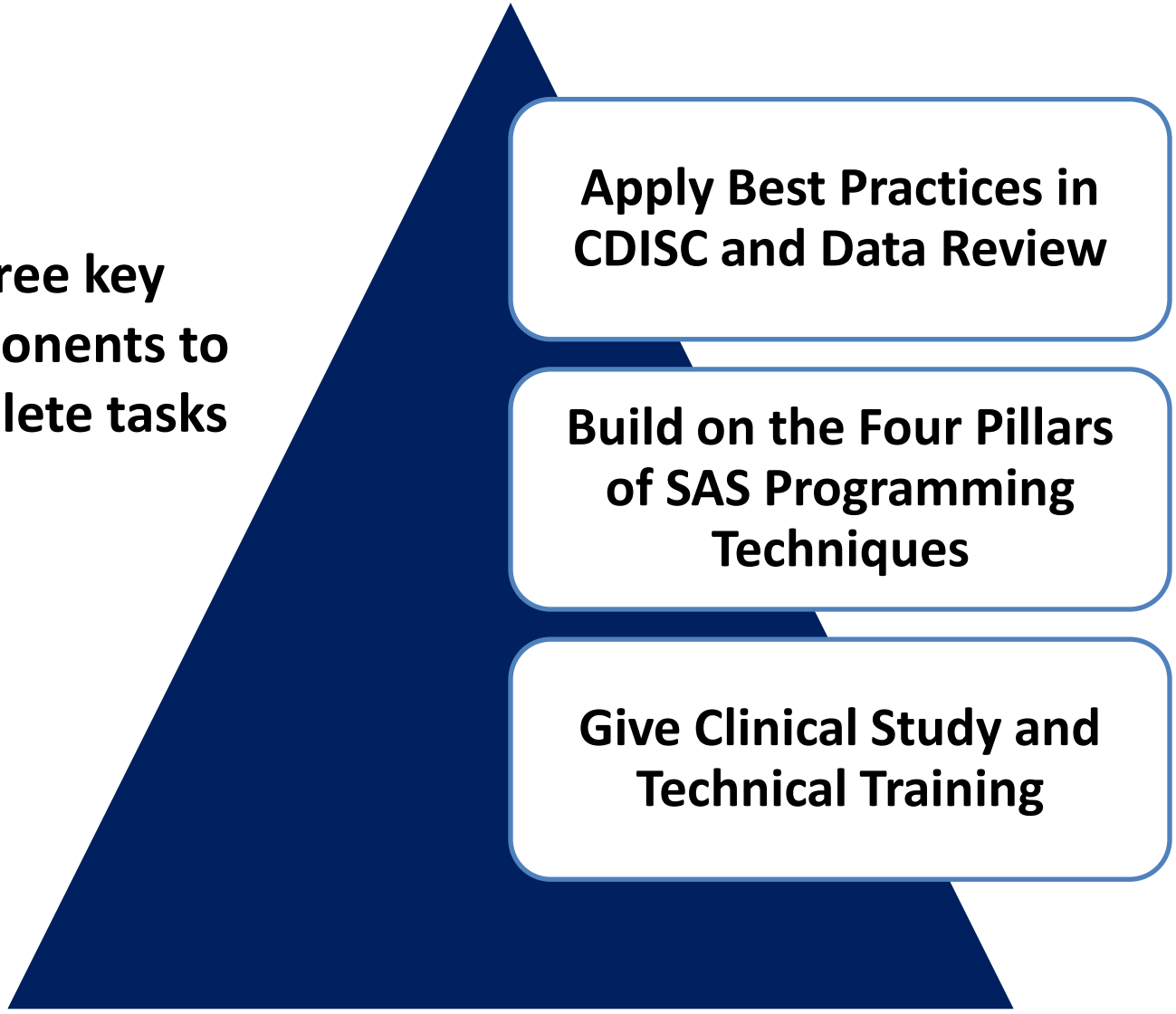
Best of Both Worlds: Better Management and Higher Quality Deliverables

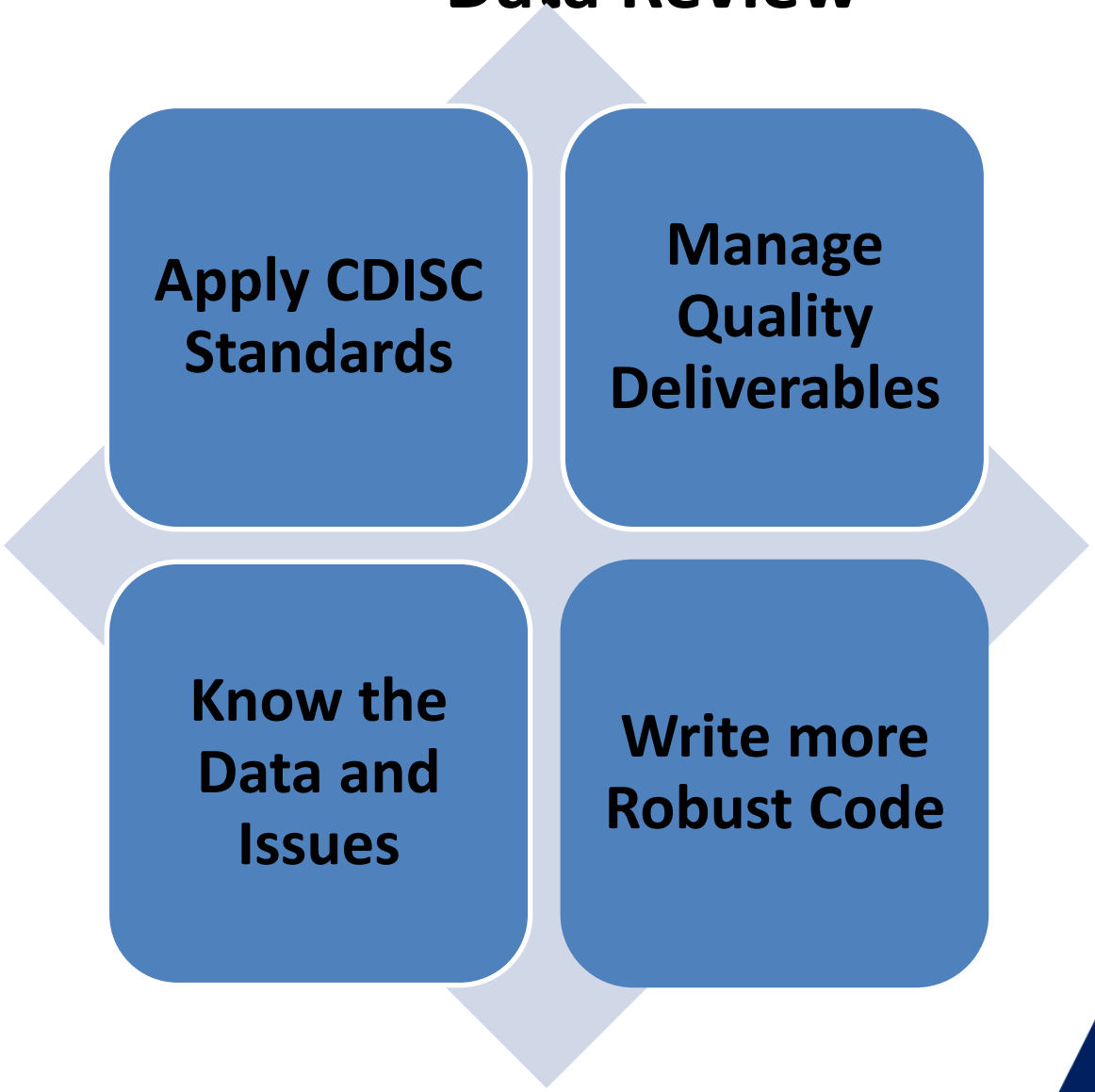
Sunil Gupta
Associate Director, Statistical Programming
Cytel

