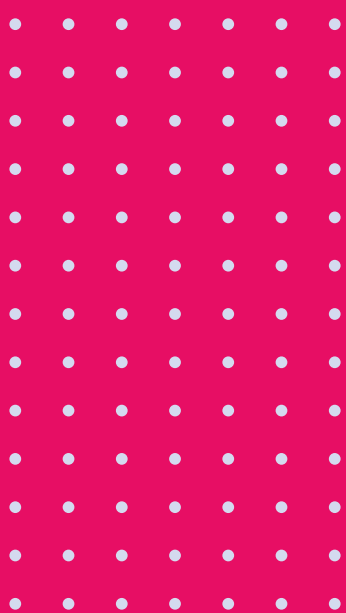




Comparing Apples and Oranges: ISS Comparison

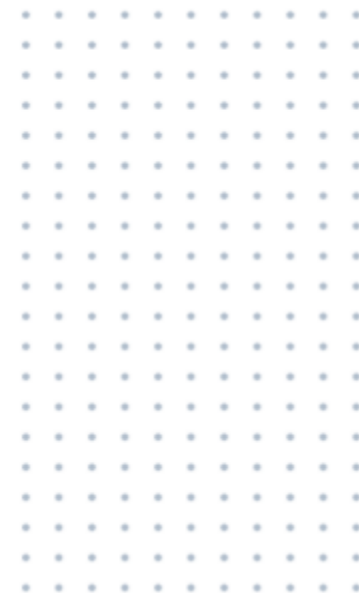
Elliot Pearn

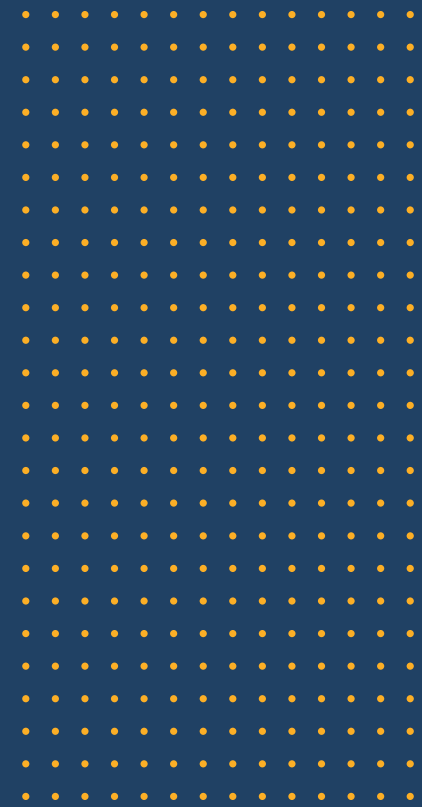
Senior Programmer II, Veramed

PHUSE US Connect 2023



- What is an ISS?
- Development
- Future Updates
- My ISS experience
- How does it work?
- Questions

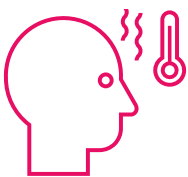




What is an Integrated Summary of Safety (ISS)?

What is an Integrated Summary of Safety (ISS)?

Combined summary of the safety data of studies to mainly assess:

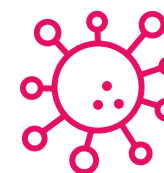


Common Adverse Events (AEs):

- Related to Study treatment
- Severity
- Fatality



Exposure



Disease Characteristics



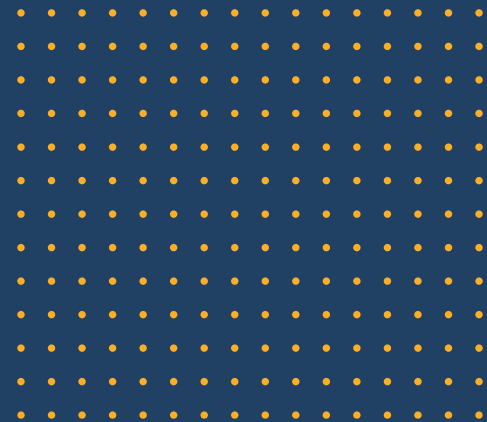
Laboratory Results



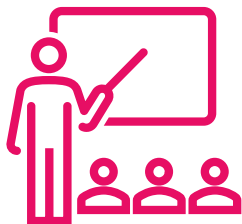
ECG/Vital Signs

Required by the FDA for New Drug Approvals (NDAs)

My ISS experience



My ISS experience



Lead COVID ISS Study
Contributed to several others



Combined 3 studies
(1 study with 3 parts)



Several Challenges:

- Data collection
- Traceability
- Values & Parameters
- Structure
- Programming

Structure Changes – Examples

Dataset	Data	Study 1	Study 2	Study 3	Study 4
Disease Characteristics	Virology	ADDC	ADDC	ADLB	ADLB
Disease Related Events		ADAE	ADDRE	ADAE	ADDRE
Ventilation		ADVENT	ADPR	ADVENT	ADPR

Value changes – Examples

Several Causes:

- CRF differences
- Formatting inconsistencies
- Code list updates



CRF differences

PARCAT1

SCREEN
SCREEN

PARCAT1

SCREEN FAILURE
SCREEN FAILURE



Formatting inconsistencies

ADURC

0d 1h 0m
0d 5h 31m
0d 0h 25m
0d 1h 42m

ADURC

1h
5h 31m
25m
1h 42m



Code list updates

AREL

Not related
Not related
Not related

AREL

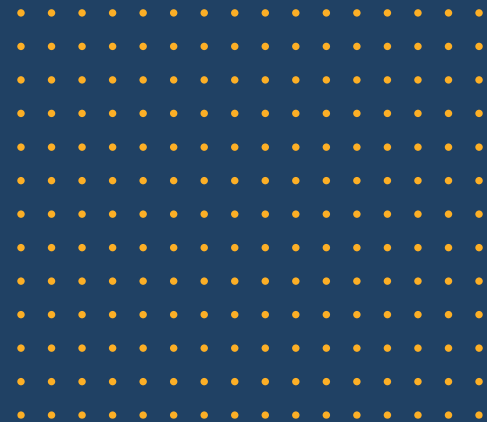
Not Related
Not Related
Not Related

Variable Changes - Examples

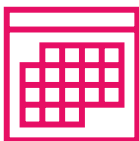
Variable	Values	Study 1	Study 2	Study 3	Study 4
Age Group 1 (AGEGR1)	<55, 55-64, 65-74, 75-84	AGEGR4	n/a	AGEGR3	AGEGR1
Age Group 2 (AGEGR2)	<=18,19-64, >=65	AGEGR2	AGEGR1	AGEGR1	AGEGR2

