

%batchrunprograms:

SAS Studio Background Submit on Multiple Programs

Luis Andre Soriano

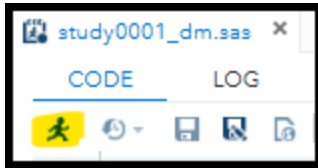
Statistical Programmer

- Introduction
 - Importance of a clean full run
 - Sequential, independent, and frequent clean full run on multiple programs
 - Lack of built-in functionality within SAS Studio
 - Current solutions and their issues

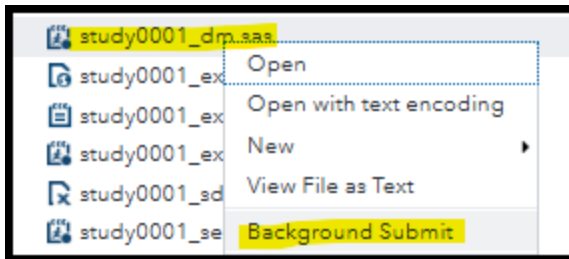
• **%batchrunprograms**: The Solution

- Extra functionalities of the tool
- Conclusion

- Running a program:
 - Interactive



- Clean full run



- A clean full run:
 - e.g. “Batch Submit” in locally-installed SAS
 - e.g. “Background Submit” in SAS Studio
 - Other ways based on structure
- Importance of clean full run vs interactive run
 - Independence from other programs
 - Reproducible results
 - True and consistent errors/warnings
 - Logs and results are saved as external file

- Quick to do sequential, independent, and frequent clean full runs with few programs
- Right-click > Batch Submit/Background Submit on programs one-by-one
- But typical projects include multiple runs of multiple programs
- Tedious
- Time-consuming
- Error prone

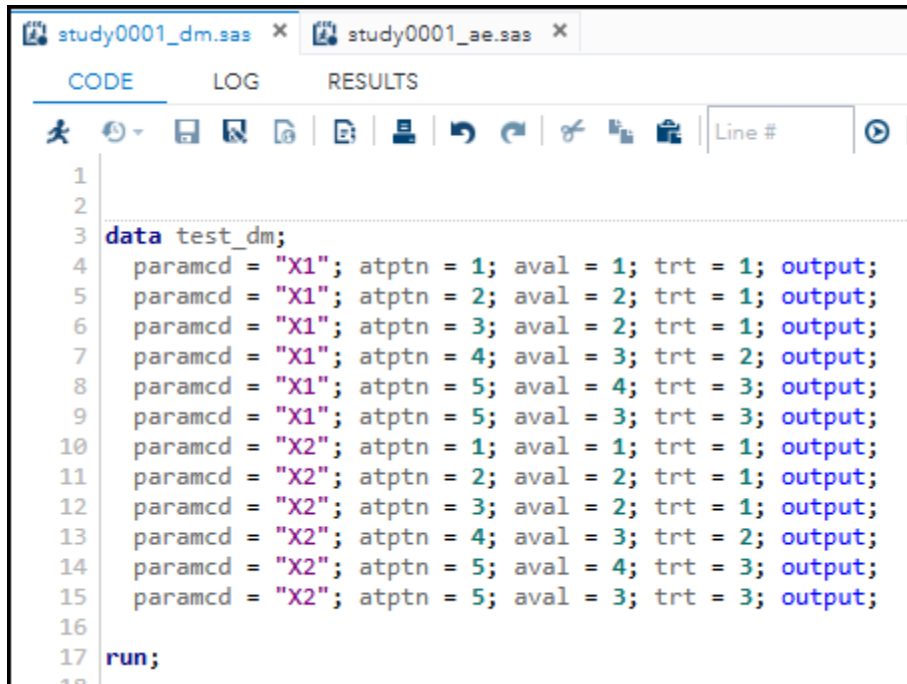
- Highly recommended:
 - To have list of programs
 - To have clear sequence of run order
 - To perform clean full run on all
- Easy to implement in locally-installed SAS (e.g. .bat files)
- Currently no easy way in server-based SAS like SAS Studio
- Some solutions include:
 - Enabling XCMD option (security risk on servers)
 - Creating custom tools outside SAS (requires a separate skill and budget)
 - %include on multiple programs
 - (has risk of one program affecting another program unintentionally)
 - (not able to save the log and result of each program)