Better Management & Higher Quality Deliverables

Three key components to complete tasks



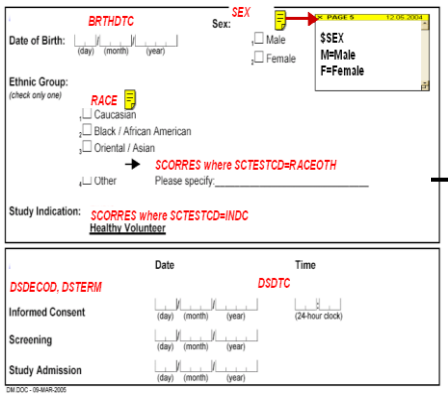
Apply Best Practices in CDISC and Data Review



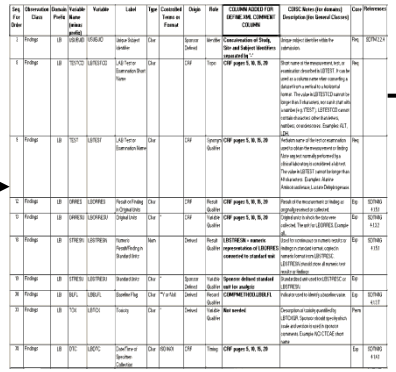
Apply CDISC Standards: SDTMs/ADaMs Process Flow

e-submission deliverables

Annotated CRF



SDTM /ADaM Specifications



(dataset templates)

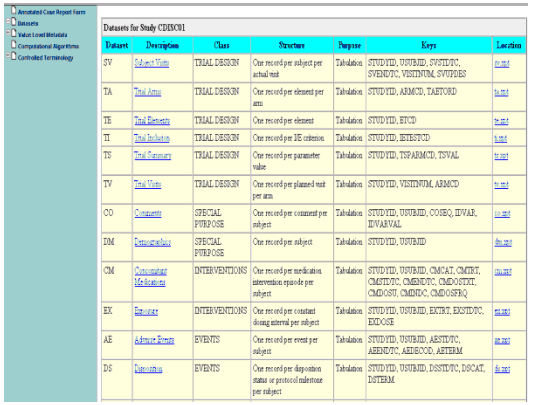
(metadata source for SDTMs, ADaMs, and define.xml)

Mapping programs

SDTMs

ADaMs

define.xml (SDTMs/ADaMs)



Tables, Lists and Figures

Summary of table of Creatine at baseline
ADLB.AVISIT='BASELINE'
(Per Protocol Population)
ADLB.PPROTFL='Y'

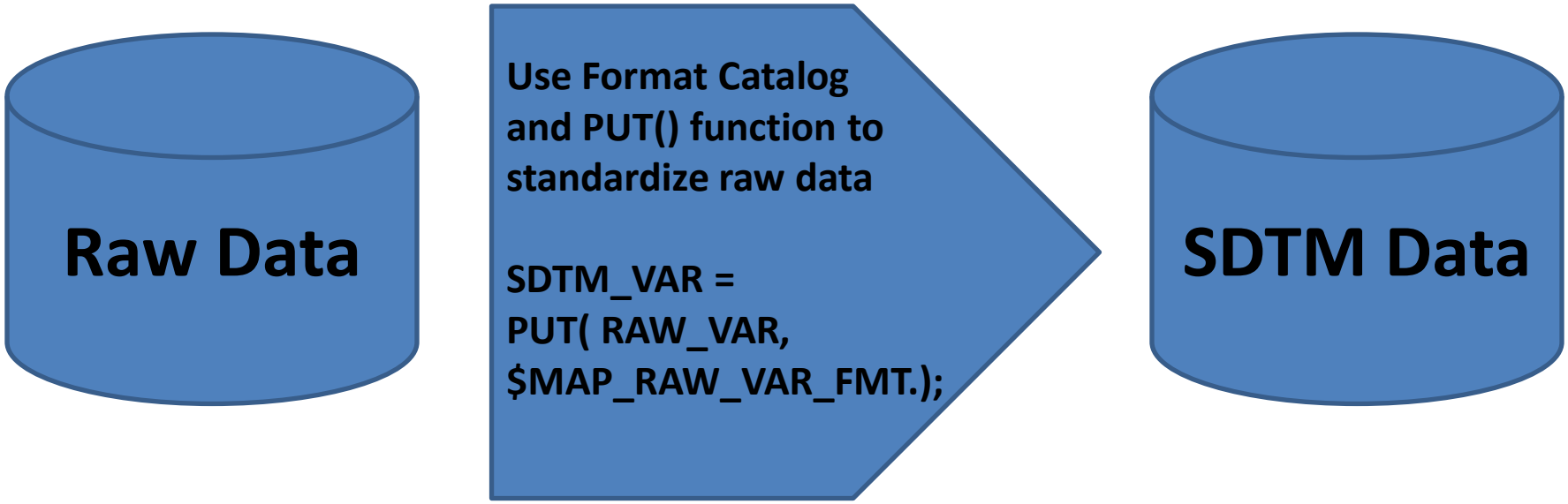
ADLB.PARAMCD	Group 1: Treatment 1 (N=xxx) ADLB.TRTAN = 1		Group 2: Placebo (N=xxx) ADLB.TRTAN = 2	
	n	Observed Mean Value	n	Observed Mean Value
Creatine WHERE PARAMCD='CREAT'	COUNT(ADLB.AVAL)	MEAN(ADLB.AVAL)	COUNT(ADLB.AVAL)	MEAN(ADLB.AVAL)
Log of Creatine WHERE PARAMCD='L10CREAT'				