Variable	Values	Study 1	Study 2	Study 3	Study 4
CQ01NAM	Infusion-related reactions	CQ01NAM	CQ03NAM	CQ02NAM	CQ01NAM
CQ02NAM	COVID-19 symptoms	CQ02NAM	CQ02NAM	CQ03NAM	CQ03NAM

Development



Planning & Creation



Planning:

- Where will it fit?
 - Can run at any time during the ISS process once ADaMs completed
- Desired outcomes
 - Ensure traceability – be able to track any changes to source datasets to ISS
 - Compare displays as well as ADaMs
 - ISS & ISE?
- Research
 - Little experience in tool creation
 - Macro Programming
 - SAS 2 BASH
- Usable across several studies
- Setting aside time



Creation:

- Break it down into sections:
 - Dataset filtering first
 - Dataset manipulation
 - Display comparison
- Fluidity
- Testing
- Documentation

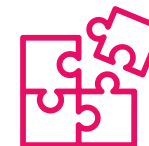
ISSUES FACED



Experience



Systems



Inconsistencies

TESTING



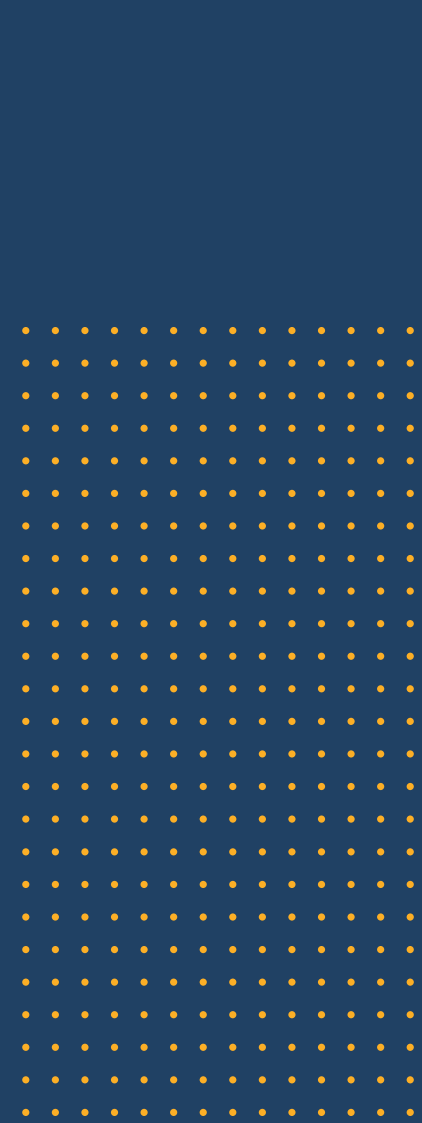
2 studies mainly
tested



Different studies to get a
comparison (ISS only)



ADaMs integrated
at SDTM level



How does it work?

Introduction to the tool

3 main parts:



Filtering and
outputting

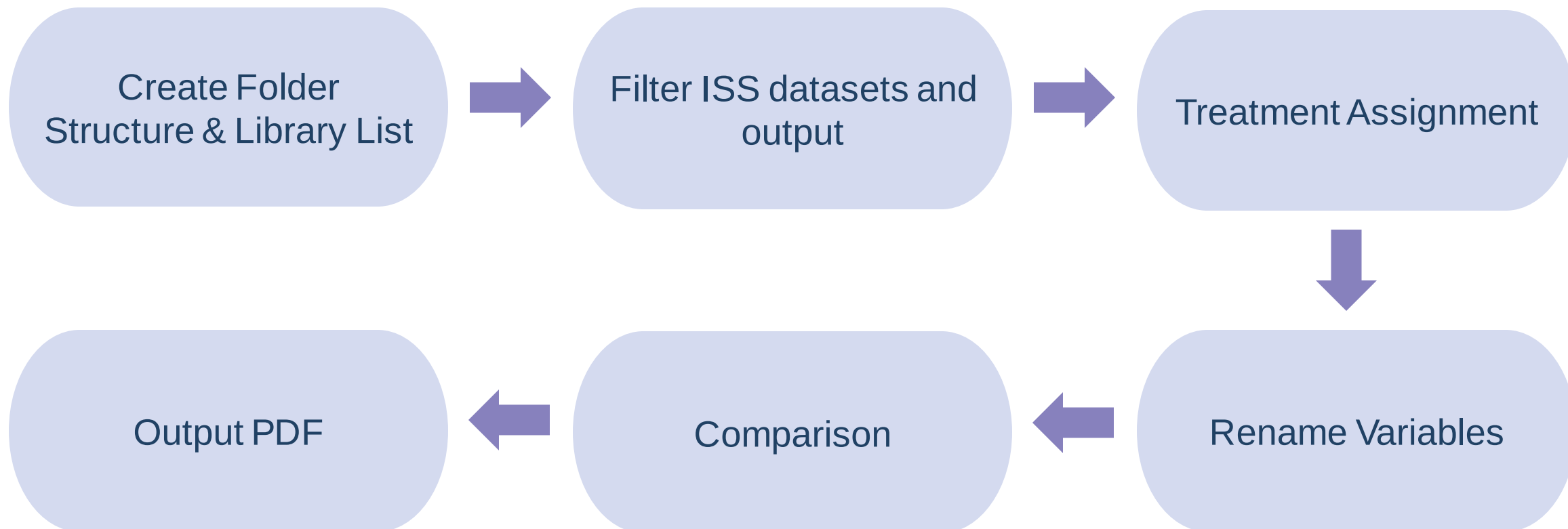


Dataset
manipulation &
comparison



Display rerun
& comparison

Datasets:



Preamble

Macro Variables:

