

Ready for Next Level SDTMs and ADaMs Compliance with End-to-End Processing?



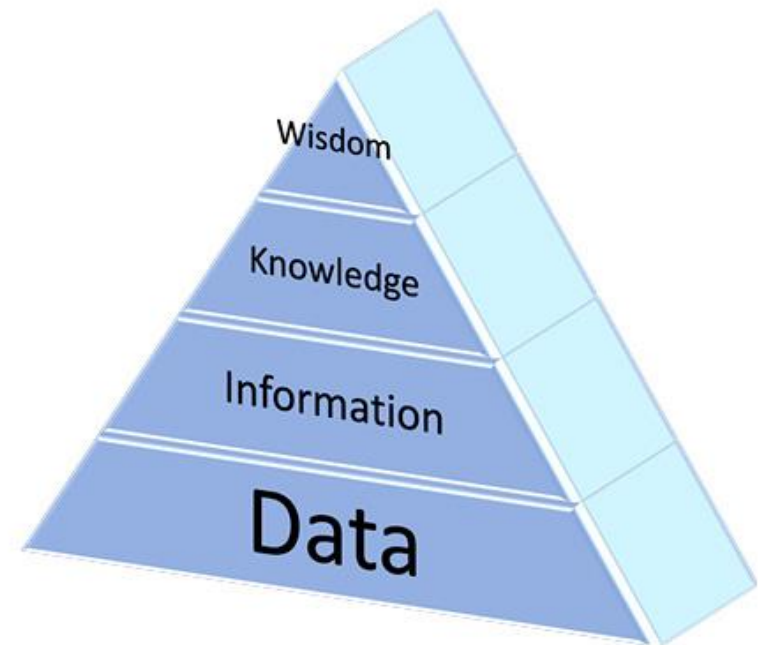
Sunil Gupta

GuptaProgramming@gmail.com
CDISC, SAS and R SME and Mentor,
Strategic Advisor to Verisian

Metadata Driven Transformation Process from Raw to SDTMs, ADaMs and Tables, Lists and Graphs



Same Single Version of the Truth to Answer Study Questions



What is Your SDTM Automation Role and Goals?

☐ **Statistical Programming Director – *Better Manage Timelines***

- Project Management, Time, Budget, Compliance, Metadata

☐ **Statistical Programmer – *Better Manage Submission Process***

- SDTMs / ADaMs / Define.xml, SDRG / ADRG, Mapping, Specifications

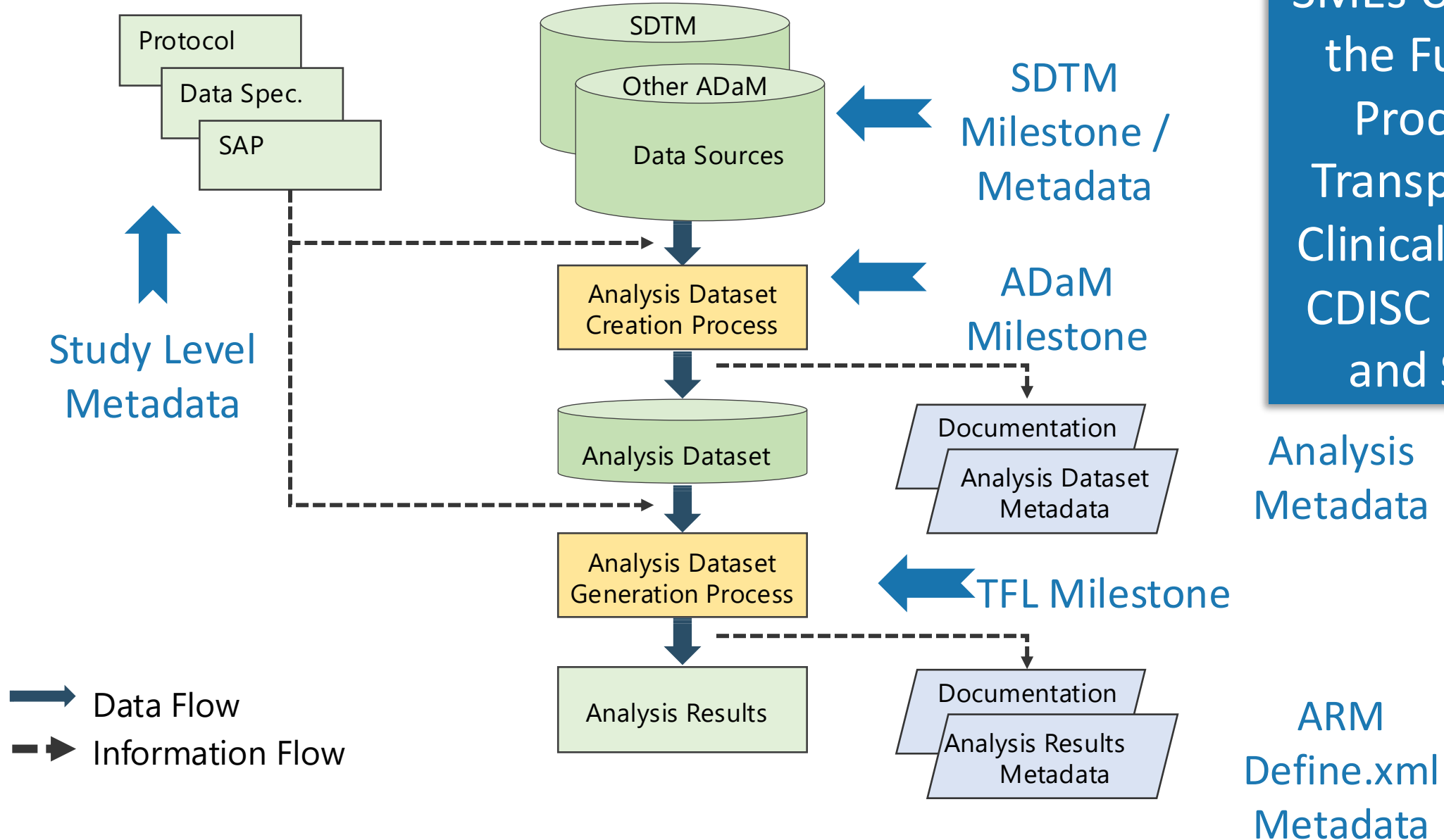
☐ **Translational Scientist – *Better Understand Safety Data Issues***

