

Utilizing SAS[®] for Cross-Report Verification in a Clinical Trials Setting

Daniel Szydlo, SCHARP/Fred Hutch, Seattle, WA
Iraj Mohebalian, SCHARP/Fred Hutch, Seattle, WA
Marla Husnik, SCHARP/Fred Hutch, Seattle, WA

Outline

- * Motivation
- * General framework
- * Details of the SAS[®] code
- * Limitations
- * Conclusions

Motivation to develop a cross- report verification system

Motivation

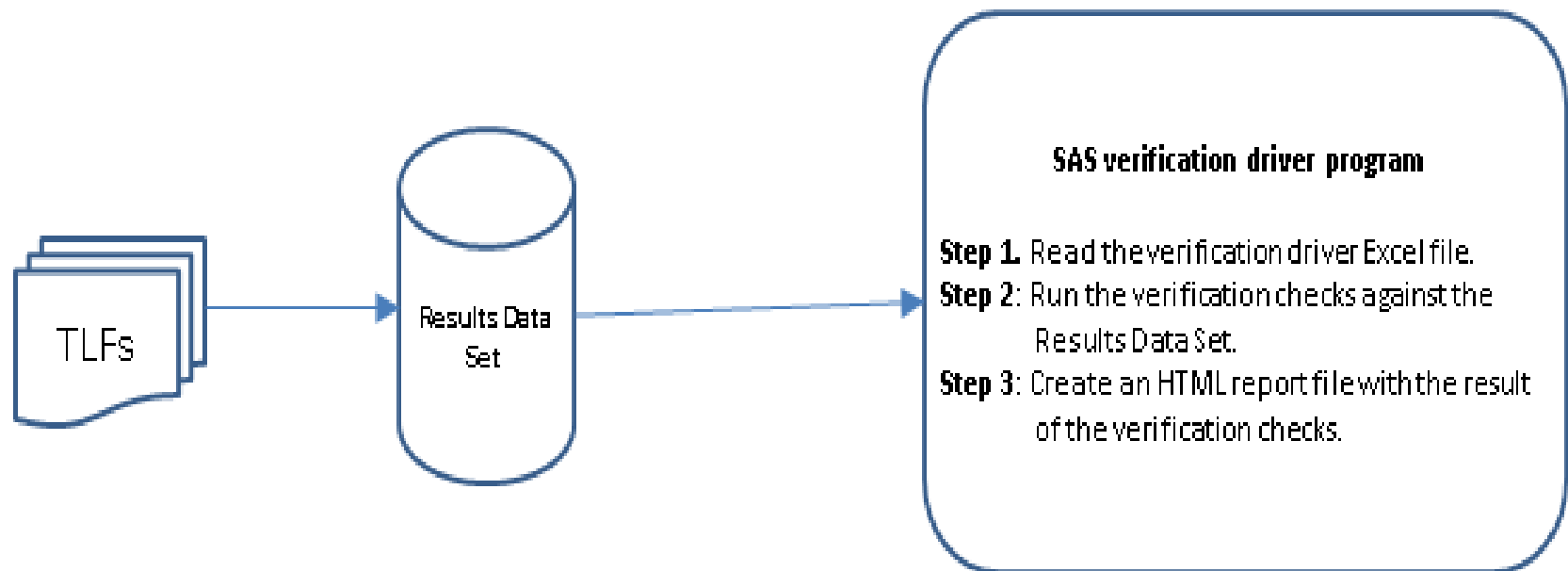
- * We were working on a large phase III clinical trial for HIV prevention with requirement for bi-annual Data Safety and Monitoring Reports (DSMB).
- * Faced with verifying hundreds of pages of tables, listings and figures (TLFs) under tight timelines.
- * Manual review of TLFs is very time-consuming and error-prone.
- * Reports were growing in size over time and frequency of reporting was enough to warrant investing in the development of an automated system.