Input Parameters: NAME	DESCRIPTION	DEFAULT
FOLDER	Specifies whether a 'TEST' area for comparison needs creating	N
MAINLIB	Specifies the file directory of the ISS	(None)
COMPLIB	Specifies the file directory where the compare datasets will be created. RECOMMEND: Creating a test area within the same RE and copying folder structure.	(None)
COMPN	Specifies the number of studies included in the ISS	(None)
DS_REMOVE	Specifies the name of any datasets that you do not want to compare	(None)
LIB1-LIB6	Specifies the short names of the individual origin studies used to create datasets and library names (e.g STUDY1 for 1st comparison study)	(None)
FILTER1-6	Specifies the filter required on the ISS datasets to create origin study datasets	(None)
COMPLIB1-6	Specifies the library of the origin study	(None)
CSV	Specifies the location and name of the CSV file required for renaming variables for comparison - ISS_Var_Rename.csv. CSV has 3 columns: DATASET - specify datasets where change is required in. STUDY - origin study where change is required (needs to match values in LIB1-LIB6) OLD_VAR - Name of variable in origin study NEW_VAR - Name of variable in ISS study	(None)
COMPLOC	Specifies the desired location of the Compare PDFs	(None)
DISP	Specifies whether Displays will also be compared	N
DATAC	Specifies whether Datasets will be compared	Y
DRIVLIB	Specifies the location of the display drivers	(None)

Example Calls:

```
/*
%let mainlib=%str(compound/study/area/);
%let complib=%str(compound/study/area/test/);
%let compn=3;
%let ds_remove = %str("ADAE" "ADCM");
%let lib1=STUDY1;
%let lib2=STUDY2;
%let lib3=STUDY3;
%let lib4=;
%let lib5=;
%let lib6=;
%let filter1=%str(STUDYID = "STUDY1");
%let filter2=%str(STUDYID = "STUDY2");
%let filter3=%str(STUDYID = "STUDY3" and PART = "PART A");
%let filter4=;
%let filter5=;
%let filter6=;
%let csv = %str(compound/study/area/documents/ISS_Var_Rename.csv);
%let complib1=%str('compound/study1/area/adamdata');
%let complib2=%str('compound/study2/area/adamdata');
%let complib3=%str('compound/study3/area/adamdata');
%let complib4=;
%let complib5=;
%let comploc=%str('/compound/study/area/qc/Compares/');
%let disp = Y;
%let drivlib = %str(compound/study/area/);
*/
```

Code Snippets

Create Folder Structure:

```
%if folder = Y %then %do;
%macro create_lib;
*Create a folder for comparison if required;
%let createlib = %sysfunc(dequote(&complib.));

data _null_;
  call symput("folder",&createlib.);
  call symput("adam",&createlib./adamdata/");
  call symput("ardata",&createlib./ardata/");
  call symput("code",&createlib./code/");
  call symput("dddata",&createlib./dddata/");
  call symput("documents",&createlib./documents/");
  call symput("driver",&createlib./drivers/");
  call symput("output",&createlib./output/");
  call symput("pkdata",&createlib./pkdata/");
  call symput("rawdata",&createlib./rawdata/");
  call symput("refdata",&createlib./refdata/");
  call symput("logs",&createlib./saslogs/");
  call symput("sdtm",&createlib./sdtm/");
run;

x "mkdir &folder";
x "mkdir &adam";
x "mkdir &ardata";
x "mkdir &code";
x "mkdir &dddata";
x "mkdir &documents";
x "mkdir &driver";
x "mkdir &output";
x "mkdir &pkdata";
x "mkdir &rawdata";
x "mkdir &refdata";
x "mkdir &logs";
x "mkdir &sdtm";

%do z = 1 %to &compn;
  data _null_;
    call symput("&crlib&z.C",&createlib.adamdata/&&lib&z..C/");
    call symput("&crlib&z.",&createlib.adamdata/&&lib&z../");
  run;

  x "mkdir &&&crlib&z.C.";
  x "mkdir &&crlib&z..";

%end;

%mend create_lib;
%create_lib;
```

Use SAS 2 BASH Code to create folders – if required

Create Library List:

```
**Create lib_list macro variable which is used throughout;
%macro create_lib_list;
%macro var;

data lib_list;
  %do a = 1 %to &compn.;
    length lib $200;
    lib= strip("&&lib&a..");
    output;
  %end;
run;

%mend;
%var;

%macro create();
%let dsid=%sysfunc(open(lib_list));
%let cnt=%sysfunc(attrn(&dsid,nobs));
%do b=1 %to &cnt.;
  %let rc=%sysfunc(fetchobs(&dsid,&b));
  %mpres(%sysfunc(getvarc(&dsid,%sysfunc(varnum(&dsid,lib)))));
%end;
%let rc=%sysfunc(close(&dsid));
%mend create;

%global lib_list;
%let lib_list = %create;

%mend create_lib_list;
%create_lib_list;
```

Look through the number of studies to compare and create a macro variable as "STUDY1, STUDY2..."

Create Folder Structure:

> Network > UK1SALX00175.corpnet2.com > arenv > arwork > gsk4182136 > mid215199 > iss_01 > test

Name	Date modified	Type
adamdata	02/12/2021 10:20	File folder
ardata	02/12/2021 10:12	File folder
code	21/02/2022 13:08	File folder
dddata	01/04/2022 18:19	File folder
dmdata	02/12/2021 10:13	File folder
documents	09/12/2021 13:42	File folder
drivers	01/04/2022 18:19	File folder
output	01/04/2022 18:19	File folder
pkdata	02/12/2021 10:13	File folder
rawdata	02/12/2021 10:13	File folder
refdata	02/12/2021 15:40	File folder
saslogs	01/04/2022 18:19	File folder
sdtm	02/12/2021 10:13	File folder

STUDY1
 STUDY1C
 STUDY2
 STUDY2C
 STUDY3
 STUDY3C
 STUDY4
 STUDY4C
 STUDY5
 STUDY5C

The folders created for the
 ADaMs once output.
 One for the original datasets
 with the pooled treatment
 added and one for the
 filtered ISS datasets