Apply CDISC Standards

Converting variable types—use PUT() or INPUT()?

Sunil Gupta | MAY 1, 2015

How many times have you had a need to convert between variable types such as converting character to numeric or numeric to character? For example, what if you have a character variable with numeric values but you need to perform some calculations? Or, if you have a numeric variable but you need to concatenate it to a character variable? If you are like most SAS programmers, you need to use PUT() and INPUT() at least once to complete these tasks.



Manage Quality Deliverables: Communicate / Organize / Document

Know the Data and Issues

- ✓ Data Edit Checks
- ✓ Reconciliation Reports Review
- ✓ Complex Variable Derivation Check Review
- ✓ Non-Missing and Missing Summary Report
- ✓ Challenges in assigning analysis visit windows in LB

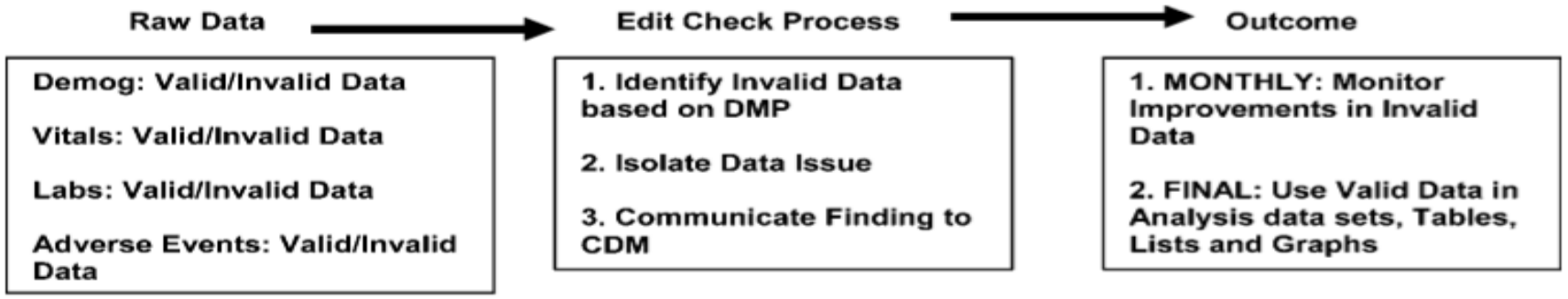


Figure 1 Clinical data process flow

Table 1 Types of data issues

Type of data issue	Brief description
Acceptable values	Values are one of the valid values for variable
Character formats	Format of values within character variables are as expected, ex. XXX-XXXX
Consistency across variables	Values are consistent across multiple variables
Consistency across datasets*	Values are consistent across multiple data sets
Non-duplicate records	Each record is unique and not duplicated
Protocol compliance rules	Study specific logic-based check to confirm data compliance, ex. lab conv
Range check	Values are within a specified range
Required value	Value is non-missing
Unique value	Values are unique

*May require extra programming step since most all edit check macros require single dataset.

Know the Data and Issues

Non-Missing and Missing Summary Report

BASEDT	PEAKDT	PEAKVAL	Frequency	Percent	Cumulative Frequency	Cumulative Percent
NON-MISSING	MISSING	MISSING	5	2.25	5	2.25
NON-MISSING	NON-MISSING	NON-MISSING	217	97.75	222	100.00

Challenges in assigning analysis visit windows in LB

By Group Variables	Date Exists?	Time Exists?	Value Exists?	Action
Non duplicate record	Y	Y	Y	Keep record
Duplicate records	Same	Same or Y/Missing Or Both Missing	Y/Y or Y/Missing	Take mean value and keep only one record
Duplicate records	Same	Not Same and Non-missing	Y/Y	Take most recent time record value
Duplicate records	Same	Missing	Y/Missing	Take non-missing record
Duplicate records	Same	Missing	Missing	Keep only one re

Write more Robust Code

SAS Tip: Map VISIT to AVISIT

1) Apply Visit Windows

Assign target visit and select one visit closest to target visit

2) Adjust for Overlap Days

Instead of mutually exclusive conditions, apply IF-THEN logic to use all data

Analysis	Visit Target Day Window
Baseline	-5 ADY in <= -5
DAY 0	0 ADY in 0
DAY 1	1 ADY in 1, **2**
DAY 3	3 ADY in **2**, 3, **4**
DAY 5	5 ADY in **4**, 5, **6**
DAY 7	7 ADY in **6**, 7, 8, 9, **10**
Week 2	14 ADY in [**10**, 21]
Week 4	28 ADY in [22, 42]

Non-mutually exclusive conditions enable greater flexibility to assign analysis visits but also present programming challenges

SAS Tip: Map VISIT to AVISIT

Key Program Steps
1. Apply visit window for non-missing aval based on priority. a. Apply window below, notice duplicates - ADY = 4 (first for DAY 3, second for DAY 5), and ADY = 6 (first for DAY 5, second for DAY 7). b. Select closest to target if equal distance since as AVISIT = 'DAY 5' for ADY = 4 instead of ADY = 6. 2. If missing AVISIT because used for another AVISIT then make sure to assign missing AVISIT.
3. Sort by usubjid paramc avisit difference with target date to get the closest to target date and ady.
4. Assign all non-closest to missing avisit.