General framework for the cross- report verification system

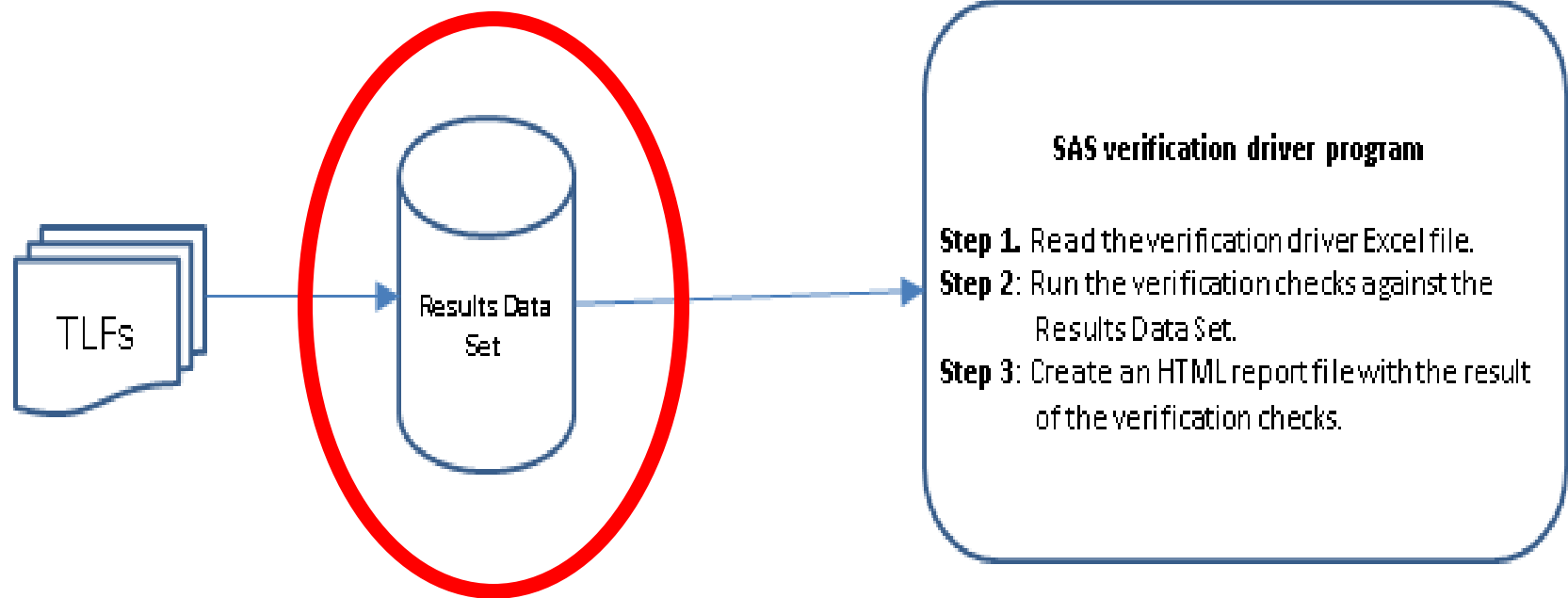
General Framework

- * We developed procedures utilizing SAS® to automate a verification checks, improving the accuracy and efficiency of the TLFS for DSMB reports
 - * **Results data sets** used to generate TLFs
 - * **Excel driver file** created and contained verification checks to be performed
 - * **Verification checks** developed and performed on the results datasets
 - * **HTML-formatted file** developed to show what errors occurred and where they occurred

General Framework



Results Data Set



Features of the Results Data Set

- * Standard data sets are created for each TLF through the use of SAS[®] macros
- * Data sets are concatenated to form one large results data set
- * Standardized format and contains enough information to make every element of the TLFs uniquely identifiable
- * Basis for the verification checks

Variables in the Results Data Set

- * **prgmid** (program ID): Directory location and file name of SAS® program
- * **dataset_created_from**: Unique identifier representing name of the standardized data set based on the TLF
- * **number_of_titles, number_of_footnotes**: Number of titles & footnotes from a TLF
- * **title1-title10, footnote1-footnote10**: Values of title1-title10 and footnote1-footnote10 from a TLF
- * **rowvar** (row variable description): Row heading in a table
- * **colvar** (column variable description): Column heading in a table
- * **repvar** (report variable): Value of a cell in a table (or data element in a figure)

Example of the Results Data Set

A Phase III Clinical Trial
DSMB Open Report - January 1, 9999
Visit Cutoff Date: April 1, 2015

Table 2. Demographics of Enrolled Participants by Site

	Site 1	Site 2	All Sites
Participants Enrolled	157	106	263
Participants with Demographics Form	157	106	263
Participant Age (years)			
N	157	106	263
Mean (SD)	26.6 (5.9)	27.6 (5.8)	27.0 (5.9)
Median	25.0	27.0	26.0
25th, 75th %tile	22, 30	23, 31	22, 31
Min, Max	18, 44	19, 44	18, 44
Participant Age			
Missing	0 (0%)	0 (0%)	0 (0%)
18-19 years	8 (5%)	2 (2%)	10 (4%)
20-24 years	64 (41%)	34 (32%)	98 (37%)
25-29 years	44 (28%)	32 (30%)	76 (29%)
30-34 years	23 (15%)	27 (25%)	50 (19%)
35-39 years	14 (9%)	7 (7%)	21 (8%)
40-45 years	4 (3%)	4 (4%)	8 (3%)

- * `prgmid =`
`/open/t_dem_site_blind.sas`
- * `dataset_created_from =`
`demout_site`
- * `number_of_titles = 4`
- * `title1-title4 =`
 - * `title1 =` A Phase III Clinical Trial
 - * `title2 =` DSMB Open Report – January 1, 9999
 - * `title3 =` Visit Cutoff Date: April 1, 2015
 - * `title4 =` Table 2. Demographics of Enrolled Participants by Site

Example of the Results Data Set

	rowvar	repvar	colvar
1	Participants Enrolled	157	Site 1
2	Participants Enrolled	106	Site 2
3	Participants Enrolled	263	All Sites
4	Participants with Demographics Form	157	Site 1
5	Participants with Demographics Form	106	Site 2
6	Participants with Demographics Form	263	All Sites
7	Participant Age (years)		Site 1
8	Participant Age (years)		Site 2
9	Participant Age (years)		All Sites
10	N	157	Site 1
11	N	106	Site 2
12	N	263	All Sites
13	Mean (SD)	26.6 (5.9)	Site 1
14	Mean (SD)	27.6 (5.8)	Site 2
15	Mean (SD)	27.0 (5.9)	All Sites
16	Median	25.0	Site 1
17	Median	27.0	Site 2
18	Median	26.0	All Sites
19	25th, 75th %tile	22, 30	Site 1
20	25th, 75th %tile	23, 31	Site 2
21	25th, 75th %tile	22, 31	All Sites
22	Min, Max	18, 44	Site 1
23	Min, Max	19, 44	Site 2
24	Min, Max	18, 44	All Sites
25	Participant Age		Site 1
26	Participant Age		Site 2
27	Participant Age		All Sites
28	Missing	0 (0%)	Site 1
29	Missing	0 (0%)	Site 2
30	Missing	0 (0%)	All Sites

