straightforward.

Hmm, extended… This sounds awfully similar to "extensible". If your first thought was the "Codelist Extensible (Yes/No)" column in the CT file, you're on the right track! **A+** for making it this far into the paper - and for retaining the information! You're already putting it to good use!

If a codelist in your define.xml file exists in the CDISC Controlled Terminology version your study is using, but not all terms within that codelist are present in the CDISC CT, then all terms *not* included in the CDISC CT codelist must have the XML coding set to "Yes" for an attribute called def:ExtendedValue. This can occur when a value other than the CDISC Submission Value (usually the Synonym or NCI Preferred Term value) is added to the define.xml and associated with the Code (for the term).

This also presents when a term and Code added to a newer version of the CDISC CT is used in a define.xml that has been validated using an older version, where that term did not yet exist within the codelist. For example, your study is using the IPSS questionnaire, for which a QSCAT term was recently added to SDTM Terminology in December 2021, but the SDTM CT used for the study was the latest version available at study start in June of 2019.[14]

However, your new colleague used the latest version of CT when creating the QSCAT codelist, and the term *IPSS* that was just added in the December 2021 version of CT was accidentally added with its NCI Term Code to the QSCAT codelist in your define.xml. Since you validate with CT version dated 2019-06-28, the DD0029 issue appears because the term *IPSS* did not exist within the QSCAT codelist in that version of SDTM Terminology.[9]

If this DD0029 issue appears in your validation report, verify if the correct version of CDISC CT was used in the validation, then confirm whether the CodedValue listed in your validation report is present within the codelist of the CDISC CT version your study is using. If the term is not listed in the same codelist within the CDISC CT, it should have def:ExtendedValue="Yes".

"But," you think, *"there's no place in the Pinnacle 21 define.xml Excel spec to put def:ExtendedValue. So, do I have to manually update the XML code? I don't even know XML!"*

Don't panic. You don't have to manually update the XML code. In order to fix this issue using the Pinnacle 21 define.xml Excel spec:

1. In the Codelists tab, locate the term within the codelist specified in the validation report for this issue. In this example, locate the row where Term is *IPSS*.
2. Remove the value for NCI Term Code - do NOT remove the value for the NCI Codelist Code!
3. Generate the define.xml file using your Pinnacle 21 define.xml Excel spec. Pinnacle 21 Community will automatically generate the def:ExtendedValue attribute for you.

	A	B	C	D	E	F	G	H
	ID	Name	NCI Codelist Code	Data Type	Order	Term	NCI Term Code	Decoded Value
	QSCAT	Category of Questionnaire	C100129	text	1	IPSS	C113446	

Figure 14. Pinnacle 21 define.xml Excel Spec Where NCI Term Code Should Be Removed

If you're feeling confident and want to do this directly in the XML code, follow these steps:

1. Open your define.xml file using a text editor and look for the code block within quotation marks for the CodedValue containing the term reported for this issue in the validation report, as in Figure 15. In this example, search for the term *IPSS*.

```
<CodeList OID="CL.QSCAT" Name="Category of Questionnaire" DataType="text">
   <EnumeratedItem CodedValue="IPSS" OrderNumber="1">
      <Alias Name="C113446" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
```

Figure 15. XML Code Showing Code Block For Term *IPSS* Within QSCAT Codelist Without def:ExtendedValue Attribute

2. Add the code [space]def:ExtendedValue="Yes", directly after the OrderNumber attribute, as in Figure 16.

3. Remove the code for Alias Name=[NCI Code], as in Figure 16. Do NOT remove the XML code "</EnumeratedItem>" for the Codelist Code.

```
<CodeList OID="CL.QSCAT" Name="Category of Questionnaire" DataType="text">
   <EnumeratedItem CodedValue="IPSS" OrderNumber="1" def:ExtendedValue="Yes">
   </EnumeratedItem>
```

Figure 16. XML Code Showing Updated Code Block For Term *IPSS* Within QSCAT Codelist With def:ExtendedValue Attribute

DD0024

Cue the drum roll. And now, for one of the top issues when validating the define.xml, it's time to discuss DD0024!

DD0024 states "Invalid Term in codelist <codelist> for variable <variable>" and appears when the define.xml includes a term within a codelist and the specified Controlled Terminology does not include that same term within the same codelist.

In many cases, this issue appears in relation to non-extensible codelists. Remember in our example for DD0031 where we discussed that the codelist Action Taken with Study Treatment (ACN) is a non-extensible codelist? If a term is added to this codelist in the define.xml file, then you'll see a DD0024 issue in the validation report.