- EDC / Biomarker Raw Staging and Data Processing, Ingestion, Curation, Harmonization

☐ **Statistician – *Better Explore and Understand Study Conclusions***

- Analysis, Views, Ad-hoc, Tables, Listings and Figures, SAP, Data Meaning / Exploration, TLFs

CDISC Standards / Metadata



SMEs Understand
the Full Scope,
Process and
Transparency of
Clinical Data with
CDISC Standards
and Systems

Study integrity score

Analysis traceability

Code traceability

Specification Validation

Code Validation

Log Validation

Results Validation

Statistics validation

Data traceability

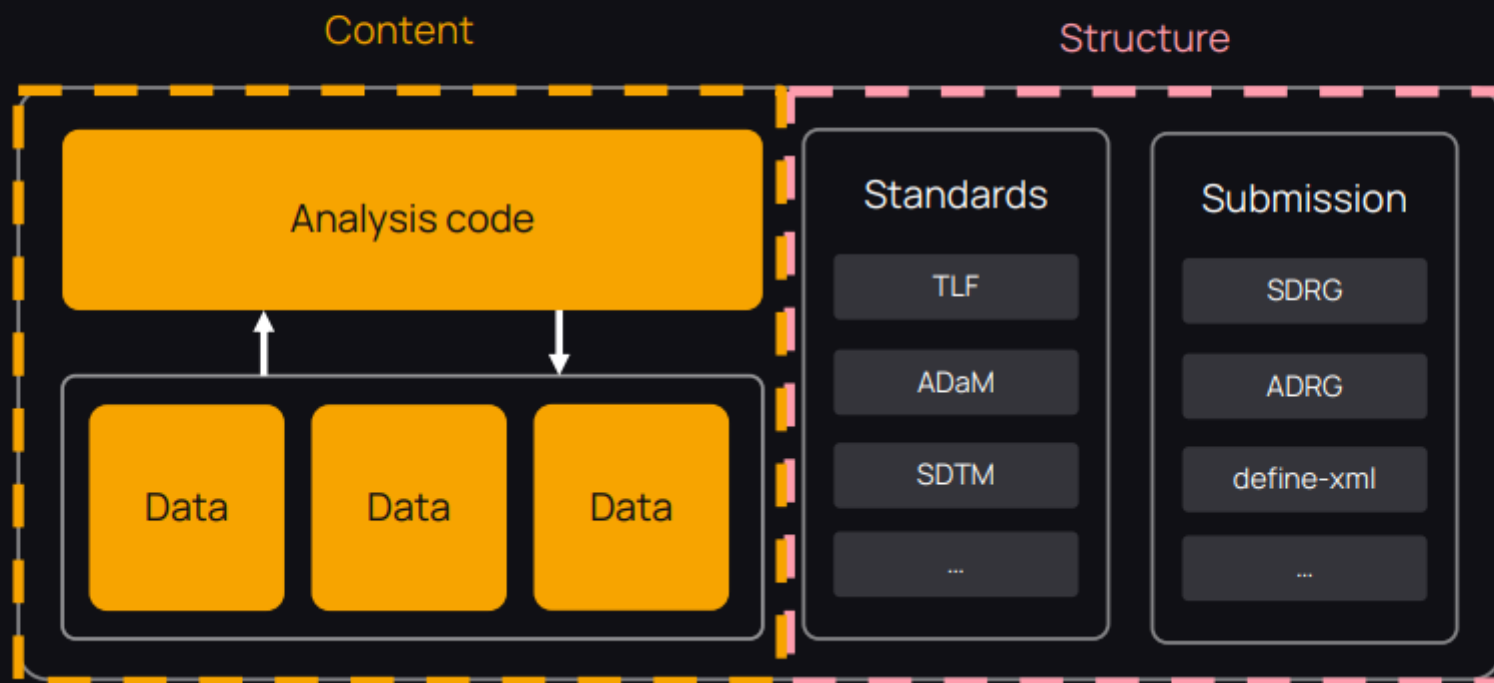
Data QC

Outliers

Distributions

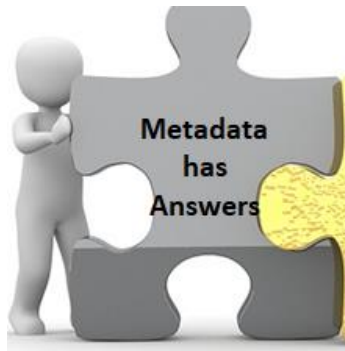
...

The **source of truth** of **content** of a trial is its code and data



SAS: Extract intelligence information from metadata and macro processing

Data-Driven process is automatic, quality controlled, transparent and saves time!



INPUT:
**Directory
of Files (Excel
files, Datasets,
SAS Programs,
Log and Proc
Compare.lst)**

SAS Tools

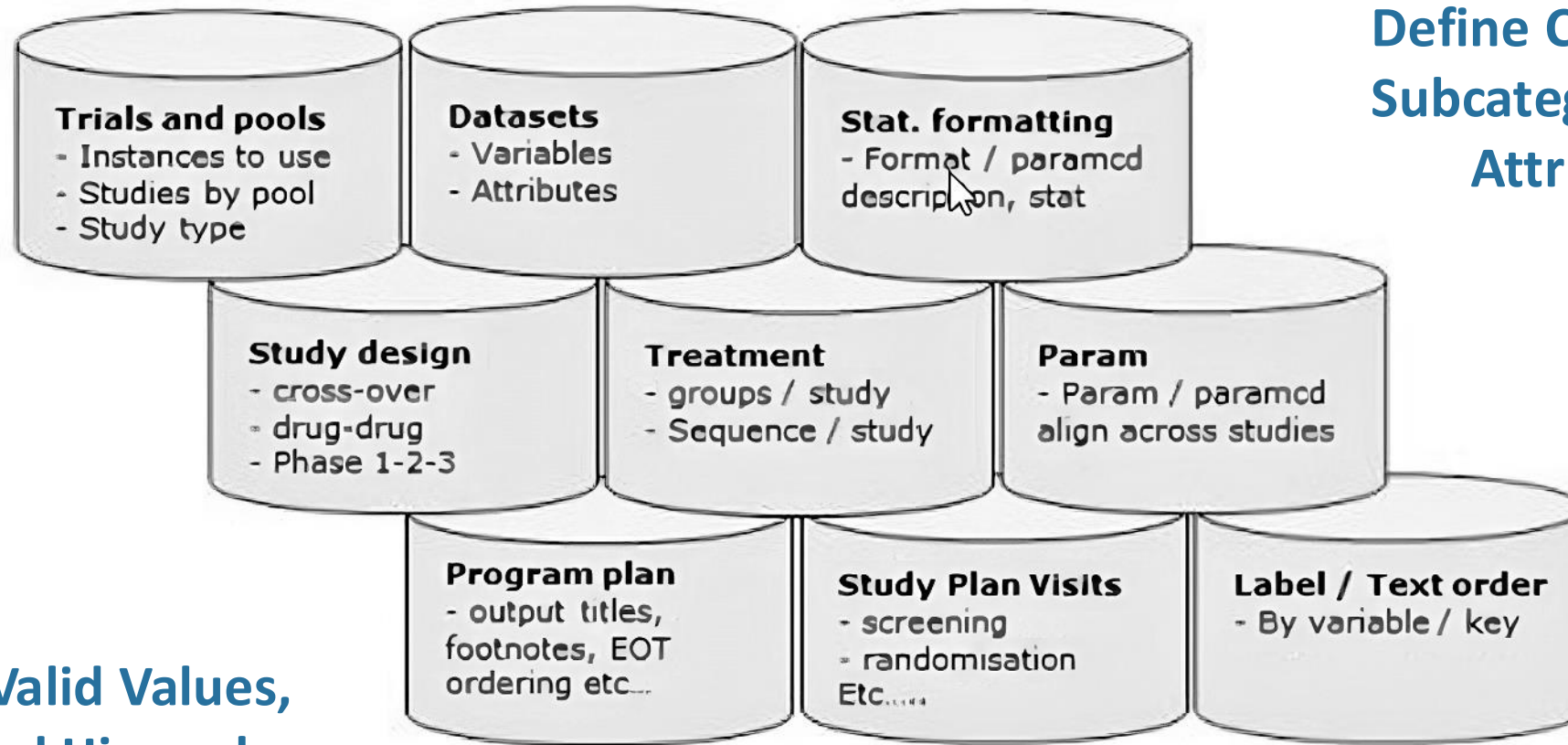
- Libnames
- Data Step
- SAS Macro Programming
- SAS & Dataset Functions
- Proc SQL
- Proc Compare
- Proc Means