First row of the Results Data Set:

- * rowvar = Participants Enrolled
- * colvar = Site 1
- * repvar = 157
 - * Corresponds to the Participants Enrolled row and Site 1 column in the table

Structure of the Results Data Set

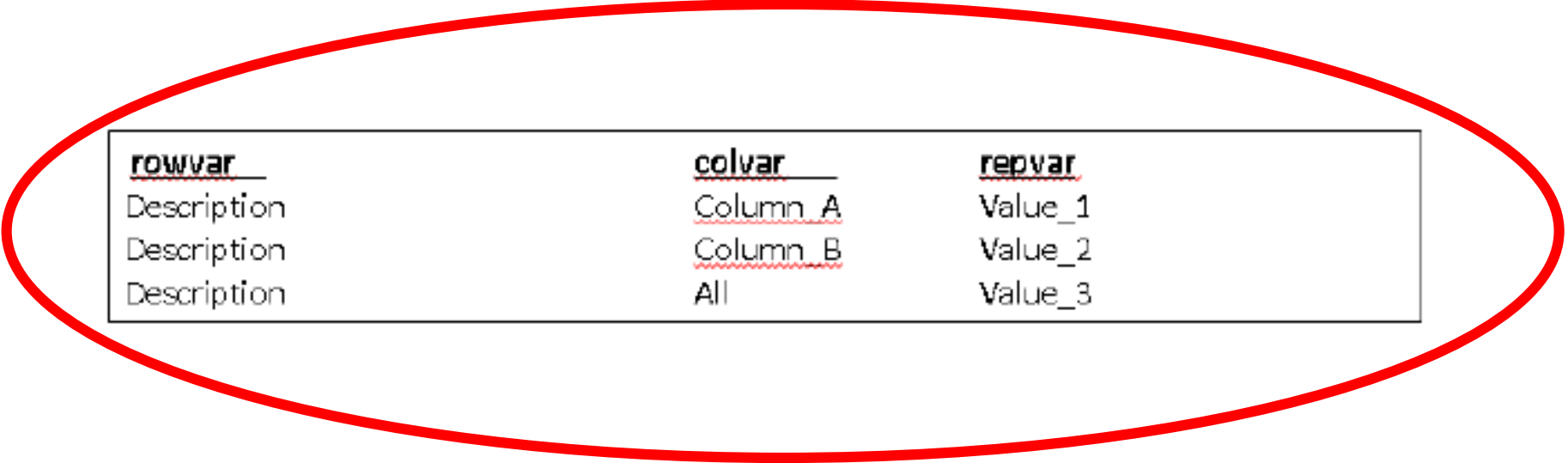
<u>rowvar</u>	<u>colvar</u>	<u>repvar</u>
Description	<u>Column_A</u>	Value_1
Description	<u>Column_B</u>	Value_2
Description	All	Value_3



	<u>Column_A</u>	<u>Column_B</u>	<u>All</u>
Description	Value_1	Value_2	Value_3

Structure of the Results Data Set

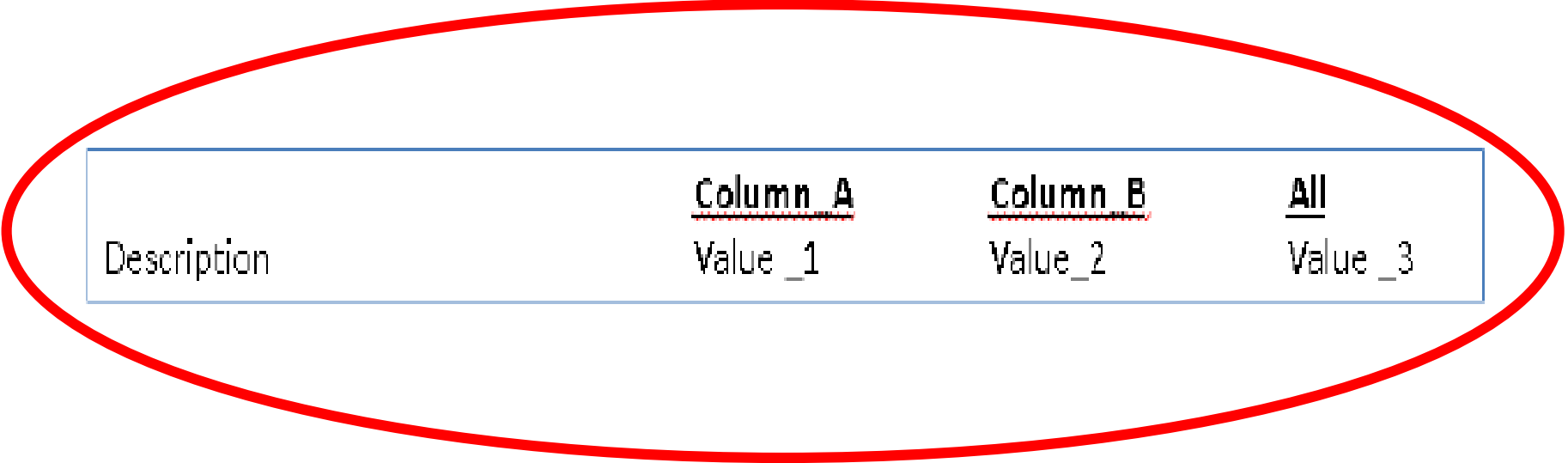
- * First box corresponds to the format the data takes in the Results Data Set



