PharmaSUG 2017 - Paper DS17

ADaM Grouping: Groups, Categories, and Criteria. Which Way Should I Go?

Jack Shostak, Duke Clinical Research Institute

ABSTRACT

ADaM has variables that allow you to cluster, group, or categorize information for analysis purposes. Sometimes it may not be entirely clear to you which variable you should be using and when. The goal of this paper is to help to provide some guidance around what ADaM grouping variables are available, what is appropriate and when, and then to discuss when more than one technique will work for a given analysis situation. We will also look at problems where a single solution isn't entirely obvious. The paper focus will be primarily on Basic Data Structure (BDS) grouping variables, although other non-BDS variables will be mentioned. The following ADaM BDS variables will be examined: *GRy(*Gy), *CATy, CRITy, MCRITy, AVALCATy, and PARAM. These will be compared and contrasted. The paper will conclude with suggested ADaM categorization strategies as well as ideas for where ADaM can be improved in this regard. This paper is targeted at CDISC users with some basic exposure to the CDISC ADaM model.

INTRODUCTION

This paper is intended to serve as guidance for programmers and statisticians in clinical research who are implementing ADaM. ADaM has a number of categorical grouping variables, and often people may implement a method that is less than optimal for their needs. At worst, they may implement an ADaM grouping variable in a way that causes a violation of ADaM rules and Pinnacle 21 validation checks. I hope that this paper helps to prevent that from happening to someone, as it is better to design ADaM properly up front, and not have to fix an ADaM design issue due to a Pinnacle 21 error at the time of submission to the FDA.

Since this paper is intended for those that have at least some experience implementing and specifying ADaM data structures, familiarity with ADaM and the SDTM would be good to have. This paper is based on ADaM 2.1 and ADaM Implementation Guide 1.1.

CATEGORIZATION AND GROUPING NEEDS

When designing ADaM analysis datasets, you may have various needs for data categorization. A primary need to categorize data is to group similar values for descriptive and inferential categorical data analysis. For example, you may need to group treatment responders and non-responders to compare them by treatment. You may need to group treatments themselves, for example if you are analyzing all-treated versus placebo patients. You may also need to group data to create covariates for your statistical models. Perhaps you need to create age groupings as a model covariate for example.

There are other needs for data grouping. You may need to group patients for population determination. You may also need to categorize data observations so that you can select the proper records for analysis purposes. Finally, you may simply want to group and categorize data for presentation purposes where you want certain parameters to appear in a specific order in your results.

REVIEW OF SOME ADAM CATEGORIZATION TOOLS

When designing ADaM analysis datasets, you may have various needs for data categorization. A primary need to categorize data is to group similar values for descriptive and inferential categorical data analysis. For example, you may need to group treatment responders versus non-responders to compare by treatment. You may also need to group treatments themselves, for example if you are analyzing all-treated versus placebo patients. You may also need to group data to create covariates for your statistical models. Perhaps you need to create age groupings as a model covariate for example.

PARCAT PARAMETER CATEGORIZATION

PARCATy is intended to group PARAM values into categories. It is important to remember that PARAM to PARCATy is designed to be a many-to-one mapping where any given PARAM may be associated with at most one level of PARCATy. Often times, people try to use PARCATy to subdivide or qualify PARAM values further, and this is not allowed. Here is a visual representation of this situation:

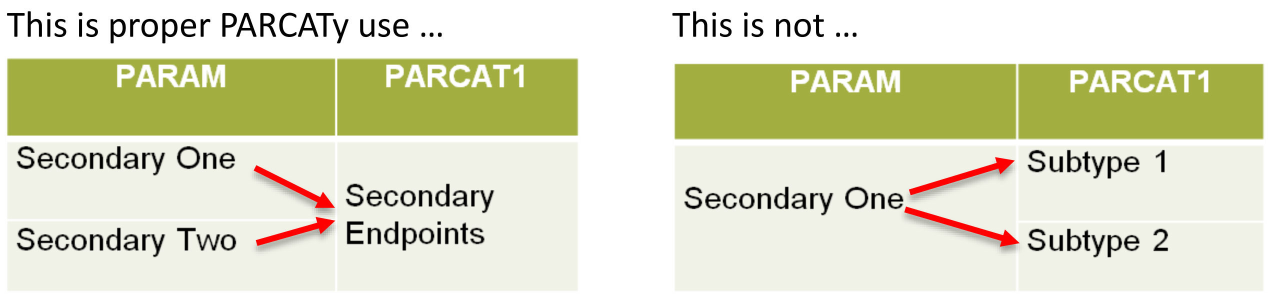

Figure 1 Proper and Improper Use of PARCATy

Keep in mind that the ADaM Implementation Guide doesn't explicitly preclude PARCATy from having a dependence on another variable beyond PARAM. However, I suggest you be wary of using PARCATy in this fashion.