OUTPUT: Metadata Attributes

- File pathname and names
- # of Files, Datetime stamps
- # and Type of variables
- Macro loop through all files
- Required datasets, variables, etc.
- Codelist dictionary, SDTM/ADaM attributes
- Maximum variable lengths
- Data cleaning & monitoring of valid variables and special characters
- Compare and contrast previous file
- Descriptive Statistics on categorical and continuous variables
- Search for ERRORS, WARNINGS or Notes in SAS Logs
- Search for QC differences in # of VARs, OBS, attributes and dups

Metadata Programming Concepts – We Define Rules and Process

Define Categories,
Subcategories and
Attributes



Define Valid Values,
Rules and Hierarchy



What if there is complete transparency between code, raw, SDTMs, ADaMs and TLGs with drill down features to confirm variable derivations?

Raw

pain.painbase

pain.pain3mo

pain.pain6mo

demographic.trt

SDTM

dm.arm

xp.xpdtc

xp.XPSTRESN

ADaM

ADSL.trt01p

ADSL.trt01pn

adef.ittfl

adef.trtpn

adef.crit1fl

Data

ADaM > adef > **TRTPN**

ADaM.TRTPN

Count by

Category



Category	Count
Y	60
N	55
Missing	5

Summary

Demographic Data Input (demographic.sas)

- Reads treatment data into the dataset source.demographic and labels the variable trt as "Treatment".

Demographic Transformation (DM.sas)

- Converts the trt variable into a formatted string arm using a format arm_demographic_trt.

Assign Treatment Codes (adsl.sas)

- Checks if arm is missing; if so, assigns it a value of 0.
- Assigns the value of arm to trt01p.
- Converts the formatted string trt01p to a numeric variable trt01pn using a format \$trt01pn.

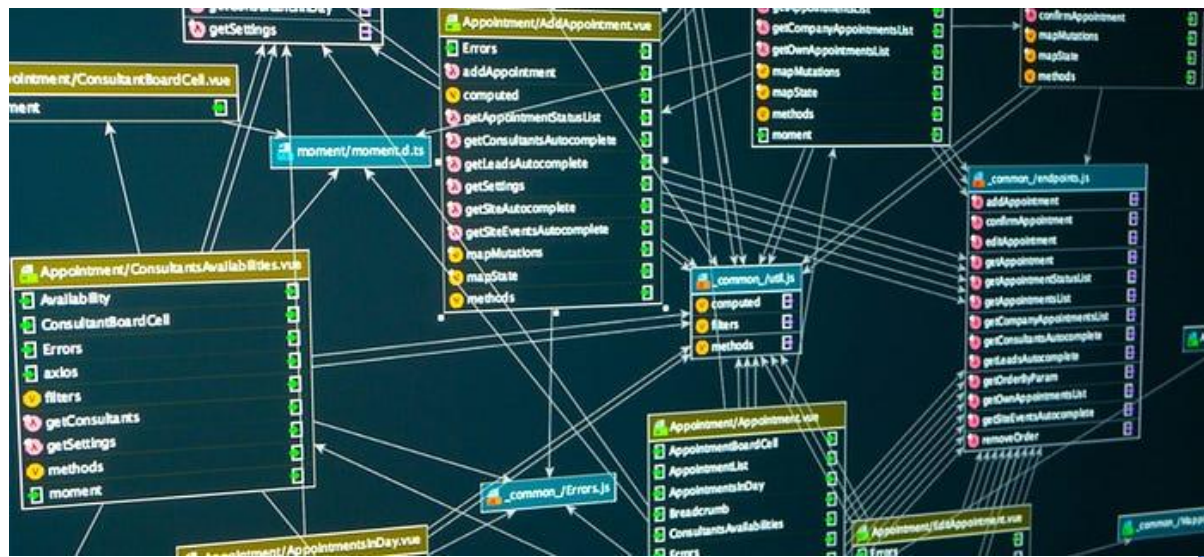
Rename Variable for Analysis (adef.sas)