<u>rowvar</u>	<u>colvar</u>	<u>repvar</u>
Description	<u>Column_A</u>	Value_1
Description	<u>Column_B</u>	Value_2
Description	All	Value_3

Structure of the Results Data Set

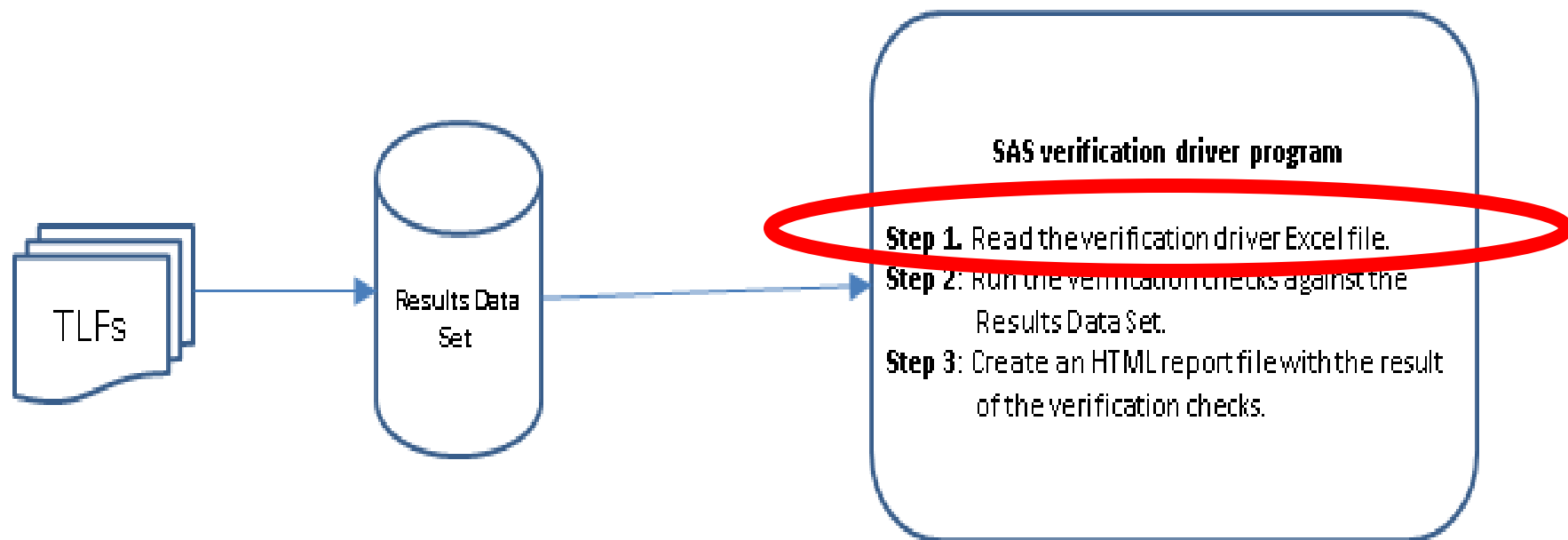
- * Second box corresponds to the output as it appears in the table used for the report



	<u>Column_A</u>	<u>Column_B</u>	<u>All</u>
Description	Value_1	Value_2	Value_3

Details of the SAS[®] Code

Step 1. Create the **Excel Verification Driver File**



Features of the Excel Verification Driver File (verification_checks.xlsx)

- * Manually created to define the verification checks to be made for each report

[illegible]

Create a SAS[®] dataset from the Excel Verification Driver File

```
PROC IMPORT
```

```
out=checks
```

```
datafile="Verification_checks.xlsx"
```


```
dbms=excelcs replace;
```

```
run;
```

Create “temp” dataset that adds macro variables to Status & Notes

```
DATA temp;  
  attrib Status length=$20 format=$20.  
    Notes length=$256 format=$256;  
    set checks(where=(^missing(Verification_check_id)));  
  Status =  
  strip(Status)||'&'||strip(Verification_check_id)||'_status';  
  Notes =  
  strip(Notes)||'&'||strip(Verification_check_id)||'_notes';  
run;
```

Example of “temp” dataset with added macro variables to Status & Notes



	status	Notes	Verification check ID	Description
1	&VC_0001_status	&VC_0001_notes	VC_0001	Total Enrolled from Accrual Summary by Site table should match Participants Enrolled when it occurs in other tables.
2	&VC_0002_status	&VC_0002_notes	VC_0002	Title 1 should match for all pages.
3	&VC_0003_status	&VC_0003_notes	VC_0003	Title 2 should match for all pages in open report.
4	&VC_0004_status	&VC_0004_notes	VC_0004	Title 2 should match for all pages in closed report.
5	&VC_0005_status	&VC_0005_notes	VC_0005	Title 3 should match for all pages.
6	&VC_0006_status	&VC_0006_notes	VC_0006	Check the SAS logs