The folder structure
 created

Filter ISS datasets and output:

```
**First call filter_dd macro - which filters ISS datasets and outputs for just origin study records;  
%macro filter_dd;  
    %macro filter(lib=dset_in=);  
  
        **Read in current ADaMs and output to compare folder filtered for each origin study;  
        libname comp "&complib.adamdata/&lib.C";  
        data comp.&dset_in;  
            set adamdata.&dset_in;  
  
            %do i = 1 %to &compn.;  
  
                %if &lib.=&&lib&i.. %then if &&filter&i.;;  
  
            %end;  
        run;  
  
    %mend;  
  
    ** Create macro variables for all ADaMs created in ISS;  
    data files;  
        keep filename;  
        length fref $8 filename $80;  
        rc = filename(fref, "&mainlib.adamdata/");  
        if rc = 0 then  
            do;  
                did = dopen(fref);  
                rc = filename(fref);  
                end;  
                dnum = dnum(did);  
                do m = 1 to dnum;  
                    filename = dread(did, m);  
                    n = m;  
                    /* If this entry is a file, then output. */  
                    fid = mopen(did, filename);  
                    if fid > 0  
                        then  
                            output;  
                        end;  
                    rc = dclose(did);  
                run;  
  
        **Remove extension from filename and filter out any datasets we do not want to compare;  
        data files;  
            set files (where=(index(filename, ".sas7bdat")>0));  
  
            filename=compress(tranwrd(filename, ".sas7bdat", ""));  
  
            if upcase(filename) not in (&ds_remove.);  
        run;
```

Read in the ISS ADaM,
filter and output to
allocated comparison
folder

Create a list of ADaMs
(minus ".sas7bdat") in
order to create a macro
variable

```
**Count datasets;  
%global ds_count;  
proc sql noprint;  
    select count(*)  
        into :ds_count  
        from files;  
quit;  
  
**Create macro variable for each dataset;  
proc sql noprint;  
    select filename into: dsnam1 - :dsnarn%left(&ds_count.)  
        from files;  
quit;  
  
    %do g = 1 %to &ds_count;  
    %global ds&g.;  
    %let ds&g.=&str(&&dsnarn&g.);  
    %end;  
  
**Loop filter macro through each library and each dataset;  
%local j next_lib;  
%let j=1;  
%do %while (%scan(&lib_list, &j) ne );  
    %let next_lib = %scan(&lib_list, &j);  
  
    %do p = 1 %to &ds_count.;  
        %filter(lib=&next_lib,dset_in=&&dsnarn&p.);  
    %end;  
  
    %let j = %eval(&j + 1);  
%end;  
%mend;  
%filter_dd;
```

Loop through created list
an allocate a numeric
macro variable to
dataset name...
e.g. dsnarn1=ADSL

Loop macro
through each
dataset and study

Filter ISS datasets and output:

The ISS ADaMs output to a compare folder – containing only STUDY1 records

test > adamdata > STUDY1C			
Name	Date modified	Type	Size
 adsaf.sas7bd		SAS7BDAT File	5,312 KB
 adae.sas7bdat	14/04/2022 15:09	SAS7BDAT File	1,152 KB
 addc.sas7bdat	14/04/2022 15:09	SAS7BDAT File	24,960 KB
 addre.sas7bdat	14/04/2022 15:09	SAS7BDAT File	128 KB
 adds.sas7bdat	14/04/2022 15:09	SAS7BDAT File	3,392 KB
 adex.sas7bdat	14/04/2022 15:09	SAS7BDAT File	5,632 KB
 adlb.sas7bdat	14/04/2022 15:09	SAS7BDAT File	69,760 KB
 adpr.sas7bdat	14/04/2022 15:09	SAS7BDAT File	192 KB
 adsl.sas7bdat	14/04/2022 15:09	SAS7BDAT File	1,536 KB

Treatment Assignment:

```
%put _global_;
**Get origin datasets ready for compare:
- Rederive treatment variables in ADSL and merge onto required datasets
- Rename or rederive any variables to allow for an easier compare;
%macro ds_comp_ready(inds=);

*Individual Study ADaMs - check latests REs;
%do k = 1 %to &compn.;
libname &&lib&k.. &&complib&k..;
%end;

%macro assign_trt(lib=);
data adsl_&lib.;
```



Manual Assignment of treatment information – duplicate of original ADSL code

```
**If NON-ADSL dataset is required for TLF comparison - then merge rederived treatments onto old dataset;
**If ADSL then just rename dataset to be in correct format;
libname comp "&complib.adamdata/&lib./";
%let null = 0;

%if %upcase(&inds.) ne ADSL %then %do;
%if %sysfunc(exist(&lib..&inds.)) %then %do;
data comp.&inds.;
merge &lib..&inds. (drop=trt0: in=inDS) adsl_&lib. (keep=studyid usubjid trt0: tr0:);
by studyid usubjid;

if inDS;

run;
%end;

%else %do;
%let null = 1;

data comp.&inds.;
null = 1;
run;

%end;
%end;
%else %do;
data comp.&inds.;
set adsl_&lib.;

run;
%end;
```

Assign ISS treatment allocations to pre-existing study datasets – in order to get a cleaner compare and run displays correctly

Treatment Assignment:

The original study ADaMs output and with the pooled treatment variables added for display running

test > adamdata > STUDY1			
Name	Date modified	Type	Size
 adsl.sas7bdat	14/04/2022 15:09	SAS7BDAT File	2,496 KB
 adsaf.sas7bdat	14/04/2022 15:09	SAS7BDAT File	3,136 KB
 adpr.sas7bdat	14/04/2022 15:09	SAS7BDAT File	192 KB
 adlb.sas7bdat	14/04/2022 15:09	SAS7BDAT File	278,848 KB
 adex.sas7bdat	14/04/2022 15:09	SAS7BDAT File	768 KB
 adds.sas7bdat	14/04/2022 15:09	SAS7BDAT File	1,792 KB
 addre.sas7bdat	14/04/2022 15:09	SAS7BDAT File	128 KB
 addc.sas7bdat	14/04/2022 15:09	SAS7BDAT File	19,072 KB
 adae.sas7bdat	14/04/2022 15:09	SAS7BDAT File	768 KB

Rename variables:

```

**Import CSV file for renaming variables - all variables that are in both studies just with diff names;
%if &null. ne 1 %then %do;
proc import datafile="&csv."
    out=renam dbms=csv replace;
    getnames=YES;
    guessingrows=max;
run;

```

A note on the CSV:

Name	Type
 ISS_Var_Rename	Microsoft Excel Comma Separated Values File

DATASET	STUDY	OLD_VAR	NEW_VAR
ADSL	STUDY1	AGEGR4	AGEGR1

DATASET: Dataset to be manipulated

STUDY: Study manipulation applicable too

OLD_VAR: Variable name in origin study

NEW_VAR: Variable name in the ISS

Rename variables (cont):

```
**Format memname to be the same as current datasets;
data renam_format;
  set renam;
  length memname $32;
  memname = strip(dataset);

  drop dataset;
run;

%let nam = %str(&lib._&inds._cont);

data ds_in;
  set %put &nam;;
run;

**Proc Contents to get attributes for variables;
proc contents data=comp.&inds. out=ds_in noprint;
title 'before renaming';
run;

**Set all attribute dataset together;
data cont_all;
  set ds_in;

  lib="&lib.";
run;

**Merge attributes with renaming dataset and only keep variables that are in both;
**Create a macro variable of the list of dataset that will need a variable renamed;
proc sql noprint;
  create table rename_identify as
  select a.memname, a.name, b.new_var, b.study
  from cont_all (where=(lib="&lib.")) as a, renam_format (where=(study="&lib.")) as b
  where a.memname = b.memname and a.name=b.old_var;

  select distinct(memname) into: ds_list
  separated by " "
  from rename_identify;
quit;
```

Import CSV and
format to merge
with data

Take attributes
from existing
data and keep if
in the CSV

```
**Create order variable;
proc sort data=rename_identify;
  by memname name;
run;

data rename_identify;
  set rename_identify;
  by memname name;

  retain varnum 0;
  if first.memname then varnum=1;
  else varnum=varnum+1;
run;

%end;
```

Rename variables (cont):

```
%macro renam(dsin=);  
  
  **Read in required dataset and find number of vars required;  
  data rename_num;  
    set rename_identify (where=(memname="&dsin" and study="&lib."));  
    by memname varnum;  
  
    if first.memname then call symput("start",strip(put(varnum,best.)));  
    if last.memname then call symput("end",strip(put(varnum,best.)));  
  run;  
  
  **Count number of obs and only run renaming code if needed;  
  proc sql noprint;  
    select count(*) into: nobs  
    from rename_num;  
  quit;  
  
  %if &nobs ne 0 %then %do;  
  
    **Create a list the old variables and the new variables into a macro variable;  
    proc sql noprint;  
      select distinct(name) into: old_list  
      separated by " "  
      from rename_identify (where=(memname="&dsin"));  
      select distinct(name) into: old_sql  
      separated by " "  
      from rename_identify (where=(memname="&dsin"));  
      select distinct(new_var) into: new_list  
      separated by " "  
      from rename_identify (where=(memname="&dsin"));  
    quit;  
  
    **Pull out the new variables and transpose so there is 1 record and all variables;  
    proc transpose data=rename_identify (where=(memname="&dsin")) out=new_var_t (keep=&new_list);  
      by memname;  
      id new_var;  
    run;  
  
    data sortedby;  
      set sashelp.vcolumn;  
      where libname="ADAMDATA" and memname=upcase("&inds") and sortedby ne 0;  
    run;
```

Read in dataset
and keep
variables that
require an update

Create macro
variables of "old" and
"new" variables in
order to loop

Use VCOLUMN to
get ADaM dataset
sort orders

```
proc sql noprint;  
  select distinct(name) into: sorted_list  
  separated by " "  
  from sortedby;  
  select distinct(name) into: sorted_sql  
  separated by " "  
  from sortedby;  
quit;
```

Create macro
variable for sort
order

```
**Create a temporary dataset containing old variables and merge variables;  
proc sql noprint;  
  create table temp as  
  select &old_sql, &sorted_sql  
  from comp.&dsin;  
quit;  
  
**Create a second temporary dataset containing all variables but the old variables;  
proc sort data=comp.&dsin. out=temp2 (drop=&old_list);  
  by &sorted_list;  
run;  
  
%LET ds=%SYSFUNC(OPEN(temp,i));  
%LET renam=%SYSFUNC(OPEN(new_var_t,i));
```

Create a dataset
containing variables to
update and sort

Rename variables (cont):

Use VCOLUMN
to get lengths
and labels of
“new” variables

```
**Renaming the variable process;
%do i=&start %to &end;
  %let newnam&i=%SYSFUNC(VARNAME(&renam,&i));
  %let dsvn&i=%SYSFUNC(VARNAME(&ds,&i));
  %if %index(&sorted_list,%upcase(&&dsvn&i)) eq 0 %then %do;
    %let vn&i=&&newnam&i;

    **Use VCOLUMN dataset to get attributes so that we can set lengths/label variables;
    data var_props;
      set sashelp.vcolumn;
      where libname="COMP" and memname="&dsin." and name="&&dsvn&i.";
      if type ne "" then call symputx("type&i",type,"G");

    run;

    data var_props2;
      set sashelp.vcolumn;
      where libname="ADAMDATA" and memname=upcase("&inds") and name=upcase("&&newnam&i.");
      if label ne "" then call symputx("lab&i",label);
      if type = "char" then if length ne . then call symputx("len&i",length);

    run;

  %end;
%end;

**Finalising renaming process;
data temp_renam;
set temp;

%do i=&start %to &end;
  %if %index(&sorted_list,%upcase(&&dsvn&i)) eq 0 %then %do;
  %if &&type&i eq char %then %do;
    length &&vn&i $&&len&i;
  %end;
  &&vn&i=&&dsvn&i;
  label &&vn&i = "&&lab&i.";
  drop &&dsvn&i.;
%end;
%end;
%let rc=%SYSFUNC(CLOSE(&ds));
%let rc=%SYSFUNC(CLOSE(&renam));
run;

proc sort data=temp_renam;
  by &sorted_list;
run;

**Merge data back together with new variables;
data comp.&dsin;
  merge temp2 (drop=&new_list) temp_renam ;
  by &sorted_list;
run;

%end;

%mend renam;
```

Loop through
variables, renaming
and assigning new
length and label

Loop macro
through datasets
and studies

```
**Take list of datasets required and loop renaming macro through;
%local k next_ds;
%let k=1;
%do %while (%scan(&ds_list, &k) ne );
  %let next_ds = %scan(&ds_list, &k);
  %renam(dsin=&next_ds);
  %let k = %eval(&k + 1);
%end;
%end;

%mend assign_trt;
**Specify list of libraries and run macro through them;
%local q next_lib;
%let q=1;
%do %while (%scan(&lib_list, &q) ne );
  %let next_lib = %scan(&lib_list, &q);
  %assign_trt(lib=&next_lib);
  %let q = %eval(&q + 1);
%end;

%mend ds_comp_ready;
%macro test;
%do g = 1 %to &ds_count.;
  %ds_comp_ready(inds=&&ds&g.);
%end;
%mend test;
%test;
```