- Renames the variable trt01pn to trtpn in the dataset adef to standardize the variable name for further analysis.

Derivation

(Macro-resolved)

```
1 * demographic.sas;
2 data source.demographic;
3   . label trt="Treatment";
4   input trt 5;
5   *...;
6 run;
7
8 * DM.sas;
9 data dm;
10  *...;
11  arm=put(trt, arm_demographic_trt.);
12  . *...;
13 run;
14
15 * adsl.sas;
16 data ADSL;
17  *...;
18  trt01p=ARM;
19  *...;
20  trt01pn=input(put(trt01p, $trt01pn.), best.);
21  *...;
22 run;
23
24 * adef.sas;
25 data adef;
26  *...;
27  rename trt01pn=trtpn;
28  *...;
29 run;
```



Pyramid of Proc SQL

Metadata

Dataset / Group Filters

Append / Join Datasets

Create / Delete Datasets / Views

Create / Drop Dataset / Group Variables

Dataset / Group Queries / Subqueries

Macro / Descriptive Statistics Variables

Update / Insert / Delete / Group / Order Records

- Clinical Data has Many Touchpoints
- Need a System to Monitor and Trace Data Updates

***What if** study queries are as easy as Google searches to get reliable and reproducible clinical study results?*

Question

How is AESTDTC derived?

Answer

AESTDTC is imported from *raw_cleaned.xlsx* into the *rawae* dataset with its original values.

In *EMPTY_AE*, *AESTDTC* is initialized as an empty string ("") with a length of 20 characters but does not affect the dataset since *EMPTY_AE* has 0 observations and acts as a structure placeholder.

When creating *ae_1*, *AESTDTC* is modified by trimming leading and trailing spaces and extracting the first 10 characters; this happens for each of the 134 observations in *rawae*.

The *final_ae* dataset is created by concatenating *EMPTY_AE* and *ae_2*. Since *EMPTY_AE* has 0 observations, *final_ae* inherits the modified *AESTDTC* from *ae_2* without additional alterations.

Evidence

```
1 * AE.sas;
2 proc import datafile="/.../raw_cleaned.xlsx" out=rawae dbms=XLSX replace;
3   getnames=yes;
4   sheet="raw_ae";
5 run;
6 * rawae has 134 observations;
7
8 * AE.sas;
9 data EMPTY_AE;
10  *...;
11  attrib AESTDTC label="Start Date/Time of Adverse Event" length=$20. ;
12  retain AESTDTC '' ;
13  *...;
14  if 0;
15 run;
16 * EMPTY_AE has 0 observations;
17
18 * AE.sas;
19 data ae_1;
20   set rawae;
21   *...;
22   AESTDTC=substr(left(trim(AESTDTC)), 1, 10);
23 run;
24 * ae_1 has 134 observations;
25
26 * AE.sas;
27 data ae_2;
28   set ae_1;
29   *...;
30 run;
31 * ae_2 has 134 observations;
32
33 * AE.sas;
34 data final_ae;
35   retain studyid domain usubjid aeseq aeterm aellt aellcd aedecod aeptcd
36   set EMPTY_AE ae_2;
37 run;
38 * final_ae has 134 observations;
```

What If:

SAS Programs

Clinical Studies

Deliverables

Reading 100's of
SAS Programs from
10's of
Programmers

Automatically
Track and Resolve
Needle in Haystack

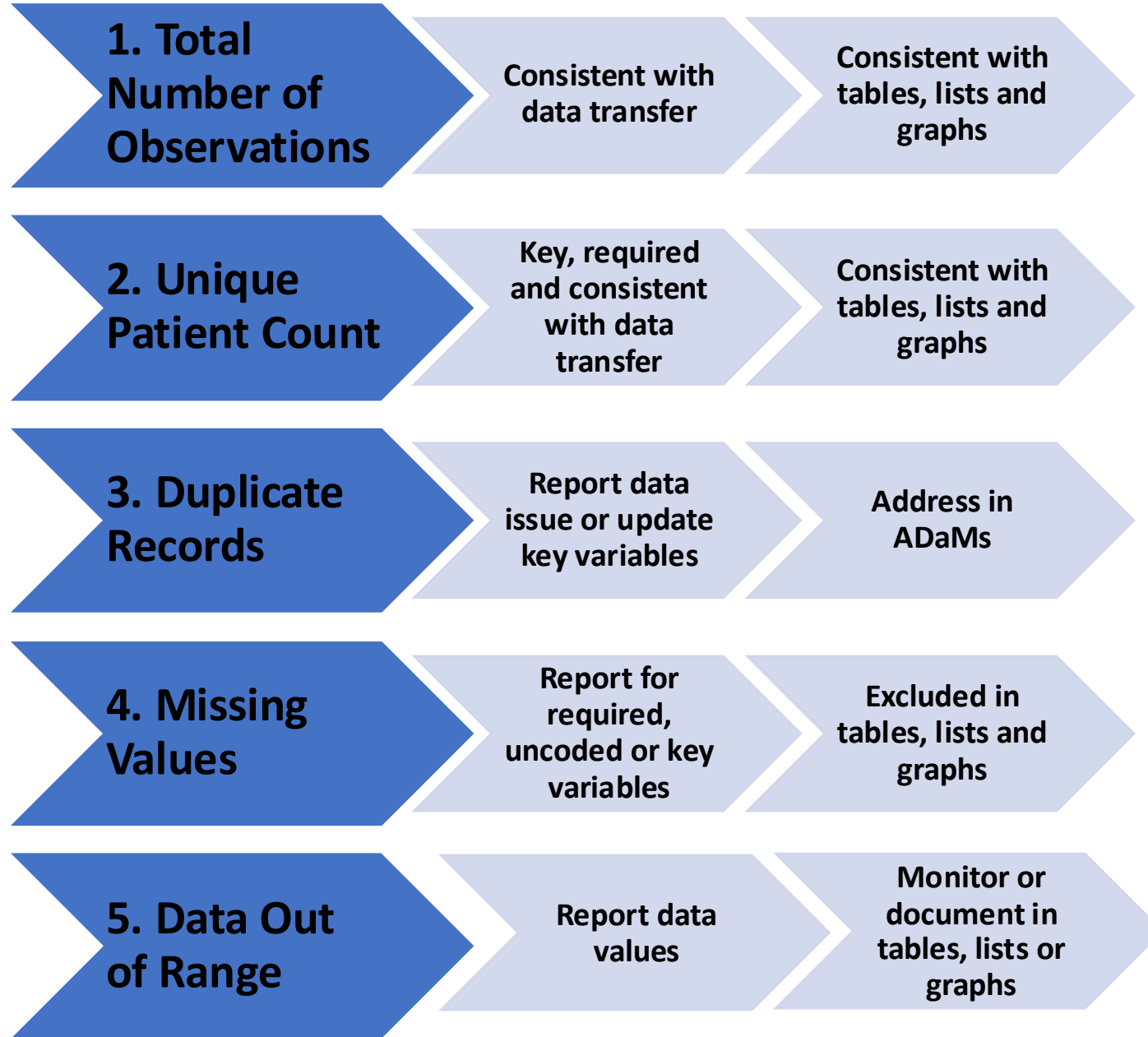
Transform SAS into
Process Flow
Charts

Input:
Program Index Sequence
file
100's SAS Program
10's Macros & 100's Logs

Issues in SAS Logs:
Errors, Warnings and
Notes

10's Datasets, 10's Tables,
10's Lists, 10's Graphs,
etc.

Know Your Data Stats to Results: Raw, SDTMs, ADaMs and TLGs



What If:

Build Systems to Improve the Process with Dashboards

**Visualize and Track Data
Issues**

from Raw to SDTMs and
ADaMs

**Point-n-Click Process
Flow**

Image to Review Data
Step/Procedure Snippets

**Monitor and Reduce SAS
Log Issues**

(Errors, Warnings and
Notes)