Create **HTML Template File** using Proc Report and ODS HTML

```
ODS HTML file= "Verification_template.htm";
```

```
PROC REPORT data=temp headline headskip nowd ls=256
```

```
center style(report)=table[frame=below rules=rows];
```

```
column Obs Verification_check_ID Description Status Notes;
```

```
define Obs / 'Obs' center;
```

```
define Verification_check_ID /
```

```
    'Verification Check ID' center order style(column)=[cellwidth=2in just=center];
```

```
define Description /
```

```
    'Description' flow width=50 style(column)=[cellwidth=5in just=left] ;
```

```
define Status / 'Status' flow style(column)=[cellwidth=1in just=center];
```

```
define Notes / 'Notes' style(column)=[cellwidth=4in just=left] flow;
```

```
compute obs;
```

```
    cntl+1;
```

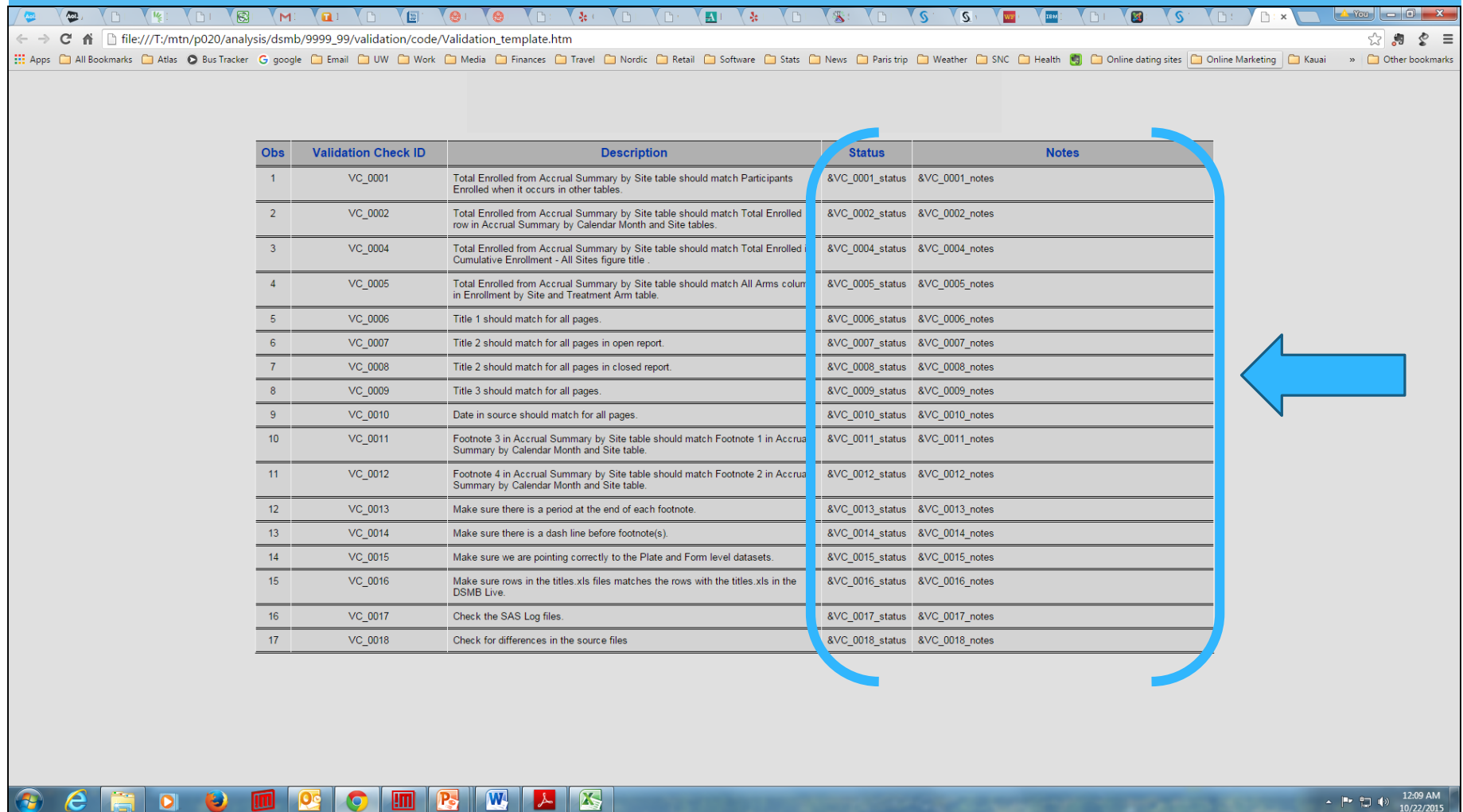
```
    obs = cntl;
```

```
endcomp;
```

```
run;
```