- %batchrunprograms

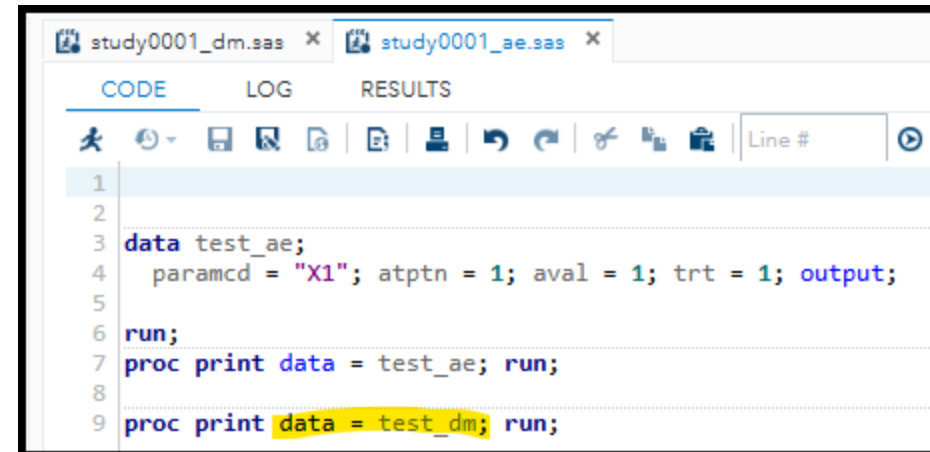
- Maintained list(s) of programs to run

1	PATH	PROGNAME	FAILSTOP	DONOTRUN
2	.	study0001_dm.sas	X	
3	.	study0001_se		X
4	.	study0001_sv		
5	.	study0001_ex		
6	.	study0001_fa		
7	.	study0001_ae	X	
8	.	study0001_tplot1		

- Every program runs on a fresh and clean session
- Does not affect other programs