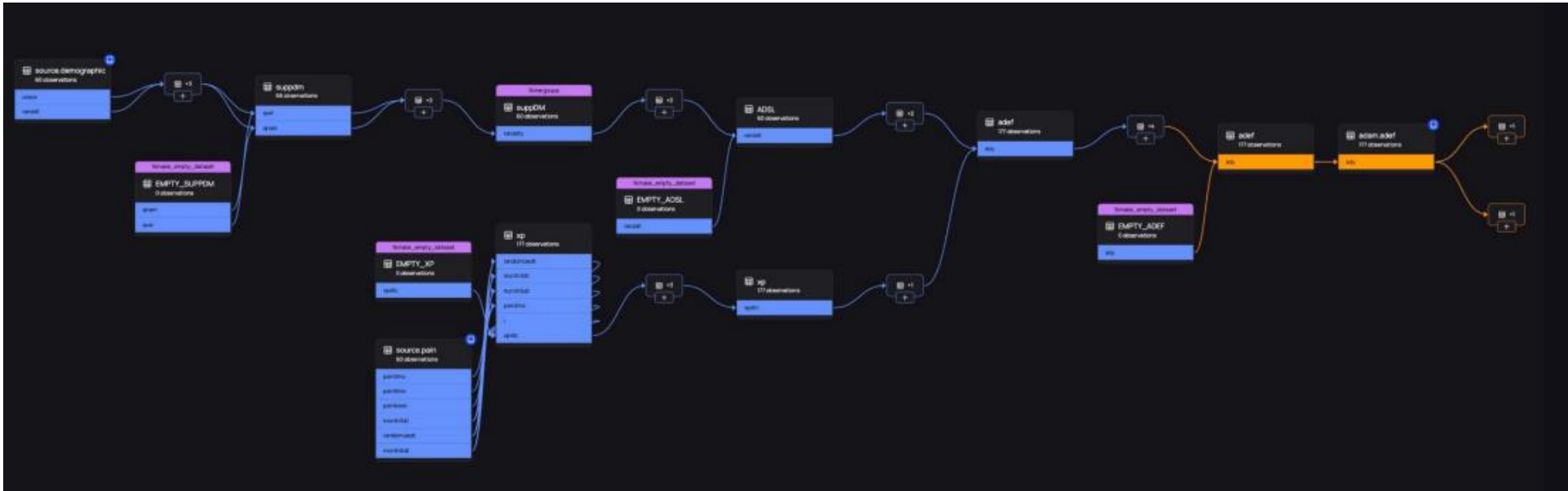
**Impact Analysis/Correct
Code**

to Automatically Update
SAS Programs

**Automatically create Un-
macro Program Version**

for easier debugging

What if code can read as metadata to 'visualize process' and 'automatically link programs by data'?



What If: Quickly Identify and Resolve Data AND Programming Issues

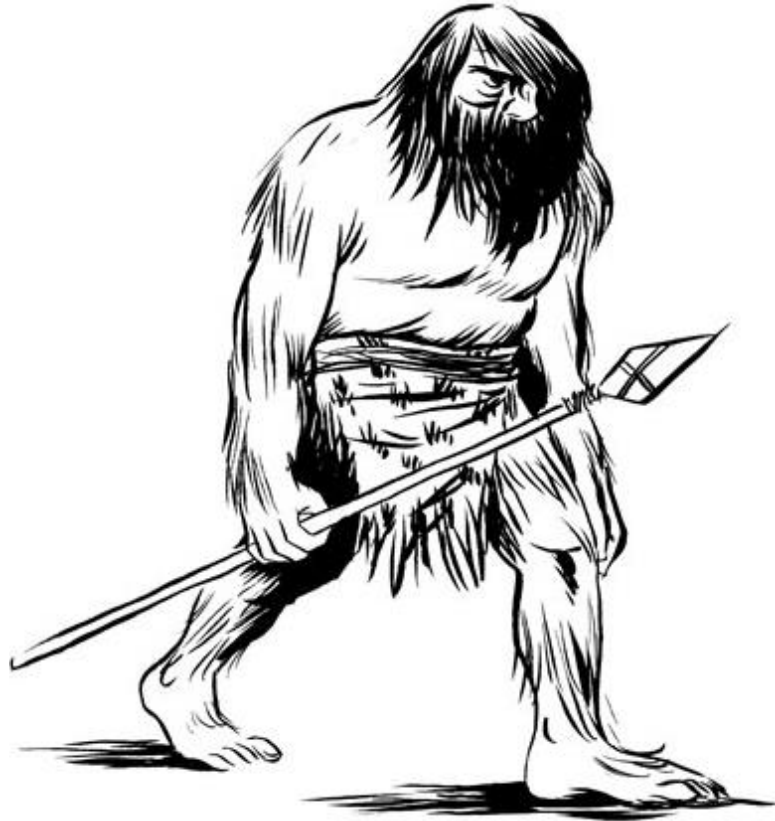
Clinical Study Issues

- Primary and Secondary Endpoints
- Data-cutoff Date
- AE Count Denominators

Track and Resolve SAS Program Issues

- Missing Program Header Information
- Intermediate instead of Overwriting Datasets
- Dataset and variable naming conventions
- Explicit instead of implicit code and options
- Program Style and Indentation for Readability
- Single instead of Multiple Purpose Programs
- Simple instead of Complex Program Logic or Multiple Layer of Macro Calls
- Separate Program Sections with Comments
- Meaningful Footnotes
- Missing Macro in Programs
- Keyword Macro Definition Programs