*GR GROUPING VARIABLES

ADaM also has a set of general grouping variables in *GRy and *GRyN. From the ADaM Implementation Guide section 3.1.1 on General Variable Conventions rule #9, it states:

> Variables whose names end in GRy, Gy, or CATy are grouping variables, where y refers to the grouping scheme or algorithm (not the category within the grouping).

In addition, in the same section 3.1.1, rule #10 states:

> It is recommended that producer-defined grouping or categorization variables begin with the name of the variable being grouped and end in GRy (e.g., variable ABCGRy is a character description of a grouping or categorization of the values from the ABC variable for analysis purposes). If any grouping of values from an SDTM variable is done, the name of the derived ADaM character grouping variable should begin with the SDTM variable name and end in GRy.

There are many use cases for grouping SDTM variables in these *GRy variables. The ADaM Implementation Guide defines a number of ADaM *GRy variables such as these:

- SITEGRy
- RACEGRy
- AGEGRy
- TRxxPGy/TRxxAGy/TRTPGy/TRTAGy (note the R in GR is dropped here due to variable length issues)
- DTHCGRy (based on ADaM's DTHCAUS variable)

*GRy (and *Gy) variables are often used to group SDTM content, but they can be used for non-AVAL based ADaM variable groupings as well. The nice thing about *GRy variables is that they are inherently self-descriptive by nature of their naming convention.

*CAT ANALYSIS VARIABLE CATEGORIZATION VARIABLES

ADaM has a set of categorization variables that exist to group analysis values such as Basic Data Structure AVAL, AVALC, BASE, CHG, and PCHG ADaM variables. The following categorical variables are generally used to categorize the BDS AVAL/BASE/CHG/PCHG analysis values respectively:

- AVALCATy
- BASECATy
- CHGCATy
- PCHGCATy

We can extrapolate the following definition from the ADaM Implementation Guide definitions for the *CATy variables. Note that this text is an extrapolation of the ADaM Implementation Guide text, and not the text you will find in the Implementation Guide itself.

> *CATy serves as a categorization of the analysis variable (e.g., AVAL/AVALC) within a parameter. It is intended to be a many to one mapping, not a one to many as in subcategorization or qualification of an AVAL value.

The following table shows an example of how to use AVALCATy to create a binary categorization out of four levels of pain severity:

USUBJID	PARAM	AVALC	AVALCAT1
101	Pain Severity	None	None or Mild
102	Pain Severity	Severe	Moderate or Severe
103	Pain Severity	Moderate	Moderate or Severe
104	Pain Severity	Mild	None or Mild

Table 1 Example of How to Use AVALCATy

Keep in mind that the ADaM Implementation Guide doesn't explicitly preclude analysis variable *CATy variables from having a dependence on another variable beyond PARAM. However, I suggest you avoid such a use for AVALCATy as you will see in case study #1 below.

(M)CRITY CRITERIA RECORD SELECTION VARIABLES

ADaM has a set of "criteria" variables that also serve to categorize a set of records to a specific criteria. The original intent behind the creation of (M)CRITy was to select subgroups of records that met a given criteria. The (M)CRITy variable set contains:

- A text string identifying a pre-specified criterion within a parameter (CRITy or MCRITy) and…
- For CRITy, its associated boolean flag CRITyFL

  or…
- For MCRITy, its associated multichotomous result in MCRITyML

CRITyFL and MCRITyML are defined in Implementation Guide table 3.3.4.2. These character flag variables indicate whether the criterion defined in (M)CRITy was met by the data on the record. This is a key point. You may want to use (M)CRITy to identify a criteria for data across more than one row of a BDS dataset, but that should not be done. In that case, you will want to create a new PARAM to meet that need.