SAS Tip: Map VISIT to AVISIT

```
* Assign AVISIT and TARGET based on ADY;

data adbmk7;
length avisit $20. target awtdiff 4.;
set adbmk6;

  if aval = . or ady = . then do; avisit = '';  target=.; end;
  else if BASE = 'Y' then do; AVISIT = 'BASELINE'; target=-5; end;
  else if ady = 0 then do; avisit='DAY 0'; target=0; end;
  else if ady in (1, 2) then do; avisit='DAY 1'; target=1; end;
  else if ady in (2, 3, 4) then do; avisit='DAY 3'; target=3; end;
  else if ady in (4, 5, 6) then do; avisit='DAY 5'; target=5; end;
  else if ady in (6, 7, 8, 9, 10) then do; avisit='DAY 7'; target=7; end;
  else if 10 <= ady <= 21 then do; avisit='WEEK 2'; target=14; end;
  else if 22 <= ady <= 42 then do; avisit='WEEK 4'; target=28; end;
  else avisit='';

  if nmiss(target, ady)=0 then awtdiff=abs(target - ady); else awtdiff=.;
run;
```

SAS Tip: Map VISIT to AVISIT

```
* Sort by usubjid paramc avisit difference with target date to get the closest to target
date and ady;

proc sort data=adbmk7;
  by usubjid paramcd aperiod avisit awtdiff ady;
run;

* Assign all non-closest to missing avisit;

data adbmk8;
 set adbmk7;
 by usubjid paramcd aperiod avisit awtdiff ady;

 if ^first.avisit then avisit='';

 if AVISIT > '' and AVAL > . then aucdy = input(put(avisit, $aucbmk.), best.);

run;
```

Write more Robust Code

SAS Tip: Fuzzy Join to Derive Ncart

QPPB	WBC, MONOLE and LYMLE	Ncart = 1000*(WBC)* (MONOLE + LYMLE)/ 100*QPPB/100
Date 1	All non-missing on Date 1	Derive Ncart
Date 1	All non-missing and shortest distance on Date 2 within **+/- 3 days if LBDY <= 35**	Derive Ncart
Date 1	All non-missing and shortest distance on Date 2 within **+/- 10 days if LBDY > 35**	Derive Ncart
Date 1	Individual non-missing and equal distance than take **average of WBC, MONOLE and LYMLE**	Derive Ncart

SAS Tip: Fuzzy Join to Derive Ncart

```
data lb_newa1;
set lb_new;
by usubjid aperiod lbdt lbdy;

 * 1 - create lbdymin lbdymax var that define windows based on +/- 3 days if
lbdy <= 35 and +/- 10 days if lbdy > 35;
if lbdy <= 35 then do; lbdymin = lbdy - 3;
   lbdymax = lbdy + 3; end;
else if lbdy > 35 then do; lbdymin = lbdy - 10;
   lbdymax = lbdy + 10; end;

rename monole=monole_i wbc=wbc_i lymle=lymle_i;
 * 2 - identify non-missing records for monole, wbc, lymle and nmcyt;
   if nmiss(monole, wbc, lymle) = 0 or wbc = 0 then nmcyt=1;
 * 3 - filter only non-missing records;
 if nmcyt=1;
 drop QPPB; run;
```

continued...

SAS Tip: Fuzzy Join to Derive Ncart

*** 4 proc sql with overlap of lbdy and lbdymin and lbdymax;**
*** 5 join within +/- 3 days if lbdy <= 35 and +/- 10 days if lbdy > 35;**
*** 6 select only the closest to original date;**

```
proc sql;
  create table lb_newa2 as
  select a.*, c.monole_i, c.wbc_i, c.lymle_i, c.lbdymin, c.lbdymax, abs(c.lbdy -
a.lbdy) as lbdydif

  from lb_new as a left join lb_newa1 as c

  on (a.usubjid = c.usubjid and a.aperiod = c.aperiod) and
    ( (c.lbdymin <= a.lbdy <= c.lbdymax) )
  group by a.usubjid, a.aperiod, a.lbdt, a.lbdy having calculated lbdydif =
min(calculated lbdydif);

quit;
```

continued...

SAS Tip: Fuzzy Join to Derive Ncart

```
* 7 Average if same distance from original date;

proc sql;
  create table lb_newa3 as
  select usubjid, aperiod, lbdt, lbdy, mean(monole_i) as monole_i2,
mean(wbc_i) as wbc_i2, mean(lymle_i) aslymle_i2, min(qppb) as qppb1
  from lb_newa2 group by usubjid, aperiod, lbdt, lbdy;
quit;

* 8 Check for duplicate same distance from target;

proc sort data =
lb_newa3 nodupkey out=lb_newa3_nodup dupout=lb_newa2_dups;
by usubjid aperiod lbdt lbdy;
run;
```

continued...

