

# No Compromise on QUALITY: Coding tips and Macros to improve Efficiency

**Akhil Vijayan**

**Manager - Statistical Programming, Genpro Research**

**[akhil.vijayan@genproresearch.com](mailto:akhil.vijayan@genproresearch.com)**

**Limna Salim**

**India Head - Biometrics & Operations, Genpro Research**

**[limna.salim@genproresearch.com](mailto:limna.salim@genproresearch.com)**

**Roshan Stanly**

**Manager - Clinical Data Analytics, Genpro Research**

**[roshan.stanly@genproresearch.com](mailto:roshan.stanly@genproresearch.com)**

# Background

Quality and Time are the two major factors that define efficient clinical research and programming reusability is favored by many organizations to optimize time.

- Possible Issues
  - Truncations
  - Un noticed data issues
  - Temporary approaches.
  - Etc.

# User-Defined Checks

Quality assurance methods adopted by the organization.

| Methods                | Implemented |
|------------------------|-------------|
| Avoid Hardcoding       | ✓           |
| Standard Programming   | ✓           |
| Log Check              | ✓           |
| Validation programming | ✓           |
| Use of standard Macros | ✓           |
| Custom Checks          | ?           |

# Advantage

- Ensure the quality of data while reusing the program
- Helps to identify data issues
- Manage temporary Programming approaches
- Enable self QC of data

# Example

```
15
16 data adae4;
17     set adae3; *Where ADAE4 is a dataset which has TFL related transformations;
18     *Based on sponsor confirmation, when AAFL=Y and BBFL is missing then set BBFL to Y;
19     if "&studyid"='XYZ-001' then do;
20         if cmiss(bbfl) and aafl="Y" then bbfl="Y";
21     end;
22     else put "WAR" "NING: Study specific approach is copied from &studyid study";
23 run;
```

WARNING: Study specific approach is copied from ABC-001 study

NOTE: There were 1 observations read from the data set WORK.ADAE3.

NOTE: The data set WORK.ADAE4 has 1 observations and 3 variables.

NOTE: DATA statement used (Total process time):

real time 0.02 seconds

cpu time 0.01 seconds

# Programming Approaches

- Experience is your best guide
- Implement your experience in coding
- Think about the possible issues

# Example

```
31
32   proc transpose data=ae out=ae_t name=qnam label=qlabel;
33       var aeterm1 aeterm2 aeterm3 aeterm4 aeterm5 actoth;
ERROR: Variable AETERM4 not found.
ERROR: Variable AETERM5 not found.
34       by studyid domain usubjid aeseq;
35   run;
```

NOTE: The SAS System stopped processing this step because of errors.

WARNING: The data set WORK.AE\_T may be incomplete. When this step was stopped there were 0 observations and 0 variables.

WARNING: Data set WORK.AE\_T was not replaced because this step was stopped.

NOTE: PROCEDURE TRANSPOSE used (Total process time):

|           |              |
|-----------|--------------|
| real time | 0.01 seconds |
| cpu time  | 0.01 seconds |

```
36                               *OR;
```

```
37   proc transpose data=ae out=ae_t name=qnam label=qlabel;
38       var aeterm1-aeterm5 actoth;
ERROR: Variable AETERM5 not found.
39       by studyid domain usubjid aeseq;
40   run;
```

NOTE: The SAS System stopped processing this step because of errors.

WARNING: The data set WORK.AE\_T may be incomplete. When this step was stopped there were 0 observations and 0 variables.

WARNING: Data set WORK.AE\_T was not replaced because this step was stopped.

NOTE: PROCEDURE TRANSPOSE used (Total process time):

|           |              |
|-----------|--------------|
| real time | 0.01 seconds |
| cpu time  | 0.01 seconds |

# Macros that Aid in Reusability

- %**Varidentif** Macros to ensure that all variable lengths over 200 are properly mapped
  - As per SDTM standard user can populate 200 characters in free text variables, rest will be mapped into SUPP.
  - To ensure that all variable with a length of 200 or more is successfully mapped

## %Varidentif

- The macro will identify the variables with a length longer than the specified length and output the list.
- The programmer/lead need to check the variables against the specification,
  - Populate value 'Y' in SDTMVAR column if the variable is correctly mapped.
  - If the variable is [NOT SUBMITTED], populate value 'Y' in RMVDVAR column.
- Macro will read the data as mentioned above if it is existed and use the available information

# Macros that Aid in Reusability

- Macros for formats: **%autoformats**
  - To avoid hidden risk
  - Automate formats from specification
  - Include custom errors

# %autoformats

- Input: Codelist tab from SDTM/ADaM specification
- Include additional column for raw data values
  - Which will be used to create format
  - Suggested to add a user defined value for all other values which are not presented in codelist tab
- Data Type will be used to identify whether it is a character or numeric format



# THANK YOU

Any Questions ?