Section 4.7 of the ADaM Implementation Guide states:

> "The definition of **CRITy can use any variable(s) located on the row**, and the definition must stay constant across all rows within the same value of PARAM. **A complex criterion which draws from multiple rows (different parameters or multiple rows for a single parameter) will require a new PARAM be created.**"
>
> "CRITy for one parameter can be different than CRITy for a different parameter in the same dataset."
>
> "MCRITy is populated with a text description identifying the criterion being evaluated. The definition of **MCRITy can use any variable(s) located on the row** and the definition must stay constant across all rows within the same value of PARAM. **A complex criterion which draws from multiple rows will require a new PARAM be created**."

Here is a simple use case for CRITy/CRITyFL where we want to categorize records that have systolic blood pressure greater than 160 or not:

USUBJID	PARAM	AVAL	CRIT1	CRIT1FL
101	Systolic Blood Pressure (mm Hg)	163	SBP > 160	Y
102	Systolic Blood Pressure (mm Hg)	133	SBP > 160	N
103	Systolic Blood Pressure (mm Hg)	120	SBP > 160	N
104	Systolic Blood Pressure (mm Hg)	165	SBP > 160	Y
105	Systolic Blood Pressure (mm Hg)	140	SBP > 160	N

Table 2 Sample use of CRITy and CRITyFL

Here is a similar sample where MCRITy is being used for systolic blood pressure classification where we extend beyond the previous binary classification and go to a multichotomous solution:

USUBJID	PARAM	AVAL	MCRIT1	MCRIT1ML
101	Systolic Blood Pressure (mm Hg)	163	SBP Classification	SBP >= 160
102	Systolic Blood Pressure (mm Hg)	133	SBP Classification	120 >= SBP >= 139
103	Systolic Blood Pressure (mm Hg)	120	SBP Classification	120 >= SBP >= 139
104	Systolic Blood Pressure (mm Hg)	165	SBP Classification	SBP >= 160
105	Systolic Blood Pressure (mm Hg)	140	SBP Classification	140 >= SBP >= 159

Table 3 Sample Use of MCRITy and MCRITyML

(M)CRITy is nice in that it codifies the criteria into the dataset as a data element. It essentially places the definition of the flag variable CRITyFL/MCRITyML into the dataset itself as (M)CRITy. Again, you cannot create CRITyFL/MCRITyML results based on information found across multiple BDS rows. In that case, you likely need to create a new PARAM.

CREATING A NEW PARAM AS A CATEGORIZATION

The ADaM Basic Data Structure readily allows for you to add new PARAM values to the data structure as it is designed as a name value pair dataset to be extended vertically. If your categorization or grouping concept is one that will require information from multiple rows of a Basic Data Structure Dataset, then creating a new PARAM to capture that information is oftentimes your best solution.

CREATING A NEW BDS COLUMN/VARIABLE AS A CATEGORIZATION

Although the ADaM Basic Data Structure is readily extended with new PARAM values and rows, under limited circumstances you can also add new column variables. You should refer to section 4.2 “Creation of Derived Columns versus Creation of Derived Rows” in the ADaM Implementation Guide for more details. That said, the key thing to remember here is rule #1 that states, “A parameter-invariant function of AVAL and BASE on the same row that does not involve a transform of BASE should be added as a new column.” Otherwise, you are likely to be adding new PARAM values.

CATEGORIZING LAB CHANGE VALUES FOR SHIFT TABLES

It is worth briefly mentioning that the ADaM Basic Data Structure has special variables that are used in support of production of laboratory shift tables. Shift tables allow you to present how laboratory parameters change for patients from baseline to some follow up point in time. ADaM has Basic Data Structure variable SHIFTy that indicates the category of the lab value’s “shift” from some earlier point in time. So, you could say that for a hemoglobin value, the SHIFT1 value is “NORMAL to HIGH” which would indicate that at baseline the hemoglobin was normal but at that particular follow up assessment it was high. These are useful categorization variables because by design they allow you to look across observations in a Basic Data Structure dataset and they greatly assist in the production of shift tables.

CASE STUDIES

Now that you have seen a summary of a number of ADaM grouping and categorization variables, let’s look at a few use cases to see these ADaM variables in action.