Dataset comparison:

```
***Now run macro to compare datasets;
%macro compare_dd (in=,comp=);

  %*- SET SYSTEM OPTIONS LOCALLY FOR THIS MACRO;
  %* we are going to create a lot of output (e.g. proc compare) so set pagesize to max ;
  %* first, save the current options so that we can restore at end of macro ;
  %let original_PS = %sysfunc(getoption(PS));
  %let original_ORIENTATION = %sysfunc(getoption(orientation));
  options ps=max orientation=landscape;

  %* Setup ODS output for PDF file with name using macro parameter (filein) ;
  %* with system date attached e.g. /path/to/file/name_27JUL21.pdf ;
  %* where filein=/path/to/file/name ;
  ods listing close;
  ods noresults;
  ods pdf file="&comploc.&in._Data_Compare.pdf";

  %macro comp_loop(lib=);

    %do t = 1 %to &compn;
      %if &in.=&ds1. %then %do;
        libname compc "&complib.adamdata/&lib.C";
        libname comp "&complib.adamdata/&lib.";
      %end;
    %end;

    %* define a value to accumulate all the proc compare results into;
    %let compres=0;

    %* Loop through every dataset in the original/old library and proc compare ;
    %* with the dataset of same name in the new library. ;
    title "Compare of &in.";
    title2 "from- &lib";
    title3 "to- ISS";
    proc compare base=comp.&in.
      compare=compc.&in.
      listall /* List all vars and obs differences */
      maxprint=(50,2000); /* limit diffs reported (per-var,total)*/
    run;
    %* accumulate the sysinfo (which is a binary value);
    %let compres = %sysfunc(bOR(&sysinfo, &compres));

    %* once completed all proc compare, output result to log;
    %put %str(IN)FO: PROC COMPARE result for all datasets is [&compres.];

  %mend comp_loop;
```

Run compares and
output to PDF

```
**Specify list of libraries and run macro through them;
%local s next_lib;
%let s=1;
%do %while (%scan(&lib_list, &s) ne );
  %let next_lib = %scan(&lib_list, &s);
  %comp_loop(lib=&next_lib);
  %let s = %eval(&s + 1);
%end;
ods pdf close;

%* TIDY UP;

%* restore the original PS (pagesize) option ;
options PS=&original_PS orientation=&original_ORIENTATION;

%mend compare_dd;

%do p = 1 %to &ds_count.;
  %compare_dd(in=&ds&p.);
%end;
```

Loop through
every dataset

Dataset comparison:

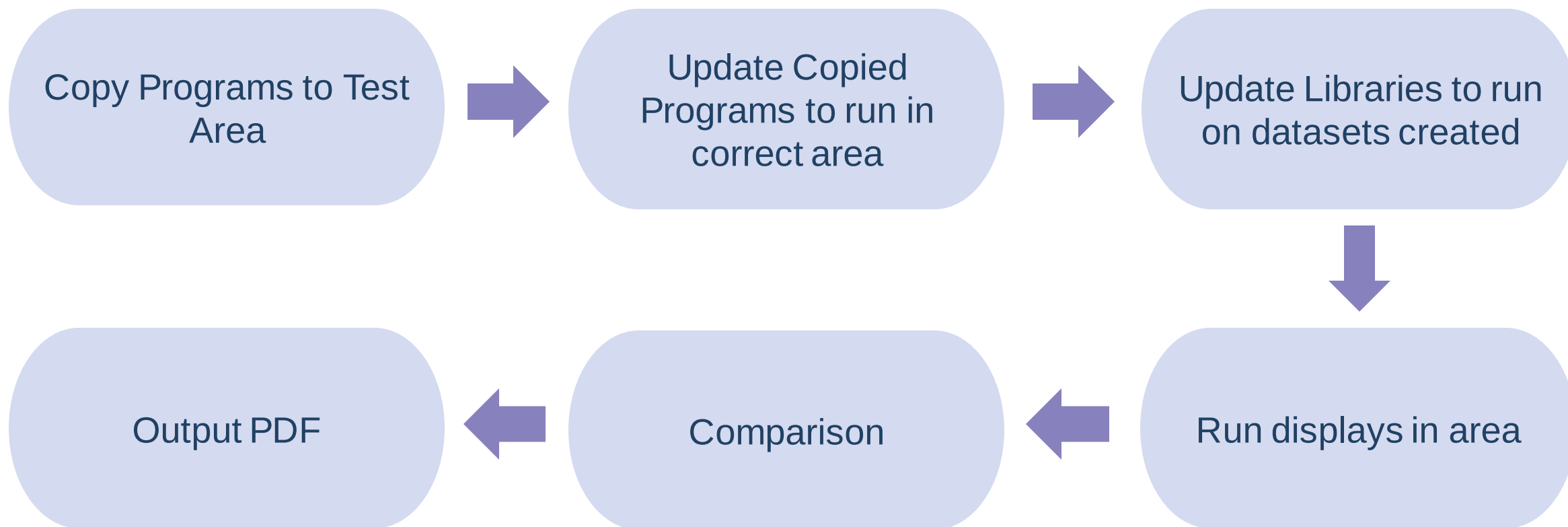
Name	Date modified	Type	Size
adae_Data_Compare	14/04/2022 15:09	Adobe Acrobat Docu...	179 KB
addc_Data_Compare	14/04/2022 15:09	Adobe Acrobat Docu...	426 KB
addre_Data_Compare	14/04/2022 15:09	Adobe Acrobat Docu...	127 KB
adds_Data_Compare	14/04/2022 15:09	Adobe Acrobat Docu...	184 KB
adex_Data_Compare	14/04/2022 15:09	Adobe Acrobat Docu...	262 KB
adlb_Data_Compare	14/04/2022 15:09	Adobe Acrobat Docu...	514 KB
adpr_Data_Compare	14/04/2022 15:09	Adobe Acrobat Docu...	136 KB
adsaf_Data_Compare	14/04/2022 15:09	Adobe Acrobat Docu...	118 KB
adsl_Data_Compare	14/04/2022 15:09	Adobe Acrobat Docu...	166 KB

A PDF is created for each ADaM – containing one compare for each study

Obs	BMI (kg/m ²) at Baseline				
	Base BMIBL	Compare BMIBL	Diff.	% Diff	
1	28.7000	28.7037	0.003704	0.0129	
2	19.6800	19.6846	0.004595	0.0233	
3	31.6800	31.6772	-0.002786	-0.008740	
4	24.9800	24.9770	-0.002957	-0.0118	
7	36.3800	36.3769	-0.003063	-0.008473	
8	32.9800	32.9819	0.001947	0.005903	
9	39.2400	39.2388	-0.001246	-0.003175	
10	36.4400	36.4440	0.003985	0.0109	