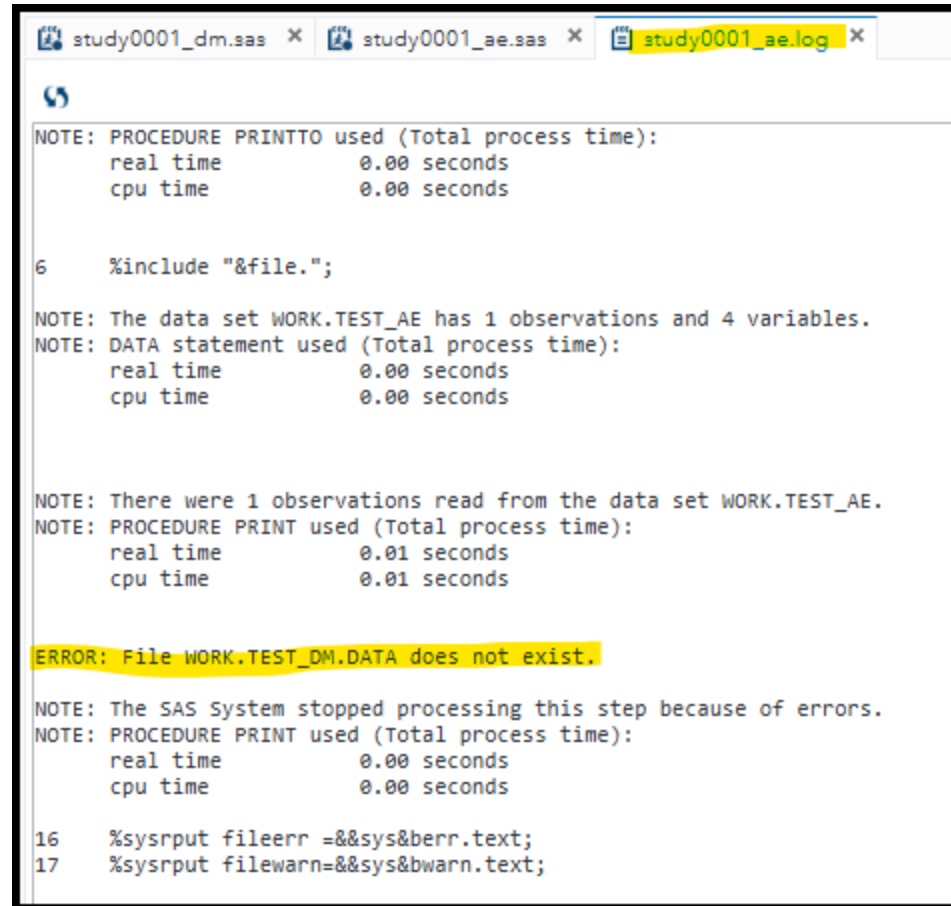


```
1  
2  
3 data test_dm;  
4   paramcd = "X1"; atptn = 1; aval = 1; trt = 1; output;  
5   paramcd = "X1"; atptn = 2; aval = 2; trt = 1; output;  
6   paramcd = "X1"; atptn = 3; aval = 2; trt = 1; output;  
7   paramcd = "X1"; atptn = 4; aval = 3; trt = 2; output;  
8   paramcd = "X1"; atptn = 5; aval = 4; trt = 3; output;  
9   paramcd = "X1"; atptn = 5; aval = 3; trt = 3; output;  
10  paramcd = "X2"; atptn = 1; aval = 1; trt = 1; output;  
11  paramcd = "X2"; atptn = 2; aval = 2; trt = 1; output;  
12  paramcd = "X2"; atptn = 3; aval = 2; trt = 1; output;  
13  paramcd = "X2"; atptn = 4; aval = 3; trt = 2; output;  
14  paramcd = "X2"; atptn = 5; aval = 4; trt = 3; output;  
15  paramcd = "X2"; atptn = 5; aval = 3; trt = 3; output;  
16  
17 run;
```



```
1  
2  
3 data test_ae;  
4   paramcd = "X1"; atptn = 1; aval = 1; trt = 1; output;  
5  
6 run;  
7 proc print data = test_ae; run;  
8  
9 proc print data = test_dm; run;
```

- Every program runs on a fresh and clean session
- Does not affect other programs



```
study0001_dm.sas x study0001_ae.sas x study0001_ae.log x
NOTE: PROCEDURE PRINTTO used (Total process time):
      real time          0.00 seconds
      cpu time           0.00 seconds

6      %include "&file.";

NOTE: The data set WORK.TEST_AE has 1 observations and 4 variables.
NOTE: DATA statement used (Total process time):
      real time          0.00 seconds
      cpu time           0.00 seconds

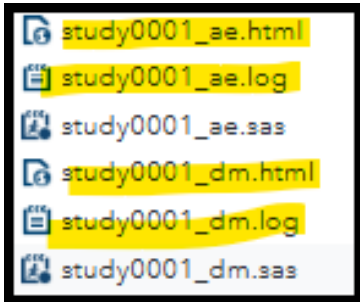
NOTE: There were 1 observations read from the data set WORK.TEST_AE.
NOTE: PROCEDURE PRINT used (Total process time):
      real time          0.01 seconds
      cpu time           0.01 seconds

ERROR: File WORK.TEST_DM.DATA does not exist.

NOTE: The SAS System stopped processing this step because of errors.
NOTE: PROCEDURE PRINT used (Total process time):
      real time          0.00 seconds
      cpu time           0.00 seconds

16     %sysrput fileerr =&&sys&berr.text;
17     %sysrput filewarn=&&sys&bwarn.text;
```