CASE STUDY 1: CLINICAL RESPONSE

In this case study, you have a nootropic drug study and the Basic Data Structure analysis value contains the cognitive score response value. Your goal is to create a Basic Data Structure clinical response variable containing “Not effective”, “Effective”, or “Very effective” which also happens to be dependent on the subject’s AGE. Here are the assessment criterial for categorizing AVAL by age:

Age 18-50		Age > 50	
AVAL	**RESULT**	**AVAL**	**RESULT**
<15	Not Effective	<10	Not Effective
15-30	Effective	10-20	Effective
>30	Very Effective	>20	Very Effective

Table 4 AVAL Categorization Rules

Here is the raw cognition score data as a piece of a BDS dataset with AGE included from ADSL:

USUBJID	AVISIT	PARAM	AVAL	AGE
101	Month 1	Cognition	15	20
101	Month 2	Cognition	25	20
101	Month 3	Cognition	29	20
102	Month 1	Cognition	15	65
102	Month 2	Cognition	25	65
102	Month 3	Cognition	26	65

Table 5 BDS Cognition Scores Dataset

Can you use AVALCAT to categorize AVAL here?

Per the ADaM Implementation Guide, AVALCATy is, “A categorization of AVAL or AVALC within a parameter.” However, in this case, your categorization is not solely based on AVAL. You have a dependency on the AGE variable for categorization of AVAL. The Implementation Guide text does not explicitly preclude AVALCATy from including a dependency on something other than AVAL, but it is implied by the text and the variable name itself. Because of this ambiguity, and the dependency on a variable outside of AVAL, using AVALCATy may not be the best approach.

Can you use (M)CRIT here?

Yes you can, because all of the needed data for the categorization is on the row and the definition is consistent within a parameter. However, you would need to use MCRITy and MCRITyML due to the multichotimous response. Such an approach might look like this in a Basic Data Structure dataset:

USUBJID	AVISIT	PARAM	AVAL	AGE	MCRIT1	MCRIT1ML	MCRIT2	MCRIT2ML
101	Month 1	Cognition	15	20	Clinical Response (Age 18-50)	Effective	Clinical Response (Age over 50)	
101	Month 2	Cognition	25	20	Clinical Response (Age 18-50)	Effective	Clinical Response (Age over 50)	
101	Month 3	Cognition	29	20	Clinical Response (Age 18-50)	Effective	Clinical Response (Age over 50)	
102	Month 1	Cognition	15	65	Clinical Response (Age 18-50)		Clinical Response (Age over 50)	Effective
102	Month 2	Cognition	25	65	Clinical Response (Age 18-50)		Clinical Response (Age over 50)	Very Effective
102	Month 3	Cognition	26	65	Clinical Response (Age 18-50)		Clinical Response (Age over 50)	Very Effective

Table 6 Using MCRITy/MCRITyML to Categorize

Keep in mind that a dataset in this type of structure may strain a principal of ADaM in that the analysis dataset should be analysis ready. Using MCRITy in this fashion may actually make report production a bit difficult as you may have to transform the data a bit in your table production, or have some slightly awkward coding.

Can you use a new PARAM here?

Yes you can. You will find that you can almost always generate a new PARAM to suit your needs. Such a solution might look like this in a sample BDS dataset where we add “Clinical Response” as a new PARAM:

USUBJID	AVISIT	PARAM	AVAL	AVALC	AGE
101	Month 1	Cognition	15		20
101	Month 1	**Clinical Response**		Effective	20

101	Month 2	Cognition	25		20
101	Month 2	**Clinical Response**		Effective	20
101	Month 3	Cognition	29		20
101	Month 3	**Clinical Response**		Effective	20

Table 7 PARAM Used for Clinical Response

Creating a new PARAM actually works pretty well to produce a traditional table that looks like this:

```
                         Treatment A  Treatment B
Parameter                  (n=xxx)      (n=xxx)        p-value

Cognition                                              xxxx.x
    N                        xxx          xxx
    Mean                     xxx.x        xxx.x
    Std                      xxx.xx       xxx.xx
    Min-Max                  xxx-xxx      xxx-xxx
Clinical Response                                      xxxx.x
    Not Effective            xxx(xxx.x%)  xxx(xxx.x%)
    Effective                xxx(xxx.x%)  xxx(xxx.x%)
    Very Effective           xxx(xxx.x%)  xxx(xxx.x%)
```

Table 8 Table Shell with Cognition and Clinical Response

If you look at that BDS dataset with in Table 7 the new PARAM that was created, you might notice that the two PARAMs are closely related in a stair-step data shape. Why can't you just collapse those two rows, and make AVALC look like what you see here in Table 9?