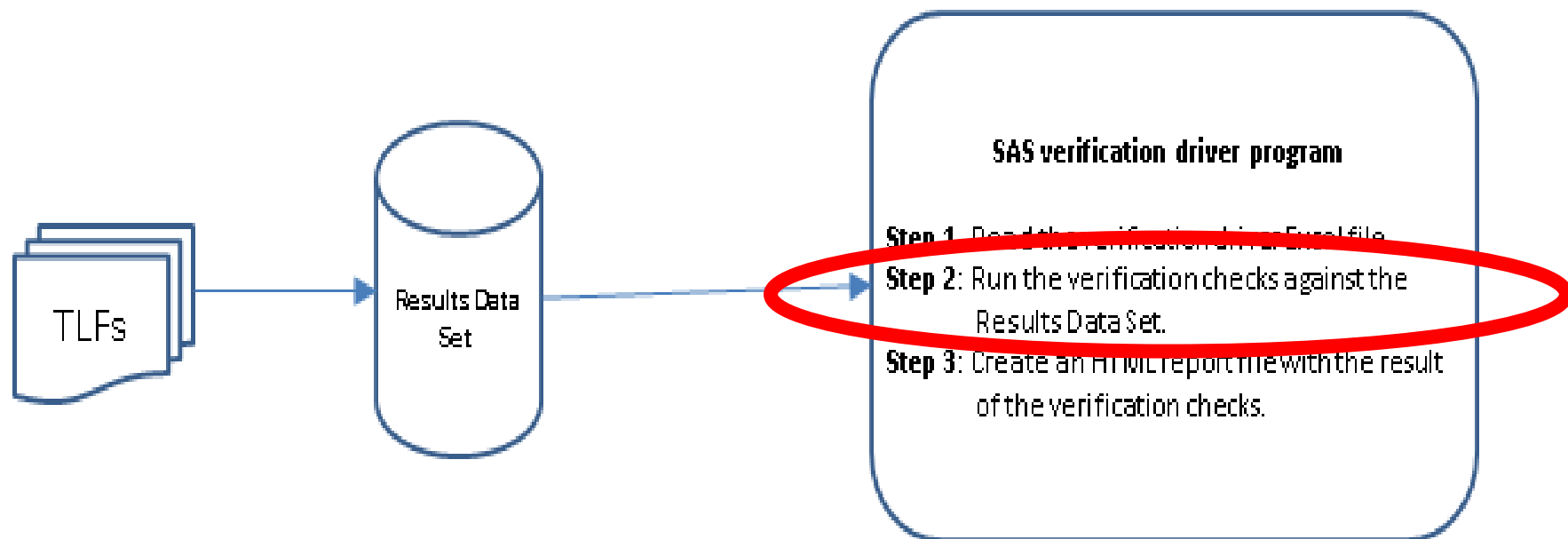
```
ODS HTML close;
```

Example of the HTML Template File



Obs	Validation Check ID	Description	Status	Notes
1	VC_0001	Total Enrolled from Accrual Summary by Site table should match Participants Enrolled when it occurs in other tables.	&VC_0001_status	&VC_0001_notes
2	VC_0002	Total Enrolled from Accrual Summary by Site table should match Total Enrolled row in Accrual Summary by Calendar Month and Site tables.	&VC_0002_status	&VC_0002_notes
3	VC_0004	Total Enrolled from Accrual Summary by Site table should match Total Enrolled Cumulative Enrollment - All Sites figure title .	&VC_0004_status	&VC_0004_notes
4	VC_0005	Total Enrolled from Accrual Summary by Site table should match All Arms column in Enrollment by Site and Treatment Arm table.	&VC_0005_status	&VC_0005_notes
5	VC_0006	Title 1 should match for all pages.	&VC_0006_status	&VC_0006_notes
6	VC_0007	Title 2 should match for all pages in open report.	&VC_0007_status	&VC_0007_notes
7	VC_0008	Title 2 should match for all pages in closed report.	&VC_0008_status	&VC_0008_notes
8	VC_0009	Title 3 should match for all pages.	&VC_0009_status	&VC_0009_notes
9	VC_0010	Date in source should match for all pages.	&VC_0010_status	&VC_0010_notes
10	VC_0011	Footnote 3 in Accrual Summary by Site table should match Footnote 1 in Accrual Summary by Calendar Month and Site table.	&VC_0011_status	&VC_0011_notes
11	VC_0012	Footnote 4 in Accrual Summary by Site table should match Footnote 2 in Accrual Summary by Calendar Month and Site table.	&VC_0012_status	&VC_0012_notes
12	VC_0013	Make sure there is a period at the end of each footnote.	&VC_0013_status	&VC_0013_notes
13	VC_0014	Make sure there is a dash line before footnote(s).	&VC_0014_status	&VC_0014_notes
14	VC_0015	Make sure we are pointing correctly to the Plate and Form level datasets.	&VC_0015_status	&VC_0015_notes
15	VC_0016	Make sure rows in the titles.xls files matches the rows with the titles.xls in the DSMB Live.	&VC_0016_status	&VC_0016_notes
16	VC_0017	Check the SAS Log files.	&VC_0017_status	&VC_0017_notes
17	VC_0018	Check for differences in the source files	&VC_0018_status	&VC_0018_notes

Step 2. Run the **Verification Checks** against the Results Data Set



Create separate SAS[®] programs for each of the **Verification Checks**

- * The **Verification Checks** are detailed in the **Excel Verification Driver File**.
- * Each program has a specific verification aim, such as confirming that the enrollment totals match across all TLFs.
 - * Accomplished by comparing all occurrences of the relevant data element within the **Results Data Set** and determining whether the values of those cells satisfy the conditions of the check.
 - * During this process the values for the macro variables **&VC_<IDnumber>_status** and **&VC_<IDnumber>_notes** are created and stored.