- Log file saved
- Result file saved



- Summary of the batch run

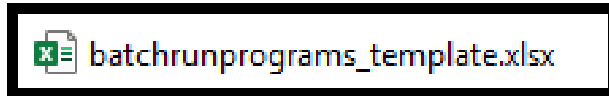
Results of Batch Run for:
./study0001_sdtm.xlsx
Run: 2025-09-08T14:31:05

Obs	Excel Row	PATH	PROGNAME	RESULTS
1	2	.	study0001_dm	Has warning(s).
2	4	.	study0001_sv	Has error(s).
3	5	.	study0001_ex	Successful Run.
4	7	.	study0001_ae	Has error(s).

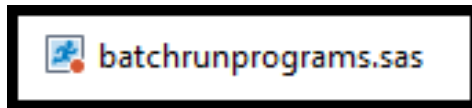
- Can **stop** the batch run in case of errors on critical programs
- Can **skip** programs

PATH	PROGNAME	FAILSTOP	DONOTRUN
.	study0001_dm.sas	X	
.	study0001_se		X
.	study0001_sv		
.	study0001_ex		
.	study0001_fa		
.	study0001_ae	X	
.	study0001_tplot1		

- Preparation:
 - Save the accompanying Excel file template



- Make sure %batchrunprograms macro program is ready (e.g. autocall)



- Preparation:

PATH	PROGNAME	FAILSTOP	DONOTRUN
.	create_combined_input_sdtm	X	
P:\Projects\Client\Study ID\Development\Programs	create_dummied_input_sdtm	X	
P:\Projects\Client\Study ID\Development\Programs	tdomains_to_sas	X	
P:/Projects/Client/Study ID/Development/Programs	create_sdtm_dm	X	
.	create_sdtm_se.sas	X	
.	create_sdtm_sv.sas	X	
.	create_sdtm_ae		
.	create_sdtm_ag		
.	create_sdtm_cm		
.	create_sdtm_cp		
.	create_sdtm_ds		
.	create_sdtm_dv		X
.	create_sdtm_ec		X