USUBJID	AVISIT	PARAM	AVAL	AVALC	AGE
101	Month 1	Cognition	15	Effective	20
101	Month 2	Cognition	25	Effective	20
101	Month 3	Cognition	29	Effective	20

Table 9 Collapsing PARAMs

As nice as that would be, and as nice as that looks, you cannot do that because it would violate the definition of AVAL and AVALC. In this case, AVAL to AVALC isn't 1-1 within the PARAM, as you see circled here in Table 10:

USUBJID	AVISIT	PARAM	AVAL	AVALC	AGE
101	Month 1	Cognition	15	Effective	20
101	Month 2	Cognition	**25**	**Effective**	20
101	Month 3	Cognition	29	Effective	20
102	Month 1	Cognition	15	Effective	65
102	Month 2	Cognition	**25**	**Very Effective**	65
102	Month 3	Cognition	26	Very Effective	65

Table 10 Invalid Use of AVAL and AVALC

You cannot have the same value of AVAL within a PARAM as you see here mapping to two different values of AVALC.

Can you use ANLzzFL here?

Can you just use ANLzzFL to flag the records that have AVAL values that are a positive clinical response? As nice and as easy as that might seem, you should not do that because ANLzzFL variables are meant to serve as additional and more granular record selection flags and are not intended to be used as results values themselves. In the ADaM Implementation Guide, it states:

> “ANLzzFL is a conditionally required flag to be used in addition to other selection variables when the other selection variables in combination are insufficient to identify the exact set of records used for one or more analyses.”
>
> “When one is defining the set of records used in a particular analysis or family of analyses, ANLzzFL is supplemental to, and is intended to be used in conjunction with, other selection variables, such as subject-level, parameter-level and record-level population flags, AVISIT, DTYPE, grouping variables such as SITEGRy, and others.”

The “generic” nature of ANLzzFL has made it so that people want to use it to define new analysis variables, but that is not how ANLzzFL is intended to be used.

Can you create a custom BDS column variable here?

Could you create a custom BDS variable named CRESP here to indicate clinical response? Per the ADaM Implementation Guide section 4.2 it says, “Rule 1: A parameter-invariant function of AVAL and BASE on the same row that does not involve a transform of BASE should be added as a new column. So, that would probably not be the best decision because of the dependency on the AGE variable.

Conclusion on Case Study #1

Your ADaM compliant solutions to categorizing the AVAL value as being a clinical responder with AGE as a codetermining factor is to either use MCRITy/MCRIyML, or to create a new PARAM to capture clinical response. I would suggest that the best option would be to create a new PARAM, as that would be most likely to produce an analysis ready dataset. Whichever design works best for your end result, and makes your analysis work easier, is the way to go.

CASE STUDY 2: HIGH BLOOD PRESSURE (STAGE 2)

In the prior case study, you were categorizing an analysis value in a Basic Data Structure dataset. For this case study, you want to create an ADSL patient level flag that identifies subjects with Systolic BP >= 160 and Diastolic BP >= 100 at baseline. This stage 2 high blood pressure categorization will be used in your statistical modelling in later Basic Data Structure datasets. We will call this variable HBP2FL. How can we do this with categorical variables in ADaM?

Can you just add a new ADSL variable to capture the high blood pressure?

Yes you can, and your new ADSL high blood pressure flag variable might look like this:

USUBJID	HBP2FL
101	Y
102	N
103	Y

Table 11 Simple ADSL Flag Variable

That is a legal ADSL variable addition, but where is the traceability for this variable? The only way to know what this variable is would be to go look at the algorithm definition in define.xml, and see how this variable was calculated. Is there another way to do this that would add some better transparency and traceability to this created variable?