Frequently, this occurs when the terms *MULTIPLE* or *OTHER* are added as extended terms to codelists. According to various SDTMIG versions, the terms *MULTIPLE* and *OTHER* can be used in special cases; however, when these terms are not officially part of the CDISC published codelists, the DD0024 issue will be present in the validation report for SDTM and ADaM, and it will need to be explained in the Reviewer's Guide.

This may lead you to scratch your head and say, *"Hmm, why would Pinnacle 21 still choose to include these terms within the logic for this issue? Can't they just exclude them?"* Currently, CDISC has not published guidance on acceptable scenarios that would (and would not) be appropriate for these terms to be present. Thus, preventing this rule from firing for *MULTIPLE* and *OTHER* is not a simple determination.

In addition, according to the FDA's TCG, the use of *OTHER* as a variable term is disparaged in cases where a sufficient term within Controlled Terminology exists, and all –DECODE values of *OTHER* must be further explained in the Reviewer's Guide.[8] This scenario makes the DD0024 warning extremely valuable.

Aside from cases of terms added to extensible codelists, there are other scenarios where this issue is found as well. Sometimes, this can stump users because the Implementation Guides can seem so straightforward - until you realize there are several caveats that apply to specific cases or learn there are other documents to consider, such as those containing specific conformance rules.

A more complicated example of the DD0024 issue is when the define.xml is created using a codelist present within the IG for a variable, but a caveat that limits the codelist's use for that variable has not been considered. This occurs more frequently for a few specific variables in SDTM, such as –PRESP and –BLFL, as well as Flag variables in ADaM and SEND.

For example, the variable LBBLFL is associated with the No Yes Response (NY) codelist within the SDTMIG, beginning in v3.1.2.[4, 5, 6, 7] Often, this leads define.xml creators to include all terms from the NY codelist within their define.xml file because they see this specific codelist has been specified for this variable in the IG. However, the text in the CDISC Notes column of the IG further instructs that values for the LBBLFL variable can only be *Y* or null. Therefore, *Y* is the only allowable term for the codelist associated with the LBBLFL in the Define.

LBBLFL	Baseline Flag	Char	(NY)	Record Qualifier	Indicator used to identify a baseline value. The value should be "Y" or null.	Exp

Figure 17. SDTMIG v3.2 LBBLFL Variable Associated with NY Codelist

If the codelist associated with LBBLFL in the define.xml file contains terms other than just *Y*, the DD0024 issue will appear in the validation report. This occurs because the validator is enhanced to account for the note related to baseline flag values as a matter of compliance.

To resolve the issue in this scenario, a subset NY codelist is used for –BLFL variables that includes only "Y" as a term.

To research and resolve the DD0024 issue when it occurs for –FL variables, we suggest the following:

In the Pinnacle 21 define.xml Excel spec:

1. Go to the Variables tab and search for the variable indicated by the DD0024 issue. In this example, it would be "LBBLFL".

Order	Dataset	Variable	Label	Data Type	Length	Significant Digit	Format	Mandatory	Codelist	Origin
18	LB	LBBLFL	Baseline Flag	text	1		$1	No	NY	Derived

Figure 18. Pinnacle 21 define.xml Excel Spec Displaying Incorrect Codelist for LBBLFL

2. Note the value in the Codelist column. In this example, it is "NY".
3. Change the Codelist value to the value of the subset codelist. In this example, the subset codelist is "Y".

Order	Dataset	Variable	Label	Data Type	Length	Significant Digit	Format	Mandatory	Codelist	Origin
18	LB	LBBLFL	Baseline Flag	text	1		$1	No	Y	Derived

Figure 19. Pinnacle 21 define.xml Excel Spec Displaying Corrected Codelist for LBBLFL

4. Navigate to the Codelists tab in the Pinnacle 21 define.xml Excel spec.
5. Filter the ID column to show only the codelist associated with the variable from Step 2. In this example, filter ID on "NY" since the NY codelist was associated with the variable LBBLFL as seen on the Variables tab.
6. Add a record for the subset codelist, including the NCI Codelist Code and NCI Term Code values.