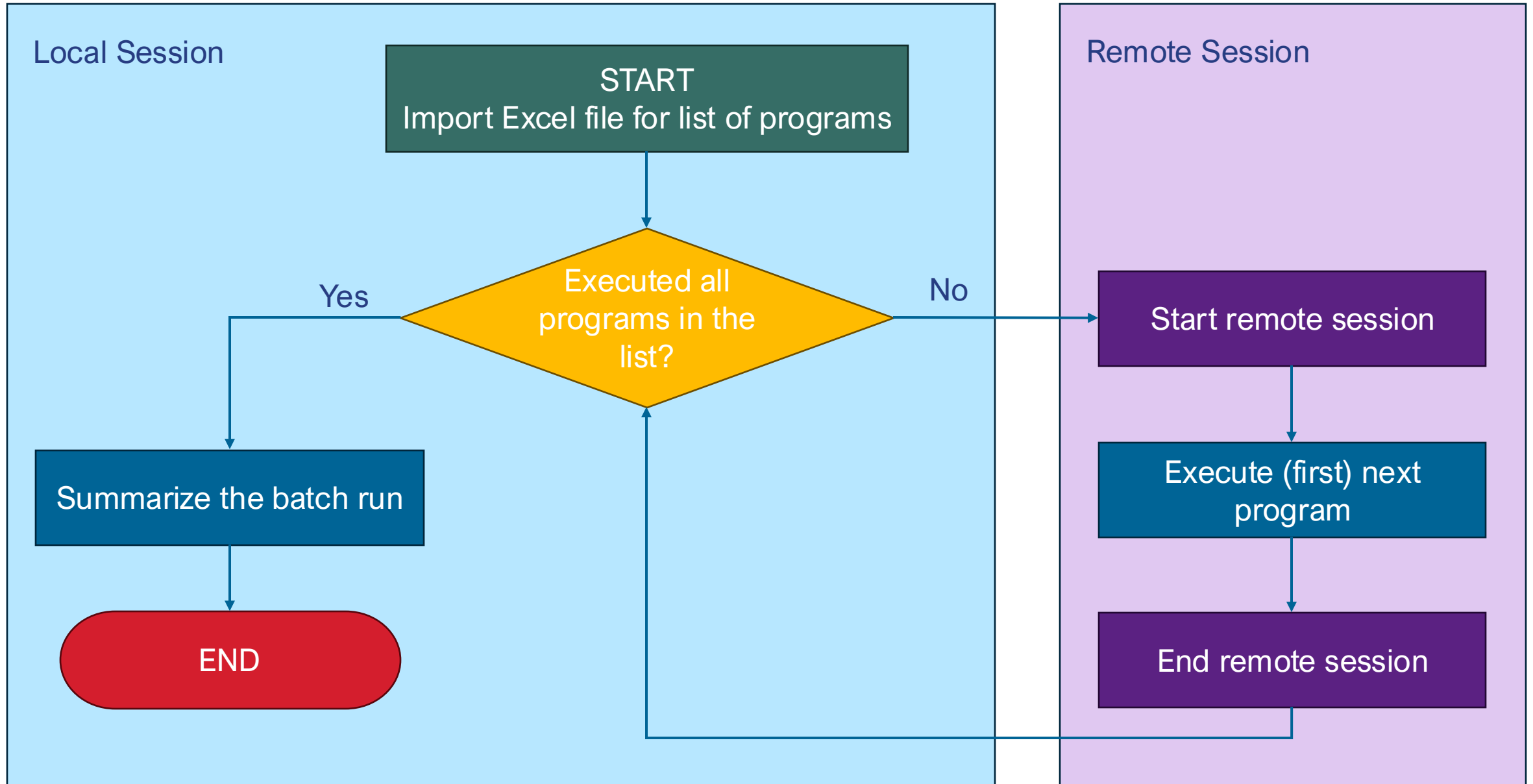
- Update the Excel file to list all programs
- Paths can be absolute or relative
- Paths can have forward slash (/) or backward slash (\)
- FAILSTOP = “X” to stop if error encountered or file does not exist
- DONOTRUN = “X” to skip program
- Tip: you can have multiple Excel files for different purposes.

- Preparation:

```
%batchrunprograms (proglst=._sdtm.xlsx);  
%batchrunprograms (proglst=P:/Projects/Client/Study ID/Development/Programs/_adam.xlsx);
```

- Invoke the %batchrunprograms to point to the Excel file(s).
- Relative paths work.
- NOTE: Relative paths (both in Excel file(s) and in the macro call should of course be set up properly in your SAS session).

- Process:
 - To achieve complete independence, %batchrunprograms uses SAS/CONNECT functionalities
- Each program has its own separate remote session
 - Remote session: a session outside the local session
 - A clean session that is not affected by the local session
 - Can be closed once a program has been run on it



For each program, the process is as follows:

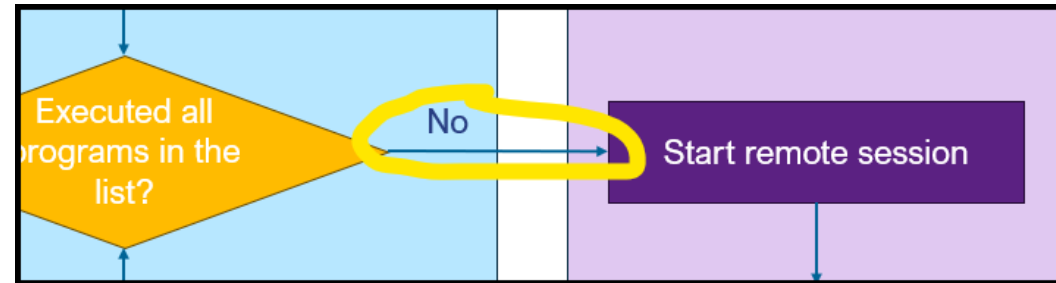
1. Open a clean remote session and sign on to it.

```
signon remote0 sascmd="!sascmd -nosyntaxcheck -noterminal" cmacvar=remote0_chk;
```