Adding supportive variables to HBP2FL

Instead of just flagging the patient as having stage 2 high blood pressure, what if you added flags for the component blood pressure values? How about adding the baseline systolic and diastolic values as well? Such an ADSL dataset extension might look like this:

USUBJID	HBP2FL	SYSBPFL	DIABPFL	SYSBPBL	DIABPBL
101	Y	Y	Y	165	100
102	N	Y	N	162	95
103	Y	Y	Y	180	110

Table 12 Adding Supportive ADSL Variables

Now you can see the overall stage 2 high blood pressure flag in HBP2FL, the respective systolic and diastolic baseline flags in SYSBPFL and DIABPFL, and the baseline blood pressure values in SYSBPFL and DIABPFL.

Creating a supportive BDS dataset

Another way to provide traceability to the ADSL stage 2 high blood pressure flag HBP2FL might be to add a supportive Basic Data Structure dataset. Here is a Basic Data Structure dataset of blood pressure values for subject 101:

USUBJID	AVISIT	PARAM	AVAL
101	Baseline	Systolic Blood Pressure (mm Hg)	165
101	Baseline	Diastolic Blood Pressure (mm Hg)	100

Table 13 BDS Blood Pressure Dataset

So, how can you categorize those two records to help with the ADSL HBP2FL variable? Do you use AVALCATy? Can you use the (M)CRITy variables? Do you create new BDS flag variables? You can employ AVALCATy to categorize these PARAMs as follows:

USUBJID	AVISIT	PARAM	AVAL	AVALCAT1
101	Baseline	Systolic Blood Pressure (mm Hg)	165	Systolic BP>= 160
101	Baseline	Diastolic Blood Pressure (mm Hg)	100	Diastolic BP >= 100

Table 14 Using AVALCATy to Categorize Blood Pressure

Could you just create new Basic Data Structure high blood pressure flag variables to suit your needs like this?

USUBJID	AVISIT	PARAM	AVAL	SYSFL	DIAFL
101	Baseline	Systolic Blood Pressure (mm Hg)	165	Y	
101	Baseline	Diastolic Blood Pressure (mm Hg)	100		Y

Table 15 Somewhat invalid use of new BDS flags

This would get past the Pinnacle 21 validator, but it isn't a really valid use of new Basic Data Structure column variables as these new flags are PARAM dependent, which is seemingly in violation of rule #1 in section 4.2 of the ADaM Implementation Guide.

We could also use the CRITy and CRITyFL variables for this categorization purpose as follows:

USUBJID	AVISIT	PARAM	AVAL	CRIT1	CRIT1FL
101	Baseline	Systolic Blood Pressure (mm Hg)	165	Systolic BP>= 160	Y
101	Baseline	Diastolic Blood Pressure (mm Hg)	100	Diastolic BP >= 100	Y

Table 16 Using CRITy and CRITyFL to Categorize

Now we need to combine those two criteria to create a stage 2 high blood pressure entity. This can best be done with a new PARAM as follows:

USUBJID	AVISIT	PARAM	AVAL	AVALC	CRIT1	CRIT1FL
101	Baseline	Systolic Blood Pressure (mm Hg)	165		Systolic BP >= 160	Y
101	Baseline	Diastolic Blood Pressure (mm Hg)	100		Diastolic BP >= 100	Y
101	Baseline	Systolic Blood Pressure >= 160 and Diastolic Blood Pressure >= 100		Y		

Table 17 New PARAM for Stage 2 High Blood Pressure

Here you can see the two systolic and diastolic PARAMs being used to create the composite PARAM. That new composite PARAM can then be used to define the ADSL HBP2FL variable directly. That being said, how you get this new composite Basic Data Structure PARAM value "back" into ADSL is a topic for another paper. I will say that you should never have a circular process where ADSL feeds a Basic Data Structure datasets downstream which then backtracks and feeds ADSL with new content.

Conclusion on Case Study #2

How to handle the ADSL variable categorization of patients into groups of stage 2 high blood pressure becomes a trail of how far you want to go with traceability. Do you create a simple ADSL HBP2FL variable and be done with it? Do you augment the ADSL HBP2FL variable with supportive variables? Alternately, do you create a supportive Basic Data Structure dataset showing how the categorization is done and then feed that result into ADSL? This categorization example shows how you can create much more transparency and traceability with more labor and cost. In the end, it is your decision to make as to which way to go.