ID	Name	NCI Codelist Code	Data Type	Order	Term	NCI Term Code	Decoded Value
NY	No Yes Response	C66742	text	1	N	C49487	
NY	No Yes Response	C66742	text	2	Y	C49488	
Y	No Yes Response (Yes only)	C66742	text	1	Y	C49488	

Figure 20. Pinnacle 21 define.xml Excel Spec After Subset Codelist "Y" Was Added

In the XML code:

1. Search the XML code on the variable indicated by the DD0024 issue and look for the ItemDef coding. In this example, search on LBBLFL.
2. Within the ItemDef code block, find the CodeListOID and replace the value in quotations with the new value of the subset codelist ID. In this example, find the ItemDef code block that displays when searching for LBBLFL, as shown in Figure 21. Update the CodeListOID value to "CL.Y" – it was originally "CL.YN."

```
<ItemDef OID="IT.LB.LBBLFL" Name="LBBLFL" DataType="text" Length="1"
         SASFieldName="LBBLFL"
         def:DisplayFormat="$1">
   <Description>
      <TranslatedText xml:lang="en">Baseline Flag</TranslatedText>
   </Description>
   <CodeListRef CodeListOID="CL.Y"/>
```

Figure 21. XML Code for LBBLFL After CodeListOID Corrected

3. Within the XML Codelist section, which we again recommend you keep in alphabetical order by codelist, add XML code for the codelist element, including the Code as the Alias Name and the

Codelist Code as the Alias Name for the codelist. In this example, add XML code for CodeList OID="CL.Y" with Alias Name="C66742" for the Codelist, and then add XML code for the term *Y* with Alias Name="C49488" for the term.

```
<CodeList OID="CL.NY" Name="No Yes Response" DataType="text">
   <EnumeratedItem CodedValue="N" OrderNumber="1">
      <Alias Name="C49487" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <EnumeratedItem CodedValue="Y" OrderNumber="2">
      <Alias Name="C49488" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <Alias Name="C66742" Context="nci:ExtCodeID"/>
</CodeList>
<CodeList OID="CL.Y" Name="No Yes Response (Yes only)" DataType="text">
   <EnumeratedItem CodedValue="Y" OrderNumber="1">
      <Alias Name="C49488" Context="nci:ExtCodeID"/>
   </EnumeratedItem>
   <Alias Name="C66742" Context="nci:ExtCodeID"/>
</CodeList>
```

Figure 22. XML Code Updated to Add Subset Codelist "Y"

Another scenario similar to –PRESP and Flag variables occurs with Relative Timing Variables, everyone's favorite type of variables! Although –STRF/–ENRF and –STRTPT/–ENRTPT are all associated with the STENRF codelist, some terms in the STENRF codelist aren't suitable for use for all these variables, just as some terms in the NY codelist were unsuitable for use in the –BLFL variables.

To handle this within the scope of validation, subset codelists were developed to apply separately for both –STRTPT and –ENRTPT. According to the SDTMIG, beginning with v3.1.2 when these variables were first introduced, for –STRTPT, only the values *AFTER*, *BEFORE*, *COINCIDENT*, or *UNKNOWN* are allowable as terms within the codelist associated with this variable. For –ENRTPT, only the values *AFTER*, *BEFORE*, *COINCIDENT*, *ONGOING*, or *UNKNOWN* can be present in the codelist associated with this variable.[4, 5, 6, 7]

If any other terms are added to the codelist associated with the –STRTPT variable, even when present within the STENRF codelist in SDTM Terminology, the DD0024 issue appears in the validation report. Likewise, this issue will be present in the validation report for –ENRTPT if any terms other than those listed above are added into the codelist associated with the –ENRTPT variable.

CONCLUSION: DREAMING OF DEFINE

Define.xml validation issues can seem daunting at first glance, but as with anything, practice makes perfect! This guidance may have given you a mild case of OCD when it comes to your define.xml, but we hope it has also provided you with the tools and confidence you need to create impeccable data definition files.

With this in mind, you'll be sure to check the versions of the standard and Controlled Terminology configured for the validation, the IG version reported in the define.xml file itself, the Define-XML Specification version that corresponds with the version of define.xml you're creating, and the MedDRA version that should be reported for its attribute. You'll study the CDISC Controlled Terminology files and always reference the correct version when creating codelists and terms. You'll dominate the subset codelists used for flag variables and reference timepoints, and you'll crush all those codelist codes and term codes.

You've armed yourself with knowledge, and now it's time to get out there and conquer your define.xml. If there's something you don't know, don't be afraid to ask - there's likely someone within your organization who's been in your shoes. Or you can even browse for answers on the Pinnacle 21 forum or blog.

You can do this - you're a Conformance Champion! Instead of searching for bravery at the bottom of a bottle, raise your glass to toast a triumphant trial. And don't forget to replace your "swear jar" with one

labeled “Dollars for Define”; just toss in a dollar every time you complete a define.xml file, fix a validation issue, or help someone else resolve issues with theirs, and you’ll be relaxing on the beach in no time. Cheers to you and all your future successes!