- Will not inherit the temporary objects from the local session (e.g. macro vars, datasets, options)
- Does not need a true remote server
- Does not need user ID and password
- This is only a separate SAS process on the same machine

2. Supply the necessary information from the local session to the remote session.

```
%syslput file      = &&file&i.;  
%syslput log       = &&log&i.;  
%syslput output    = &&output&i.;  
%syslput berr      = &berr.;  
%syslput bwarn     = &bwarn.;
```



- Current information of the iteration (of the local session loop):
 - SAS program name
 - Log name
 - Result name
 - Error and warning flags
- Shows local session info -> remote session.

3. Begin the submission of codes in the remote session.

```
rsubmit remote0 wait=yes;
```

- The local session will wait for the remote session to finish before continuing.
- All codes after this line will be submitted on the remote session.

4. Configure the ODS to store the results in HTML. Save it as an external file.

```
ods listing close;  
ods html file = "&output." gpath = "%sysfunc(getoption(work))";  
ods graphics on;
```

- Mimics the default behavior of normal SAS Studio Background Submit.
- Can be changed to whatever format preferred.
- GPATH and ODS GRAPHICS important to check because of varying behavior of local vs remote.
- PROC OPTIONS
- Important for figure programs.

5. Redirect the remote session log into an external file.

```
proc printto log="&log." new; run;
```

- By default, printed on the log of local session.
- But log of each remote session is desired.

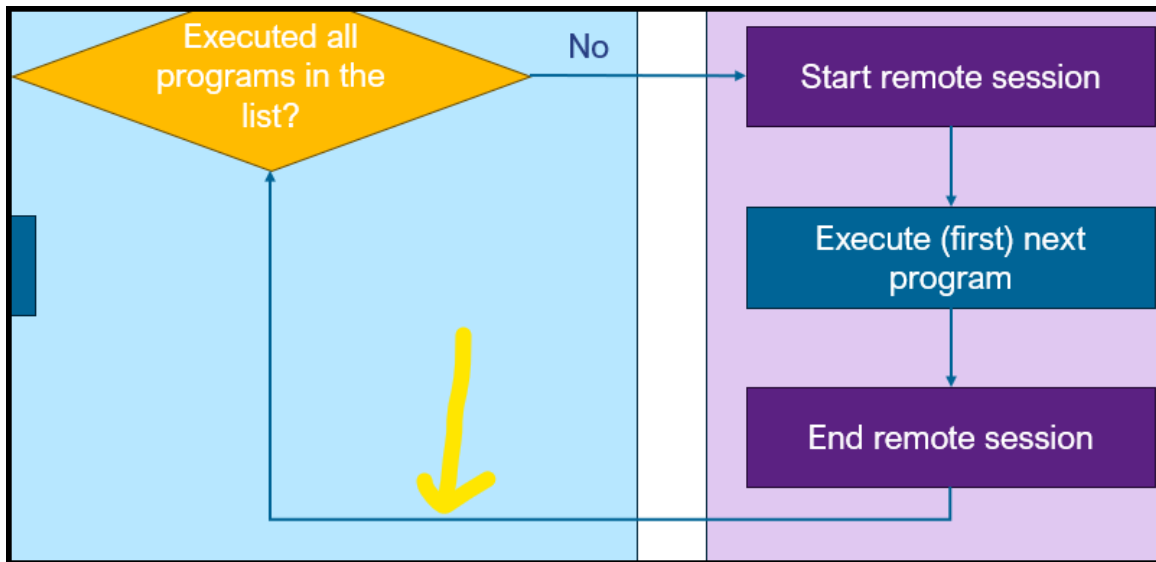
6. Execute the entire program of current iteration.

```
%include "&file.";
```