SAS Tip: Fuzzy Join to Derive Ncart

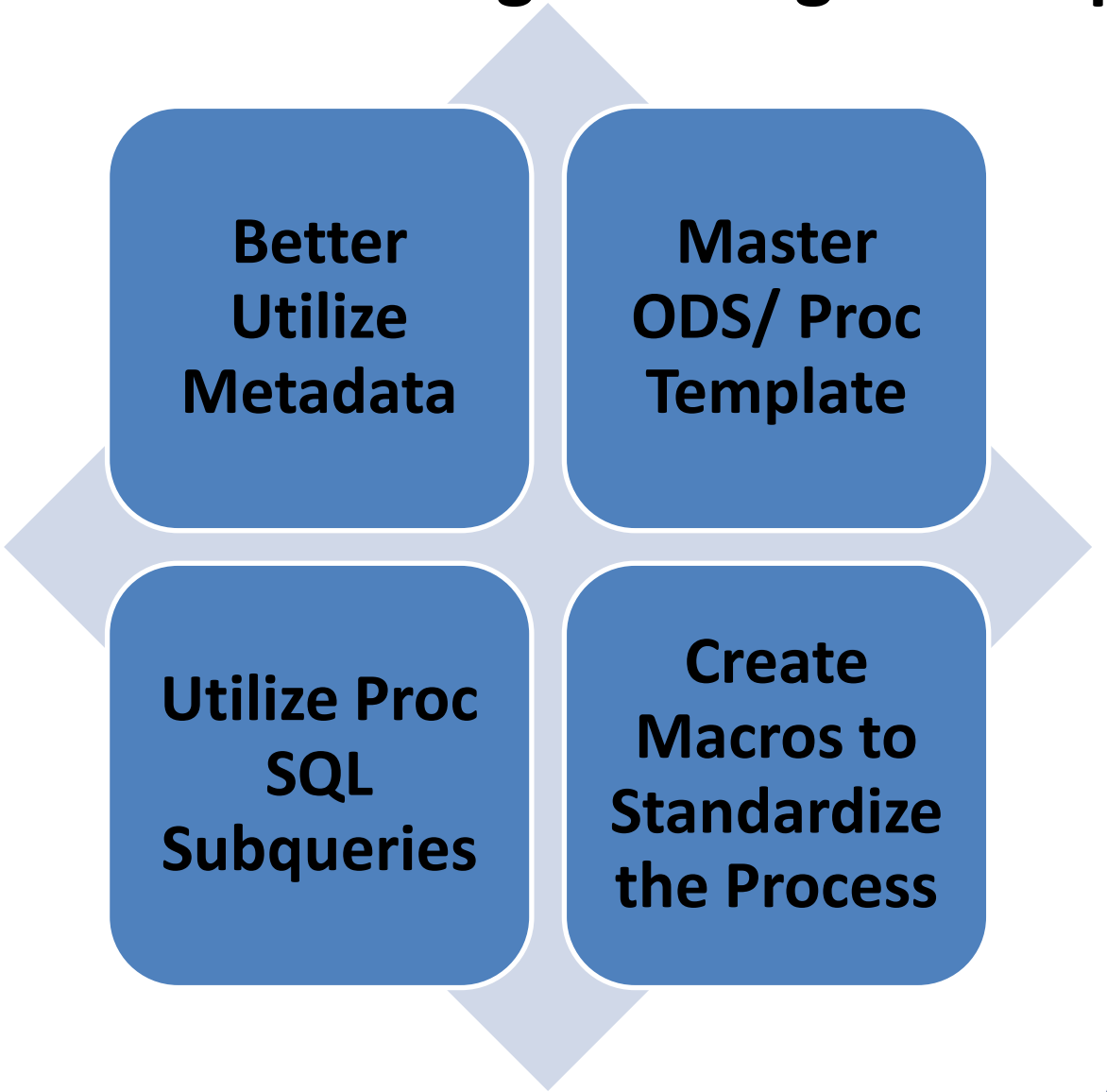
```
proc sort data=lb_newa3;
  by usubjid aperiod lbdt lbdy;
run;

data lb_newb;
  merge lb_new lb_newa3;
  by usubjid aperiod lbdt lbdy;

  * Replace missing values with fuzzy join values based on windows;
  if lymle = . then lymle=lymle_i2;
  if monole = . then monole=monole_i2;
  if wbc = . then wbc=wbc_i2;

  drop lymle_i monole_i wbc_i lymle_p1 lymle_n1 monole_p1 monole_n1
wbc_p1 wbc_n1 lbdy_p1 lbdy_n1 impute;
run;
```

Build on the Four Pillars of SAS Programming Techniques



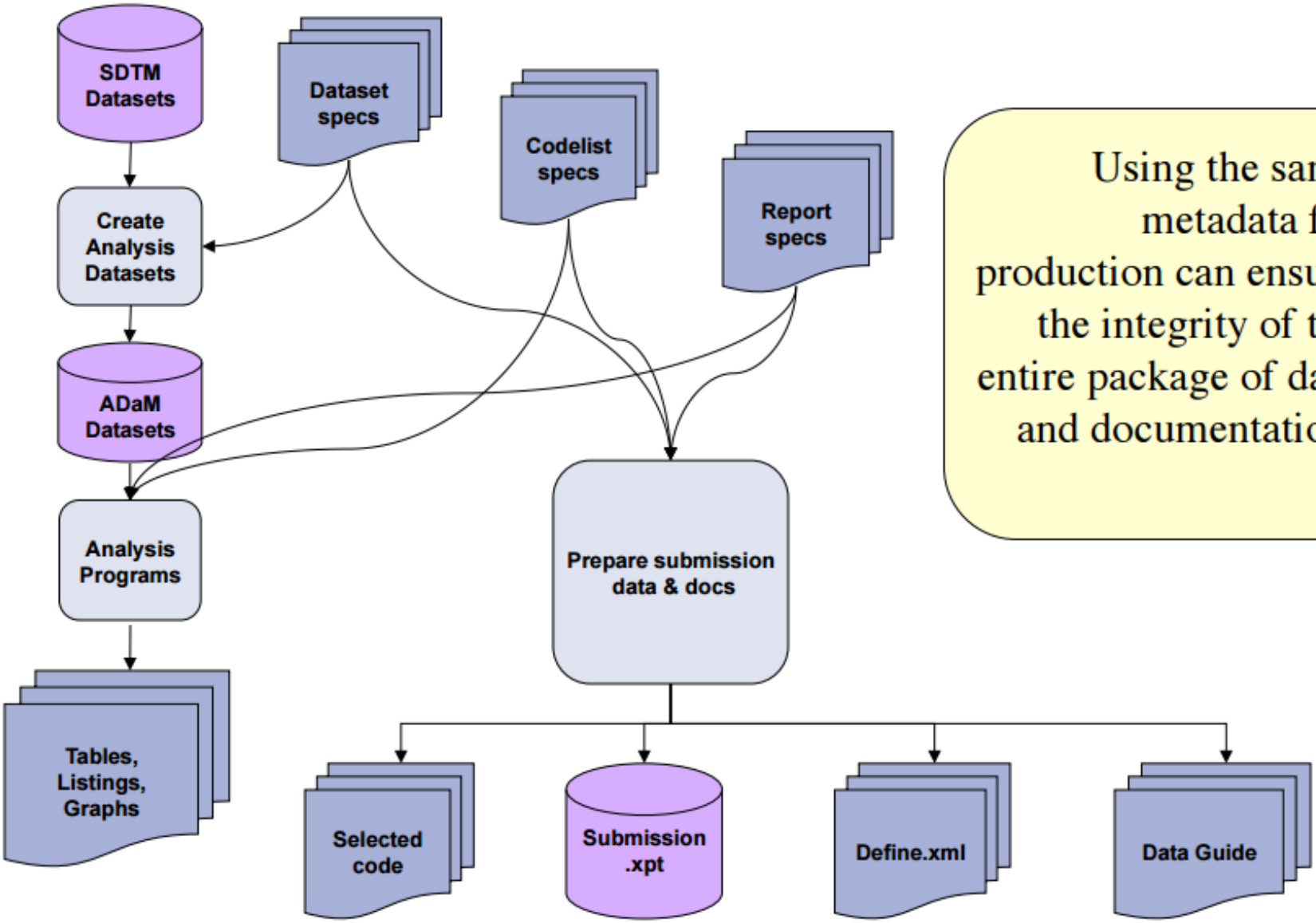
Better Utilize Metadata

- ✓ SDTM/ADaM Domain Attributes
- ✓ Format catalog
- ✓ Define.xml
- ✓ Source for SAS Programs
- ✓ SDTM/ADaM attributes
- ✓ Lookup Tables for Flags
- ✓ Program Index for Titles/Footnotes and Table Numbers