An example of a compare from ADSL – showing differences in how data was presented in the original study (BASE) and ISS (Compare)

Displays:



Display Comparison:

```
**** SAS 2 BASH CODE ****;
data copy;
  x echo Delete drivers from current folder to prevent overwriting;
  x cd &complib.drivers/;
  x \rm t_*.sas;
  x echo Copy drivers from prod into test area;
  x cd &drivlib.drivers;
  x cp -p t_*.sas &complib.drivers/;
  x echo Copy macros from prod into test area;
  x cd &drivlib.code;
  x cp -p rd_*.sas &complib.code/;
  x cp -p td_*.sas &complib.code/;
  x cp -p tu_*.sas &complib.code/;
  x cp -p ts_*.sas &complib.code/;
run;

data drivers;
  keep filename n;
  length fref $8 filename $80;
  rc = filename(fref, "&complib.drivers/");
  if rc = 0 then
  do;
    did = dopen(fref);
    rc = filename(fref);
  end;
  dnum = dnum(did);
  do c = 1 to dnum;
    filename = dread(did, c);
    n = c;
    /* If this entry is a file, then output. */
    fid = mopen(did, filename);
    if fid > 0
    then
    output;
  end;
  rc = dclose(did);
run;

data drivers;
  set drivers;

  filename=compress(tranwrd(filename, ".sas", ""));
run;

proc sql noprint;
  select count(*)
  into :file_count
  from drivers;
quit;

x echo File Count = &file_count;
```

SAS2BASH code to
copy drivers to
comparison area

Similar process to
creating ADaM macro
variables
E.g. file1 = table 1 prog
name

```
x echo File Count = &file_count;

proc sql noprint;
  select filename into: file1 - :file%left(&file_count.)
  from drivers;
quit;

data test;

%macro rename(lib=);

%let main = %sysfunc(scan(&mainlib., -1));
%let comp = %sysfunc(scan(&complib., -1));

x echo Starting Lib &lib.;
%if &lib. ne &lib1. %then %do;
x cd &drivlib.drivers;
x cp -p t_*.sas &complib./drivers/;
%end;
x cd &complib.drivers/;
%do i = 1 %to &file_count; ;
x echo Rename file &&file&i;
%if &i = 1 %then %do;
x echo Main = &main;
x echo Comp = &comp;
%end;

x cp -p &drivlib.drivers/&&file&i...sas &complib.drivers/&&file&i..._&lib...sas;
x sed -i "s+&main/+&main/&comp/+& " &complib.drivers/&&file&i..._&lib...sas;
x sed -i 's+ /+& " &complib.drivers/&&file&i..._&lib...sas;
x sed -i "s+adamdata+adamdata/&lib+g" &complib.drivers/&&file&i..._&lib...sas;
x sed -i "s+\.sas\>/\_&lib\.sas/g" &complib.drivers/&&file&i..._&lib...sas;
x sed -i "s+/&&file&i...\>/&&file&i..._&lib.+g" &complib.drivers/&&file&i..._&lib...sas;
x cd &complib.saslogs/;
x sas ../drivers/&&file&i..._&lib...sas;

%end;
%mend;

%do d = 1 %to &compn.;
%rename(lib=&&lib&d.);
**ISS data filtered on specific study for comparison;
%rename(lib=&&lib&d..C);
%end;

run;
```

SAS2BASH code to
rename display progs
to contain study name

Update libraries to
point to comparison
datasets and run

Display Comparison (cont):

```
%macro compare_displays(ddlib=);

data files;
  keep filename n rc;
  length fref $8 filename $80;
  rc = filename(fref, "&drivlib.drivers/");
  if rc = 0 then
    do;
      did = dopen(fref);
      rc = filename(fref);
    end;
    dnum = dnum(did);
    do i = 1 to dnum;
      filename = dread(did, i);
      n = i;
      /* If this entry is a file, then output. */
      fid = mopen(did, filename);
      if fid > 0
        then
          output;
      end;
      rc = dclose(did);
    run;

data files;
  set files (where=(index(filename,"t_")>0));

  filename=compress(tranwrd(filename,".sas",""));
run;

proc sql noprint;
  select count(*)
    into :file_count
    from files;
quit;

%put &file_count;

proc sql noprint;
  select filename into: file1 - :file%left(&file_count.)
    from files;
quit;
```

Create a macro
variables for each
display, similarly to
the ADaM process

```
libname dddatac &ddlib.;

%*- SET SYSTEM OPTIONS LOCALLY FOR THIS MACRO;
%* we are going to create a lot of output (e.g. proc compare) so set pagesize to max ;
%* first, save the current options so that we can restore at end of macro ;
%let original_PS = %sysfunc(getoption(PS));
%let original_ORIENTATION = %sysfunc(getoption(orientation));
options ps=max orientation=landscape;

%* Setup ODS output for PDF file with name using macro parameter (filein) ;
%* with system date attached e.g. /path/to/file/name_27JUL21.pdf ;
%* where filein=/path/to/file/name ;
ods listing close;
ods noresults;
ods pdf file="&comploc.DDDData_Compare.pdf";

%macro comp_loop(lib=);
%do i = 1 %to &file_count; ;
%* define a value to accumulate all the proc compare results into;
%let compres=0;

%* Loop through every dataset in the original/old library and proc compare ;
%* with the dataset of same name in the new library. ;
title "Compare of display";
title2 "from- &lib";
title3 "to- ISS";
proc compare base=dddatic.&file&i._&lib.
  compare=dddatic.&file&i._&lib.C
  listall /* List all vars and obs differences */
  maxprint=(50,2000); /* limit diffs reported (per-var,total)*/
run;
%* accumulate the sysinfo (which is a binary value);
%let compres = %sysfunc(bor(&sysinfo, &compres.));
%end;
%mend comp_loop;
%do d = 1 %to &compn.;
%comp_loop(lib=&&lib&d.);
%end;
ods pdf close;

%* TIDY UP;

%* restore the original PS (pagesize) option ;
options PS=&original_PS orientation=&original_ORIENTATION;

%mend compare_displays;

%compare_displays(ddlib="&complib.dddatic");

%end;

%mend iss_comparison;
```