- All process will be executed immediately.
- Will not wait for the local session (and loop) to finish.
- Reason: The next programs in the loop might need some files from the previous iterations (e.g. ADSL data is needed by ADAE).

7. Capture the last error text and last warning text from the remote session and store it in the local session.

```
%nrstr(%sysrput fileerr =&&sys&berr.text);  
%nrstr(%sysrput filewarn=&&sys&bwarn.text);
```



- To report the status of each program run.
- Shows remote session info -> local session.

8. End the submission of codes in the remote session.

```
endrssubmit;  
waitfor _all_;
```

- All codes after this line will be submitted on the local session.

9. Sign off from the remote session and close it.

```
signoff _all_;
```

SUMMARY OF THE BATCH RUN

Results of Batch Run for:
./study0001_sdtm.xlsx
Run: 2025-09-03T12:09:00

Obs	Excel Row	PATH	PROGNAME	RESULTS
1	2	.	study0001_dm	Has warning(s).
2	4	.	study0001_sv	Has error(s).
3	5	.	study0001_ex	Successful Run.
4	7	.	study0001_ae	Successful Run.
5	8	.	study0001_tplot1	Successful Run.

STOPPING THE BATCH RUN

1	PATH	PROGNAME	FAILSTOP	DONOTRUN
2	.	study0001_dm.sas	X	
3	.	study0001_se		X
4	.	study0001_sv		
5	.	study0001_ex		
6	.	study0001_fa		
7	.	study0001_ae	X	
8	.	study0001_tplot1		

Results of Batch Run for:
./study0001_sdtm.xlsx
Run: 2025-09-03T11:46:41

Obs	Excel Row	PATH	PROGNAME	RESULTS
1	2	.	study0001_dm	Has warning(s).
2	4	.	study0001_sv	Has error(s).
3	5	.	study0001_ex	Successful Run.
4	6	.	study0001_ae	Has error(s).

```
WARNING: [redacted]/Development/Programs/study0001_fa.sas
WARNING: is not a critical file (FAILSTOP is not X) and does not exist.
WARNING: Utility will not run this program but will proceed to next programs.
WARNING: ( Excel row = 6 )
```

```
ERROR: [redacted]/Development/Programs/study0001_ae.sas
ERROR: is a critical file and has an error.
ERROR: Utility will STOP now.
```