Metadata for DEFINE.XML

DEFINE.XML SECTION	TARGET COLUMN/ELEMENT ATTRIBUTE IN DEFINE.XML	SOURCE
Data Metadata	• Dataset • Description • Location (= standard path + dataset name)	DICTIONARY.COLUMNS/ SASHELP.VCOLUMN
Variable Metadata	• Variable • Label • Type (derivation required) • Controlled Terms or Format Element attributes not displayed but required for machine-readability: • Length • Significant Digits • Display Format	DICTIONARY.COLUMNS/ SASHELP.VCOLUMN
DEFINE.XML SECTION	**TARGET COLUMN/ELEMENT ATTRIBUTE IN DEFINE.XML**	**SOURCE**
Variable Value Level Metadata	• Source Variable • Value • Label	• Value = Value of source variable in source dataset (e.g., --TESTCD) • Label = Value of corresponding test name variable in source dataset (e.g., --TEST)
Controlled Terminology/Code Lists	• Reference Name • Code Value • Code Text Element attributes not displayed but required for machine-readability: • Data Type	SAS format library

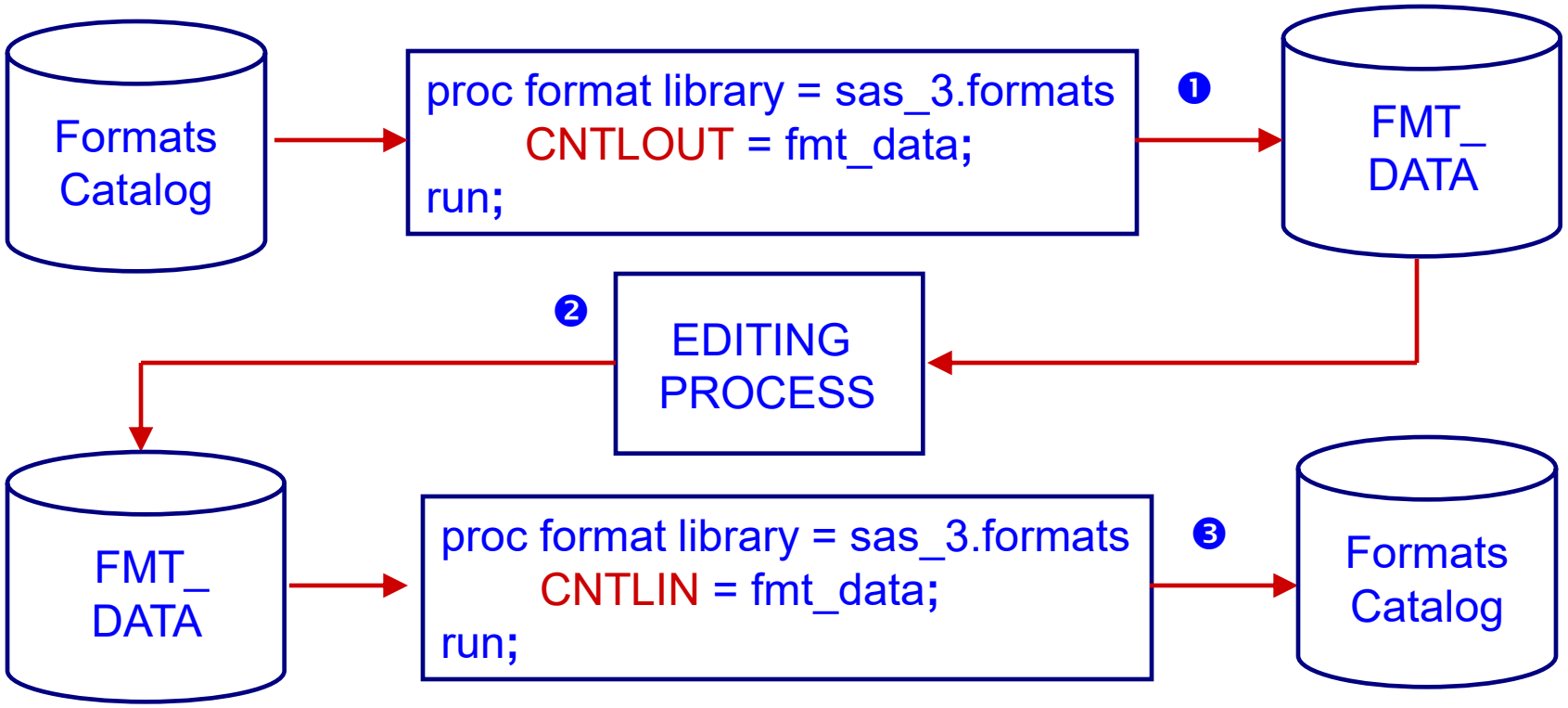
Metadata: Opportunities for Automation



Using the same metadata for production can ensure the integrity of the entire package of data and documentation.

Metadata: Maintaining Formats

One way to maintain permanent formats is to use Control Data Sets with PROC FORMAT.



1. Use the CNTLOUT option to create an output dataset that has info on all formats,
2. Use PROC FSEDIT, or VIEWTABLE to add, delete or change information
3. Use the CNTLIN option to re-create the formats from the modified data set.