CONCLUSION

SUGGESTIONS FOR THINGS TO DO WHEN CREATING ADAM CATEGORIZATIONS

Here are some suggestions for things you should try to do when creating ADaM categorizations:

- Try to keep your ADaM datasets as simple as you can, but consider traceability

  You want your ADaM datasets to be user friendly. Your datasets should allow for some level of traceability and transparency, but keep in mind end user usability as well. If you include too much additional information in your dataset, then you could actually make the data harder to understand while at the same time increasing your cost of work.

- Please refer to section 3 in the ADaM Implementation Guide for already defined ADaM categorization variables.
- Try to use *CATy variables to categorize ADaM analysis value variables, and *GRy variables to group other variable content.
- If CATy or (M)CRITy doesn't work for you, then consider creating a new PARAM instead.
- For complex categorizations, consider using (M)CRITy along with a new PARAM record to combine the composite information.
- Consider creating a new Basic Data Structure variable for additional categorizations

  Keep in mind for these new Basic Data Structure variables, that you have to follow the ADaM Implementation Guide rules for adding new columns to datasets. Also note that this method can limit traceability to the algorithm level metadata if used alone.

SUGGESTIONS FOR THINGS TO AVOID WHEN CREATING ADAM CATEGORIZATIONS

Here are some suggestions for things you should try to avoid when creating ADaM categorizations:

- Don't create new variables for categorization, when predefined ADaM categorization variables in the ADaM Implementation Guide such as SITEGRy or SAFFL exist.
- Don't use AVALC as a sub-categorization of AVAL. The AVAL to AVALC relationship must be 1-1. This is true for many other ADaM categorization variables as well (e.g., PARCATy, *GRy). This is seemingly an easy trap to fall into.
- Don't place analysis value concepts such as "clinical endpoint" into ANLzzFL variables as those are meant as a special record selection flags. It is easy to use ANLzzFL for this purpose, and some people do this to avoid Pinnacle 21 errors due to using other approaches. Remember that per the ADaM Implementation Guide ANLzzFL is, "to be used in addition to other selection variables when the other selection variables in combination are insufficient to identify the exact set of records used for one or more analyses."
- Don't use AVALCAT to subcategorize AVAL in a one to many way. AVALCAT is meant to categorize many to one. If you need a one to many categorization, then:
  - If all of your criteria data is on one row, you can use (M)CRITy.
  - If your data on one row, and it is a parameter invariant function of AVAL/BASE, you can create a new custom Basic Data Structure variable.
  - Create a new PARAM row.
- Don't create (M)CRITy variables in a way that they are defined based on the values from multiple rows. (M)CRITy must be defined on the content found on the data row per the ADaM Implementation Guide.

CLOSING THOUGHTS

ADaM gives you multiple ADaM compliant ways to categorize data and sometimes it isn't obvious as to which way to go. Often times, the most simple solution is the best one, so be careful of over-engineering your solutions. Generally, you would be well advised to work backwards from the statistical analysis plan. In other words, what analysis data structure, and by extension what ADaM categorization method will best allow you to produce the statistical summaries and figures that you wish to create? The appearance of your reporting often times will and should guide you on ADaM design. Finally, study the ADaM Implementation Guide for the categorization entities mentioned in this paper so that you are using ADaM in a compliant way. Study the ADaM implementation guide for detailed variable rules.

As ADaM progresses, more guidance around these categorization issues in general would be nice to have. It would be good if the definitions for *CATy explicitly allow or exclude dependence on other variable values on the row. Also, because it seems to happen a lot, it would be nice to have easier methods of subcategorization available so that people do not use categorization variables like AVALCATy as a subcategorization of AVAL.

ACKNOWLEDGMENTS

I would like to thank the fine people that work on ADaM development, students who take CDISC ADaM classes, and my colleagues at the Duke Clinical Research Institute. I learn far more from you all than you can imagine.

RECOMMENDED READING

- *Analysis Data Model (ADaM) Version 2.1*
- *Analysis Data Model Implementation Guide Version 1.1*

CONTACT INFORMATION

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

Jack Shostak
jack.shostak@duke.edu