

PP05 Titles are a Programmer's Best Friend

Title Driven Display Automation



Gregory Nunn, Statistical Programmer, Veramed Limited
gregory.nunn@veramed.co.uk

1. Introduction

When creating outputs for clinical reporting, much of the information a Clinical Programmer requires is contained within the title. An experienced programmer will **break the title down into components** and use these components to visualise the output. **So why don't we use these components to produce the output?**

When looking at an entire set of outputs for clinical reporting, a high percentage of them follow a clear structure, especially the basic summary tables. **This structure can be broken down into key components**, which can then be used to automate our outputs. As programmers, we read these components and are able to deduce not only which dataset is required, but also which parameters, variables and other critical characteristics of the output. By breaking the titles down into their components, programmers can utilize them to create more generic, adaptable programs.

2. Breaking Down the Title

Although it's not a comprehensive list, a large proportion of clinical reporting titles are built from any combination of the following components.

Summary of

[Treatment Phase]

[Variable]

By [Subgroup]

at/up to [Time Point]

for [Population]

- [Treatment Group]

Used to indicate the study phase, e.g. On-Treatment

Subgroup analysis usually comes in sets, defined in the SAP. Should indicate a variable from ADSL

Used when the population doesn't reflect the SAP

Should the trial look into treatment pooling, this can be described here

Will describe the dataset, parameter and variable to summarise:

- Vital Signs
- Adverse Events
- Chemistry Changes From Baseline
- PK Concentration

Often the time frame for the endpoint, e.g.: up to Week 52; at Day 10

The order of the components may differ from study to study, but the basic structure should remain consistent throughout each study.

3. Differing Approaches

Once the title has been broken down, there are two simple approaches which can be taken; (1) either **populating global macro variables** or (2) **calling pre-set macros which perform specific functions**. Depending on which component is being dealt with, one approach may be more beneficial than the other. For example, when dealing with large scale subgroup analyses, it may be efficient to create a macro that selects the subgroup from the title and then populates a global macro variable with this subgroup. Alternatively, you may want to call an existing standard macro, so if the variable component of the title is "Demographic Characteristics" this would then call the macro linked to this table.

4. Worked Example

For each component, a different check will need to be performed and should lie within their own macro, with some being more complex than others. Below is some example code which will select each component specified.

On-Treatment

Chemistry Changes From Baseline

by Region

```
if index(TITLE, 'On-Treatment') then %let phase = 'On-Treatment';;

if index(TITLE, 'Chemistry') then do;
  %let SUMM = CONT;
  %let DSET = ADAMDATA.ADLB;
  %let CAT = 'Chemistry';
  do j = 1 to countw(TITLE);
    if scan(TITLE, j, ' ') = 'Chemistry'
    and scan(TITLE, j+1, ' ') = 'Changes' then %let VAR = 'CHG';;
    else VAR = 'AVAL'; end; end;

do j = 1 to countw(TITLE);
  if scan(TITLE, j, ' ') = 'by' then do;
    select (scan(TITLE, j+1, ' '));
    when ('Region') %let BYGROUP = CTYRGR1;; end; end; end;
```

Once the macro variables have been set, they can be used to select how the table is produced. In the above, the macro variable &SUMM. can be used to decide which type of summary, continuous or categorical, is required. The remaining macro variables can then either be used to set parameters in the subsequent macro, or can be used throughout the macro. For instance, &BYGROUP. can be passed into the macro so checks can be performed as to whether subgroups are required, whereas &VAR. could be used as a general global macro variable.

5. Linking Everything Together

Once each macro has been created they can be inserted into any output program to allow for more generic, re-useable code. For example, if a study involves subgroup analysis, the production programmer would be able to create one program for an entire set of outputs and by inserting the subgroup macro, they will only have to update the title on any cloned outputs to create the entire set.

However, it is possible to link all the macros together. The most simplistic way of linking each component together would be a wrapper macro which would call the macro associated with each component. This wrapper macro could then be called itself by each output program and would allow the user to use the global macro variables produced. Alternatively, a system can be built which would read in a title and output a SAS file which in turn creates a predicted output. This would then allow for updates to be made as required.

6. Moving Forward

In the future, the incorporation of artificial intelligence into the system will produce more accurate outputs and therefore reduce time spent updating any that are produced. By reading any modifications made to the outputs back into the system, these can be interpreted and then applied in any future outputs with similar or identical titles to fine tune the system. This will lead to more consistent and reproducible outputs across studies.