Example of a **Verification Check** of the enrollment numbers

- * Total enrollment generally appears in tables in a row with the heading of “Participants Enrolled”, and occurs in a column corresponding to All Sites.
- * Using the **Results Data Set**, called work.save_all, subset on the relevant observations for this check where the rowvar value equals “Participants Enrolled” and the colvar value equals ‘All Sites’ = 999.

```
DATA VC_0001;  
set work.save_all;  
    if upcase(strip(ROWVAR)) =: 'PARTICIPANTS ENROLLED'  
    and colvar = 999;  
run ;
```

Example of a **Verification Check** of the enrollment number

- * Create a new variable called num_repvar as a numeric version of repvar, a character variable in the **Results Data Set**

```
data all ;  
length num_repvar 8;  
set VC_0001 end=eof ;  
    num_repvar = input(strip(repvar),10.);  
keep prgmid varchr colvar num_repvar repvar ;  
run ;
```

Example of a **Verification Check** of the enrollment numbers

- * Create a PDF file that summarizes the enrollment numbers found in the **Results Data Set**

```
ODS pdf file = "VC_0001.pdf" bookmarklist=hide notoc;  
title "VC_0001: Total Enrolled from Accrual Summary by Site table should match  
Participants Enrolled when it occurs in other tables";
```

```
PROC REPORT data=all nowd ls=256;  
column prgmid varchar colvar num_repvar;  
    define prgmid / 'Source File' display style(column)=[cellwidth=6in just=left];  
    define varchar / 'Description' display style(column)=[cellwidth=1.5in just=right];  
    define colvar / 'Column' display style(column)=[cellwidth=1in just=right];  
    define num_repvar / 'Result' display style(column)=[cellwidth=1in just=right];  
  
run;  
ODS pdf close;
```

Example of the PDF file created from the **Verification Check**

VC_0001: Total Enrolled from Accrual Summary by Site table should match Participants Enrolled when it occurs in other tables

Source File	Description	Column	Result
closed/t_anal_sex_arm.sas	Participants Enrolled	999	263
open/t_anal_sex_country.sas	Participants Enrolled	999	263
closed/t_base_contra_dsmb_arm.sas	Participants Enrolled	999	263
open/t_base_contra_dsmb_country.sas	Participants Enrolled	999	263
closed/t_dem_arm.sas	Participants Enrolled	999	263
open/t_dem_country.sas	Participants Enrolled	999	263
closed/t_fu_soc_harms_arm.sas	Participants Enrolled	999	263
closed/t_fu_soc_harms_country.sas	Participants Enrolled	999	263
closed/t_fu_soc_harms_site.sas	Participants Enrolled	0.9	263
closed/t_hiv_inc_arm.sas	Participants Enrolled	999	263
closed/t_hiv_inc_country.sas	Participants Enrolled	999	254

Example of a **Verification Check** of the enrollment numbers

- * The check compares the minimum and maximum values of the num_repvar variables using Proc SQL
- * If the minimum and maximum are equal then all values must be identical so it will pass, otherwise it fails

```
PROC SQL noprint;
```

```
    select min(num_repvar)- max(num_repvar) into :diff  
from VC_0001;  
quit;
```

Example of a **Verification Check** of the enrollment numbers

- * If the minimum and maximum are equal then an %IF statement gives the &VC_0001_status macro variable the value of “Passed”, otherwise it gives the value of “Failed”.

- * Assign an HTML link containing a path to the PDF summary file

%MACRO check;

 %global **VC_0001_status** VC_0001_notes;

 %let VC_0001_notes = ¬es;

 %if **&diff = 0** %then %let **VC_0001_status**=Passed;

 %else %let **VC_0001_status**=Failed;

%MEND check;

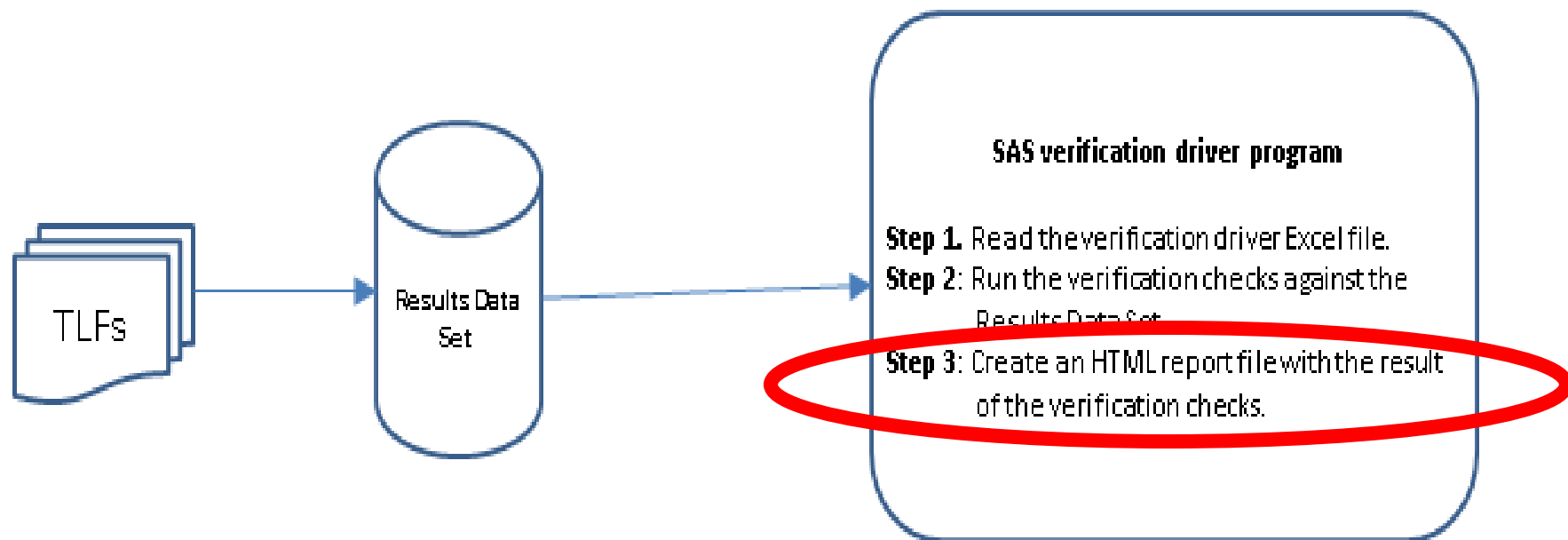
%check;

