



Quanticate

The Clinical Data Experts

Big SAS Code Navigation and Management

Fred Ross & Jessica Whittaker-Dixon

Agenda

- Introduction
- Planning, Sectioning & Naming Conventions
- PHUSE, Your Company & Your Own GPP
- Prefix Macros
- Macros in Practice
- Summary
- Questions

Introduction

Big SAS Code: Programs with a large amount of actual written code.

- Big code can provide challenges
- Greater care should be taken
- Common aspect of programming

PHUSE, Your Company & Your Own GPP

- SOPs, Training, Client Training
- PHUSE GPP + other sources
- **Your Own Programming Style** – Plan preferred practices after the above have been considered.



Contents [hide]

- 1 Good Programming Practice working project
- 2 Published Guidance Document
- 3 Introduction
- 4 Getting Started With a New Project
- 5 Language
- 6 Program header
 - 6.1 Revision history
- 7 Comments
- 8 Naming conventions
- 9 Coding conventions
 - 9.1 Log File Checking
 - 9.2 Portability
 - 9.3 Hard coding
 - 9.4 Defensive programming

Planning, Sectioning & Naming Conventions

Planning: Clear plan or guideline & naming conventions

Sections: Comments, macro calls, macro variables etc.

```
/*=====*/  
/* Main section */  
/*=====*/  
/*description: */  
/* */  
/*=====*/
```

```
/*==MAIN SECTION==*/  
%let prefix = LAB;  
  
*==sub-section==*;  
%let prefix = LAB1;  
%let subprefix =  
&prefix.1;  
%let subprefix =  
&prefix.2;  
  
*==sub-section==*;  
%let prefix = LAB2;
```

Prefix Macros

```
%macro prefix;
```

```
* Define global macros seqnumber prefix and section to  
create a unique prefix for a new dataset;
```

```
  %global seqnumber prefix section;
```

```
* If seqnumber macro variable already exists add 1 to it  
else create the macro variable as equal to 1;
```

```
  %if %symexist(seqnumber)=1 %then
```

```
    %let seqnumber=%sysfunc(putn(&seqnumber+1,z3.));
```

```
  %else
```

```
    %let seqnumber=1;
```

```
* Create prefix macro variable;
```

```
  %let prefix=&section._&seqnumber._;
```

```
%mend prefix;
```

Prefix Macros (cont.)

```
%macro prefix_wildcard(unique_dataset_name=);  
  %global &unique_dataset_name;  
  * Proc Sql to find the prefix of the unique_dataset_name  
  in the sashelp.vmember dataset;  
  proc sql noprint;  
    select memname  
    into :&unique_dataset_name  
    from sashelp.vmember  
    where libname = "WORK"  
    and  
    memname like  
  upcase('%' || "&unique_dataset_name");  
  quit;  
%mend prefix_wildcard;
```

Macros in Practice

```
%let section = A;
```

```
%prefix;
```

```
data &prefix.sdtm_input_1;
```

```
usubjid=1;
```

```
run;
```

```
%prefix_wildcard(unique_dataset_name = sdtm_input_1);
```

```
%prefix;
```

```
data &prefix.first_data_step;
```

```
set &sdtm_input_1;
```

```
run;
```

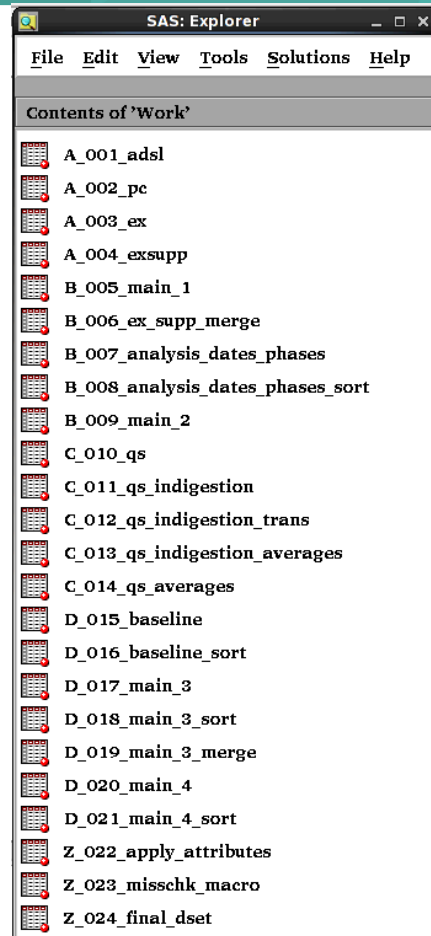
```
%prefix_wildcard(unique_dataset_name = first_data_step);
```

```
%prefix;
```