A World without Code Traceability-driven AI systems



What You can Expect *Without* Dashboards:

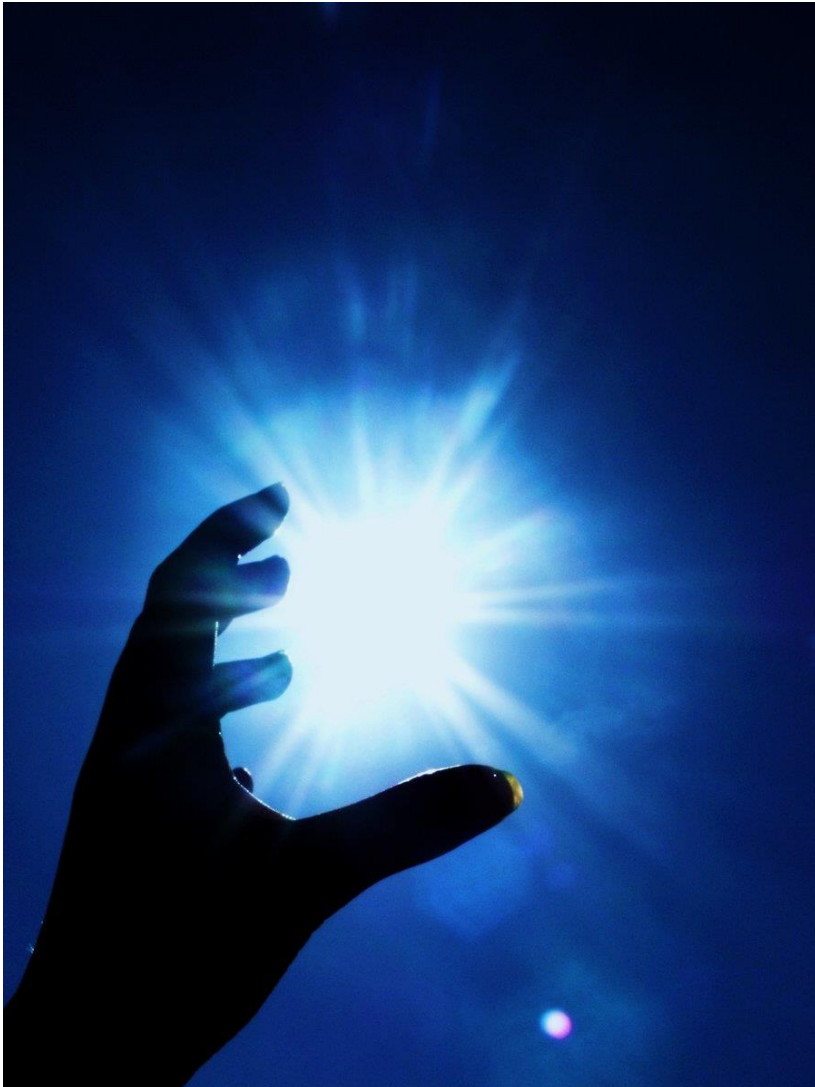
- ✓ Hundreds of hours to scan across tens of programs and macros to confirm variable derivations
- ✓ Hundreds of hours to cross-reference ADaMs with Tables, Lists and Graphs to assure consistency
- ✓ Hundreds of hours to scan across logs to address errors, warnings and notes
- ✓ No method to identify, communicate and track data and program issues to resolution

Transformations Across the Decades: *Plan for the Future Now!*



*For the first time, Gen-AI
Technology traces code
at the dataset, variable
and value-level into
metadata which
transform to clinical
study answers*

Summary: Solutions Exist so You *Can* Reach for the Stars



- ✓ Scan Tools help Find Needles in the Haystack for better CDISC Compliance
- ✓ Metadata Tools Cross-reference Touchpoints: Clinical Data > Mapping > Coding > Tables
- ✓ Dashboards Empower Programmers to write 'Smarter' and 'Qced' Programs

Sunil Gupta, GuptaProgramming@gmail.com
CDISC, SAS and R SME and Mentor, Strategic
Advisor to Verisian