REFERENCES

[1] CDISC define.xml Team. 2005. “Case Report Tabulation Data Definition Specification (define.xml).” Accessed March 7, 2022. https://www.cdisc.org/standards/data-exchange/define-xml/define-xml-v1-0

[2] CDISC Define-XML Team. 2013. “CDISC Define-XML Specification Version 2.0.” Accessed March 7, 2022. https://www.cdisc.org/standards/data-exchange/define-xml/define-xml-v2-0

[3] CDISC Define-XML Team. 2019. “CDISC Define-XML Specification Version 2.1 (Final).” Accessed March 7, 2022. https://www.cdisc.org/standards/data-exchange/define-xml/define-xml-v2-1

[4] CDISC Submission Data Standards Team. 2008. “Study Data Tabulation Model Implementation Guide: Human Clinical Trials Version 3.1.2.” Accessed March 8, 2022. https://www.cdisc.org/standards/foundational/sdtmig/sdtmig-v3-1-2

[5] CDISC Submission Data Standards Team. 2013. “Study Data Tabulation Model Implementation Guide: Human Clinical Trials Version 3.2.” Accessed March 8, 2022. https://www.cdisc.org/standards/foundational/sdtmig/sdtmig-v3-2

[6] CDISC Submission Data Standards Team. 2018. “CDISC Study Data Tabulation Model Implementation Guide: Human Clinical Trials Version 3.3 (Final).” Accessed March 8, 2022. https://www.cdisc.org/standards/foundational/sdtmig/sdtmig-v3-3

[7] CDISC Submission Data Standards Team. 2021. “Study Data Tabulation Model Implementation Guide: Human Clinical Trials Version 3.4 (Final).” Accessed March 8, 2022. https://www.cdisc.org/standards/foundational/sdtmig/sdtmig-v3-4

[8] Food and Drug Administration, Center for Drug Evaluation and Research (CDER), & Center for Biologics Evaluation and Research (CBER). 2022, March. “Study data technical conformance guide v4.9.” Accessed March 7, 2022. https://www.fda.gov/media/153632/download

[9] National Cancer Institute. 2019, June 28. “Study Data Tabulation Model (SDTM) Terminology.” Accessed March 9, 2022. https://datascience.cancer.gov/resources/cancer-vocabulary/cdisc-terminology

[10] National Cancer Institute. 2020, September 25. “Study Data Tabulation Model (SDTM) Terminology.” Accessed March 9, 2022. https://datascience.cancer.gov/resources/cancer-vocabulary/cdisc-terminology

[11] National Cancer Institute. 2020, November 6. “Study Data Tabulation Model (SDTM) Terminology.” Accessed March 9, 2022. https://datascience.cancer.gov/resources/cancer-vocabulary/cdisc-terminology

[12] National Cancer Institute. 2021, March 26. “Study Data Tabulation Model (SDTM) Terminology.” Accessed March 9, 2022. https://datascience.cancer.gov/resources/cancer-vocabulary/cdisc-terminology

[13] National Cancer Institute. 2021, June 25. “Study Data Tabulation Model (SDTM) Terminology.” Accessed March 9, 2022. https://datascience.cancer.gov/resources/cancer-vocabulary/cdisc-terminology

[14] National Cancer Institute. 2021, December 17. “Study Data Tabulation Model (SDTM) Terminology.” Accessed March 9, 2022. https://datascience.cancer.gov/resources/cancer-vocabulary/cdisc-terminology

[15] National Cancer Institute. 2022, March 25. “Study Data Tabulation Model (SDTM) Terminology.” Accessed March 7, 2022. https://datascience.cancer.gov/resources/cancer-vocabulary/cdisc-terminology

[16] Pinnacle 21. 2021, March 3. “Controlled Terminology Best Practices.” Accessed March 11, 2022. https://www.pinnacle21.com/blog/controlled-terminology-best-practices

ACKNOWLEDGMENTS

We’d like to thank Sergiy Sirichenko and Wendy Young for their valuable feedback and edits.

CONTACT INFORMATION

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

Julie Ann Hood
Pinnacle 21
jhood@pinnacle21.com
https://www.linkedin.com/in/julie-ann-hood-40350525

Sarah Angelo
Pinnacle 21
sangelo@pinnacle21.com
https://www.linkedin.com/in/sarah-angelo-2b1a7a3a

Any brand and product names are trademarks of their respective companies.