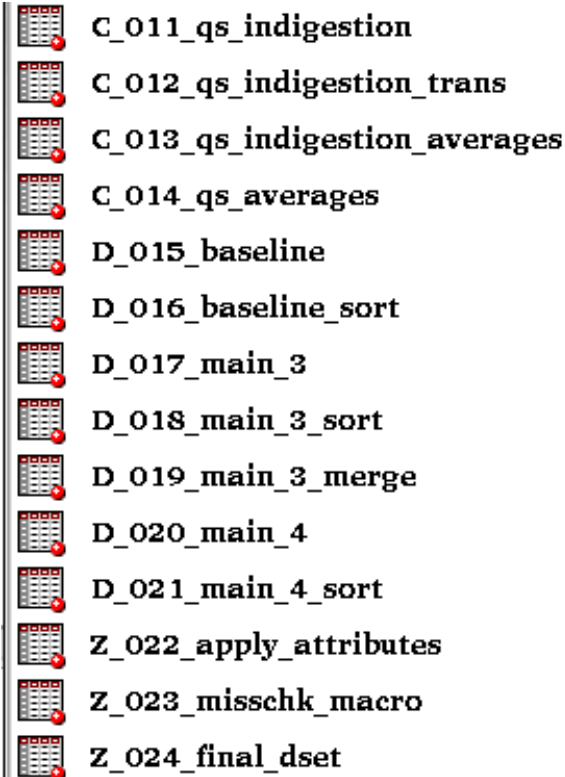
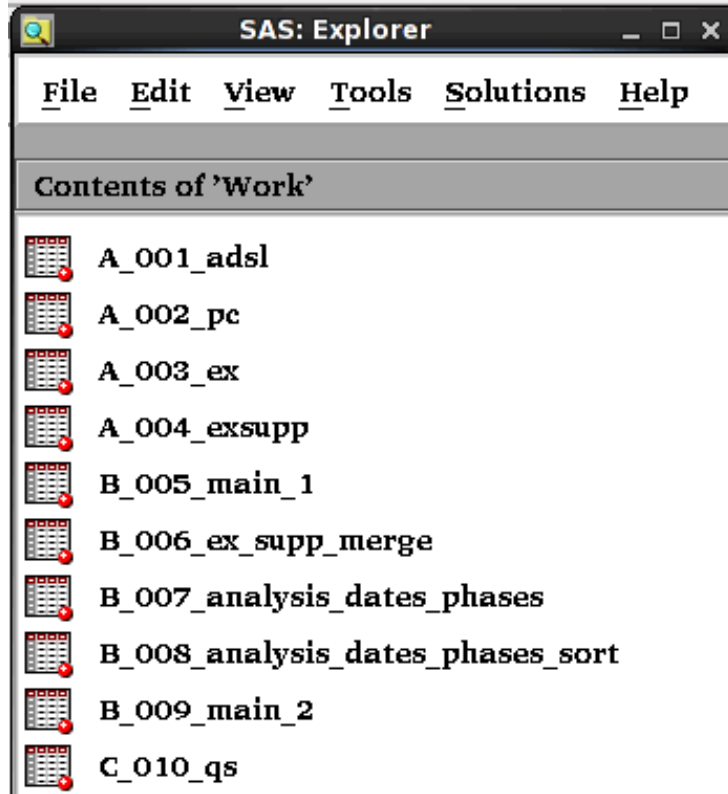
```
proc sort data=&first_data_step out=&prefix.first_data_step_sort;
```

```
by usubjid;
```

```
run;
```



Macros in Practice (cont.)



Summary

- A programmer should approach such programs well prepared
- GPP is essential and will always need to be considered first
- Sectioning and commenting are essential
- Macros can be utilised to smooth over this process
- Common issues can be pre-emptively countered and prepared for

Questions