Similar process to ADaM
comparison – loop
through every study and
display and compare to
original

Display Comparison:

test > drivers				
Name	Date modified	Type	Size	
t_sp_dm_pt2_age1	28/02/2022 13:16	SAS File	10 KB	
t_sp_dm_pt2_age1_STUDY1	01/04/2022 18:02	SAS File	10 KB	
t_sp_dm_pt2_age1_STUDY1C	01/04/2022 18:14	SAS File	10 KB	

The display program, firstly copied and then recreated to be run twice

Compares				
Name	Date modified	Type	Size	
DDData_Compare	04/04/2022 11:24	Adobe Acrobat Docu...	2,848 KB	

The PDF created containing all compares for all displays

Example 1:

Obs	Base Value TT_DECODE1	Compare Value TT_DECODE1
5	75 - 84	75 - 85
6	>=85	>85
8	<=18	<18
9	19 - 64	18 - 65
10	>=65	>65

Compare differences in the displays – due to changes in how AGEGR was defined

Placebo~(N=526)		
Obs	Base Value TT_RESULT0001	Compare Value TT_RESULT0001
8	4 (<1%)	1 (<1%)
9	414 (79%)	432 (82%)
10	108 (21%)	93 (18%)

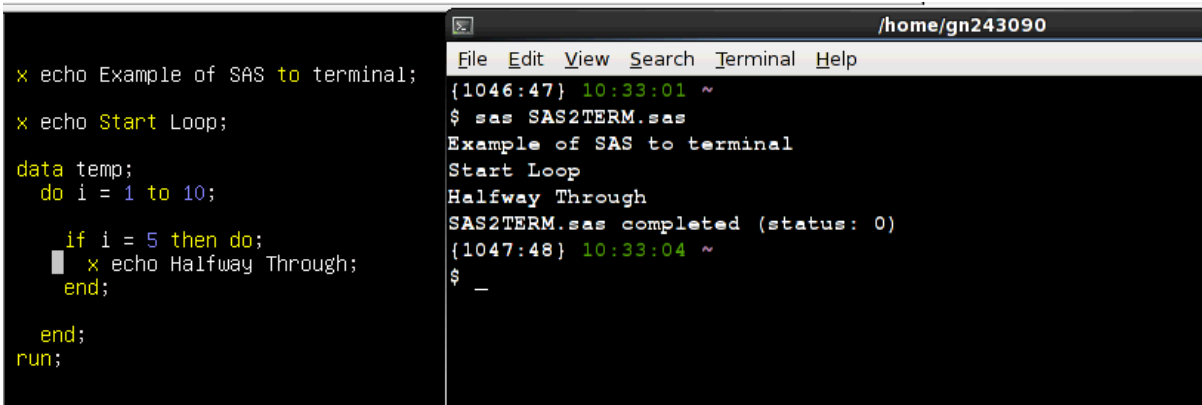
Example 2:

Number of Observations with Some Compared Variables Unequal: 0.
Number of Observations with All Compared Variables Equal: 7.
NOTE: No unequal values were found. All values compared are exactly equal.

The end goal – a clean compare!

SAS 2 BASH

- Allows running of Bash (Unix) shell commands in SAS programs
- Useful on Linux systems – similar to PowerShell
- Mainly used for printing code to check progress or debugging complex code
- Used in the macros to copy and update programs as well as automated running of displays



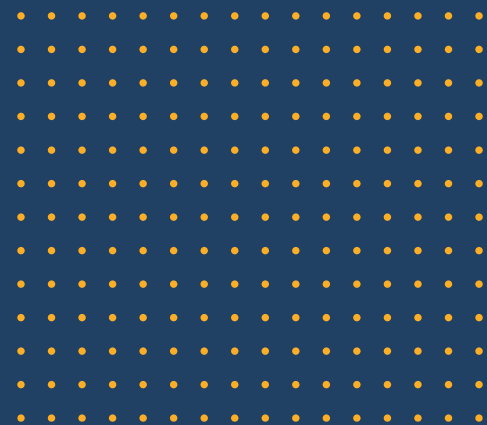
```
x echo Example of SAS to terminal;
x echo Start Loop;
data temp;
  do i = 1 to 10;
    if i = 5 then do;
      x echo Halfway Through;
    end;
  end;
run;
```

```
File Edit View Search Terminal Help
{1046:47} 10:33:01 ~
$ sas SAS2TERM.sas
Example of SAS to terminal
Start Loop
Halfway Through
SAS2TERM.sas completed (status: 0)
{1047:48} 10:33:04 ~
$ _
```

Examples:

- x [TERMINAL COMMAND]; - Running a terminal command
- x echo [TEXT TO PRINT]; - Print a message to the terminal
- x sed [CODE]; - Replace text within a program
- x cp [CODE]; - Copy a file/program/document
- x sas[CODE]; - Run a SAS program

Future Updates



Potential future updates



Parameter
Value
Changes



Improved
Renaming
Process



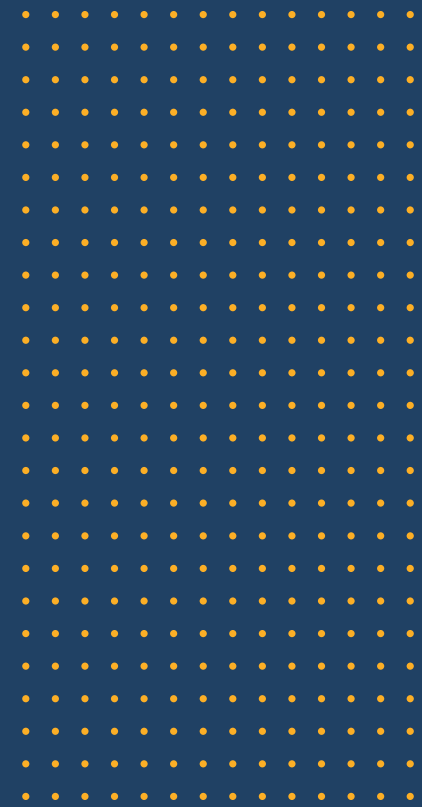
Clean up the
compares



Automated
Treatment
Assignment



Widen the
scope



Questions or Suggestions?

Thank you!

Find us at **BOOTH 5**

and learn more about our various:

- Presentations
- Workshops
- Panels

Contact me:

elliott.pearl@veramed.co.uk