Master ODS/ Proc Template

From excel files for data monitoring to submission quality summary tables

	A	B	C	D	E	F	G	H	I	J	K
1	ACME Worldwide Unit Sales										
2	Part #	Jan	Feb	Mar	Q1 TOTAL	Apr	May	Jun	Q2 TOTAL	Jul	Aug
3	Anvils										
4	A15001	443	565	731	1739	564	765	321	1650	321	346
5	A16002	121	332	56	509	876	432	232	1540	235	789
6	A17003	29	102	78	209	876	232	444	1552	345	2
7	A18004	78	345	34	457	964	658	349	1971	12	866
8	A19005	34	7	78	119	23	432	111	566	566	456
9	A19006	345	423	453	1221	454	234	567	1255	123	741
10	A19007	678	567	453	1698	3687	321	777	4785	100	358
11	A19008	456	356	561	1373	433	54	562	1049	450	290
12	A19009	890	780	125							
13	Total	3074	3477	2569							

Basic ODS syntax to create excel files for easy viewing and documenting issues by non-technical and management

Summary of table of Creatine at baseline
ADLB.AVISIT='BASELINE'
(Per Protocol Population)
ADLB.PPROTFL='Y'

Advanced ODS / Proc Template to create customized table layout and formatting for team to use

ADLB.PARAMCD	Group 1: Treatment 1 (N=xxx) ADLB.TRTAN = 1		Group 2: Placebo (N=xxx) ADLB.TRTAN = 2	
	n	Observed Mean Value	n	Observed Mean Value
Creatine WHERE PARAMCD='CREAT'	COUNT(ADLB.AVAL)	MEAN(ADLB.AVAL)	COUNT(ADLB.AVAL)	MEAN(ADLB.AVAL)
Log of Creatine WHERE PARAMCD='L10CREAT'				

Something for nothing? Adding group descriptive statistics

Sunil Gupta | JUNE 20, 2012

8533 6 Tweet 0 G+1 0 Like 3 Share 24

Can you actually get something for nothing? With PROC SQL's subquery and remerging features, yes, you can.

	Name	Sex	Age	Height	Weight	sexn
1	Judy	F	14	64.3	90	
2	Jane	F	12	59.8	84.5	
3	Joyce	F	11	51.3	50.5	
4	Barbara	F	13	65.3	98	
5	Carol	F	14	62.8	102.5	
6	Mary	F	15	66.5	112	
7	Louise	F	12	56.3	77	
8	Alice	F	13	56.5	84	
9	Janet	F	15	62.5	112.5	
10	Philip	M	16	72	150	
11	James	M	12	57.3	83	
12	Henry	M	14	63.5	102.5	
13	John	M	12	59	99.5	
14	William	M	15	66.5	112	
15	Alfred	M	14	69	112.5	
16	Jeffrey	M	13	62.5	84	
17	Thomas	M	11	57.5	85	
18	Ronald	M	15	67	133	
19	Robert	M	12	64.8	128	

Count of females

Count of males

Proc SQL Subquery Linked by Group Variable

```
proc sql;   create table class2 as
  select  a.*,
  b.sexn
  from sashelp.class as a
  left join
  (select
   sex,
   count(sex) as sexn
   from sashelp.class  where sex > ' '
   group by sex
   ) as b on
  a.sex=b.sex;
quit;
```

1. Select all master variables
2. Include the new group descriptive statistics variable
3. Apply LEFT JOIN to keep all master records
4. Create internal dataset using subquery
5. Select or create grouping variable
6. Apply one or more summary functions
7. Link internal dataset by grouping variable

Save two steps!

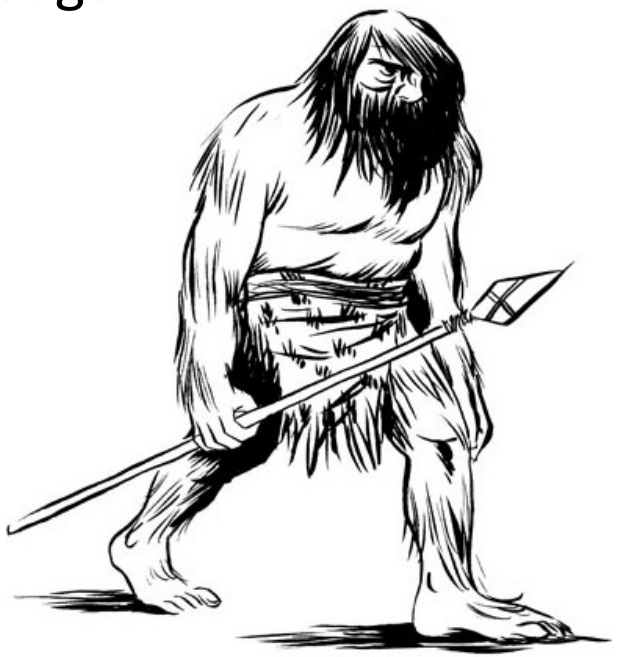
Create Macros to Standardize the Process

- ✓ Defensive Programming, Snapshots
- ✓ Setup, Relative Path Macro Variables, Libnames, Global Macros, etc.
- ✓ SDTM, ADaM Variable Derivations
- ✓ DTCTDT to convert XXDTC to XXDT
- ✓ Date Impute
- ✓ Data Cutoff based on metadata and code generator (cutoff macro)
- ✓ Log Scan
- ✓ Summary Table Macros (Demog, Disposition, AE, ConMeds, Labs, etc.)

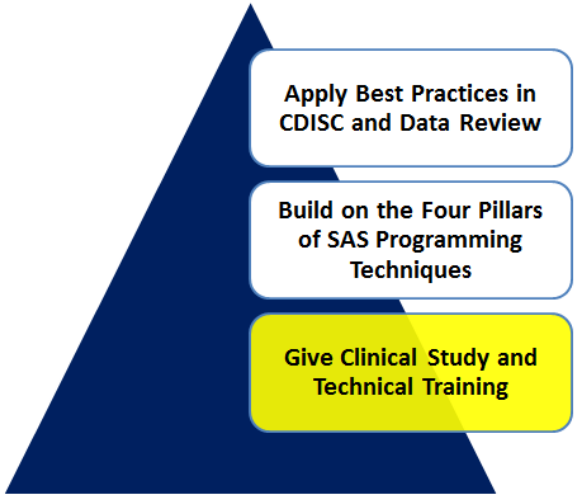
A World without SAS Macros (&, %)

What You can Expect *Without* SAS Macros:

- ✓ No order to get common tasks completed
- ✓ No control over the time to deliver
- ✓ No assurance of the quality of work
- ✓ No one in control to correct issues or errors
- ✓ No method to diagnose issues or errors
- ✓ No method to grow or build on knowledge
- ✓ Loss of client confidence
- ✓ Loss of business



Give Clinical Study and Technical Training



On-Board Training and Mentoring

Table 3: Two main components of training programme

Best practices training programme	Mastery of SAS programming	Understanding of clinical trials data
Key objective	• Learn to apply advanced techniques and macros to create reports and graphs • Learn effective debugging and validation methods • Learn effective testing and documentation methods	• Address complex clinical data issues • Apply correctly primary and secondary endpoints • Understand drug development process
Selected resources (books, SAS papers)	• Sharpening Your SAS Skills • SAS Application Guide • ODS: The Basics • The Complete Guide to the Macro Language and Proc Report	• SAS Programming in the Pharmaceutical Industry • Sample Clinical Study
Sample hands-on exercises	• Create rtf files using ODS • Validate tables, lists and graphs	• Data edit checks • Create CRTs • Create tables and lists

Fill any Technical and Soft Skills Gap

- ✓ Communication, Documentation
- ✓ Details, Summary Tables
- ✓ Macro Development, User Guides
- ✓ Deliverables, Timelines
- ✓ DDT, Table Shells, Updates
- ✓ New Team Member Skills Assessment
- ✓ SAS Debugging Skills, ODS Statistical Graphs using PROC SGPLOT

Delegate Ownership and Responsibilities

SDTMs, ADaMs, Define.xml

Summary of table of Creatine at baseline
ADLB.AVISIT='BASELINE'
(Per Protocol Population)
ADLB.PPROTFL='Y'

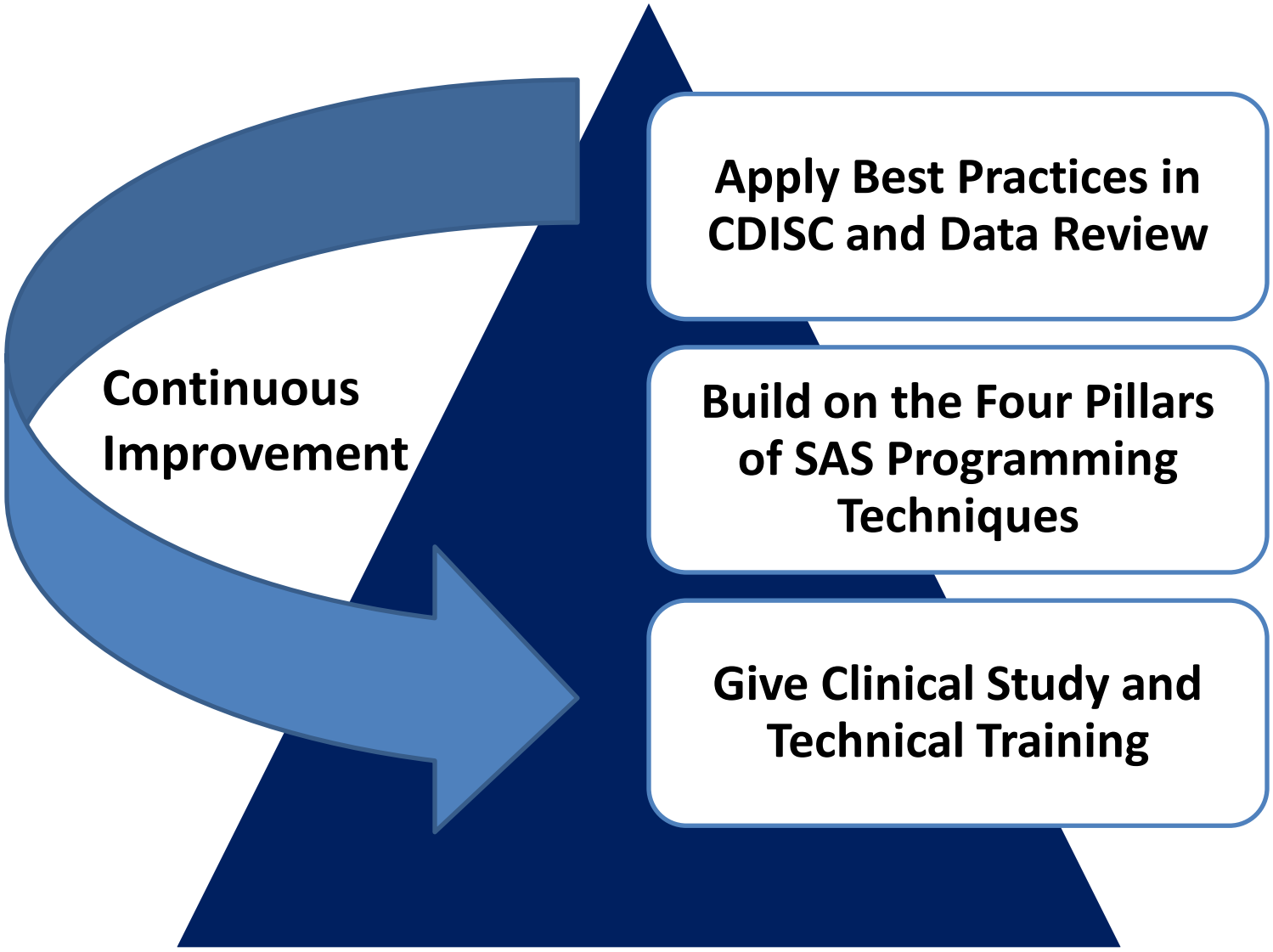
ADLB.PARAMCD	Group 1: Treatment 1 (N=xxx) ADLB.TRTAN = 1		Group 2: Placebo (N=xxx) ADLB.TRTAN = 2	
	n	Observed Mean Value	n	Observed Mean Value
Creatine WHERE PARAMCD='CREAT'	COUNT(ADLB.AVAL)	MEAN(ADLB.AVAL)	COUNT(ADLB.AVAL)	MEAN(ADLB.AVAL)
Log of Creatine WHERE PARAMCD='L10CREAT'				

Tables, Listings, Figures

Data Issues

Source/QC cross-train to support each other

Better Management & Higher Quality Deliverables



Best of Both Worlds: Better Management and Higher Quality Deliverables

Sunil Gupta
Associate Director, Statistical Programming
Cytel