Keep in mind the results as seen in the PDF file

VC_0001: Total Enrolled from Accrual Summary by Site table should match Participants Enrolled when it occurs in other tables

Source File	Description	Column	Result
closed/t_anal_sex_arm.sas	Participants Enrolled	999	263
open/t_anal_sex_country.sas	Participants Enrolled	999	263
closed/t_base_contra_dsmb_arm.sas	Participants Enrolled	999	263
open/t_base_contra_dsmb_country.sas	Participants Enrolled	999	263
closed/t_dem_arm.sas	Participants Enrolled	999	263
open/t_dem_country.sas	Participants Enrolled	999	263
closed/t_fu_soc_harms_arm.sas	Participants Enrolled	999	263
closed/t_fu_soc_harms_country.sas	Participants Enrolled	999	263
closed/t_fu_soc_harms_site.sas	Participants Enrolled	0.9	263
closed/t_hiv_inc_arm.sas	Participants Enrolled	999	263
closed/t_hiv_inc_country.sas	Participants Enrolled	999	254

Step 3. Create an **HTML Report File** with the result of the verification checks



Update the HTML Template File and make the Validation Dashboard

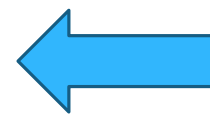
- * Once all of the verification checks specified in the Excel Verification Driver File have run, we update our HTML Template File and make the Validation Dashboard.
- * We update it with the macro variables &VC_<IDnumber>_status and &VC_<IDnumber>_notes stored for each check as a global macro variables

Update the **HTML Template File** and make the **Validation Dashboard**

```
%MACRO read_htm (webin,webout);  
    filename web_in "&webin";  
    filename web_out "&webout";  
    DATA _null_;  
        length line $ 256;  
        infile web_in length=lv;  
        file web_out;  
        input @1 line $varying200. lv;  
        line=tranwrd(line,'&','&');  
        line = trim(resolve(line));  
        put line;  
  
    run;  
    filename web_in;  
    filename web_out;  
%MEND read_htm;  
  
%read_htm (webin= Validation_template.htm, webout=Validation_dashboard.htm );
```

Update the **HTML Template File** and make the **Validation Dashboard**

```
%MACRO read_htm (webin,webout);  
    filename web_in "&webin";  
    filename web_out "&webout";  
    DATA _null_;  
        length line $ 256;  
        infile web_in length=lv;  
        file web_out;  
        input @1 line $varying200. lv;  
        line=tranwrd(line,'&','&');  
        line = trim(resolve(line));  
        put line;  
  
    run;  
    filename web_in;  
    filename web_out;  
%MEND read_htm;  
  
%read_htm (webin= Validation_template.htm, webout=Validation_dashboard.htm );
```



Required since
HTML code shows
& instead of &

Update the **HTML Template File** and make the **Validation Dashboard**

```
%MACRO read_htm(webin,webout);  
    filename web_in "&webin";  
    filename web_out "&webout";  
    DATA _null_;  
        length line $ 256;  
        infile web_in length=lv;  
        file web_out;  
        input @1 line $varying200. lv;  
        line=tranwrd(line,'&','&');  
        line = trim(resolve(line));  
        put line;  
    run;  
    filename web_in;  
    filename web_out;  
%MEND read_htm;  
  
%read_htm (webin= Validation_template.htm, webout=Validation_dashboard.htm );
```

Resolves the macro
variables defined as:

VC_0001_status=Passe
d or VC_0001_status=Failed
;

Example of the Validation Dashboard

[illegible]

Limitations

- * Verification checks are made on the SAS® data sets used to create the TLFs, rather than on the text contained in the PDF output of the TLFs themselves.
- * If one of the TLFs in the report is created by a SAS® program that is not included in the checks for that report, or if the data set that creates that TLF is not part of the Results Data Set, then it is possible for discrepancies from that TLF to go unnoticed.
- * Issues can also arise that we did not anticipate when writing the logic of the verification checks, and which subsequently are not caught by the automated process. Therefore, even when the checks identify failures, manual effort is still required to identify the source of those failures and to make the necessary corrections.

Conclusions

- * Proved to be extremely helpful in completing the review process necessary to produce large and complex DSMB reports with high accuracy and integrity.
- * By focusing on automating verification checks that were simple as well as those that would have been time consuming to do manually, we were able to focus our attention on the more complicated aspects of preparing the reports.
- * Finally, we have incorporated the SAS[®] automated cross-report verification system into our daily crons and it has also proven to be useful in identify programming bugs and other problems early on in our daily reporting process.

Thank you!