SKIPPING PROGRAMS

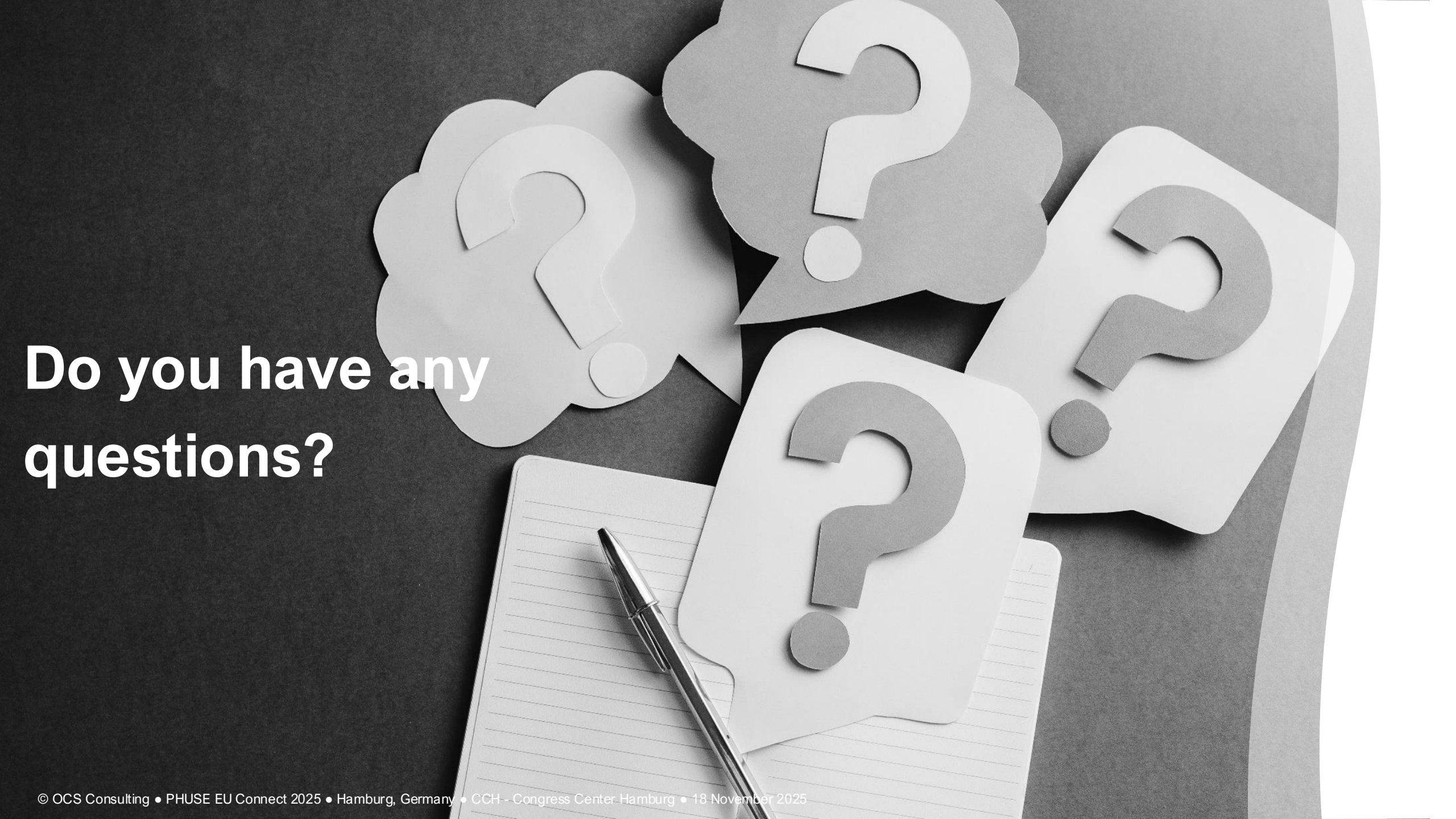
1	PATH	PROGNAME	FAILSTOP	DONOTRUN
2	.	study0001_dm.sas	X	
3	.	study0001_se		X
4	.	study0001_sv		
5	.	study0001_ex		
6	.	study0001_fa		
7	.	study0001_ae	X	
8	.	study0001_tplot1		

Results of Batch Run for:
./study0001_sdtm.xlsx
Run: 2025-09-03T11:46:41

Obs	Excel Row	PATH	PROGNAME	RESULTS
1	2	.	study0001_dm	Has warning(s).
2	4	.	study0001_sv	Has error(s).
3	5	.	study0001_ex	Successful Run.
4	6	.	study0001_ae	Has error(s).

- **%batchrunprograms**

- Has list(s) of programs
- Executes programs sequentially, independently, whenever desired
- Runs each program in separate and clean remote session
- Saves the log
- Saves the result
- Provides summary of the batch run
- Can skip programs
- Can stop the batch run
- SAS/CONNECT only requirement (no need physical/virtual remote servers)



**Do you have any
questions?**

Your comments and questions are valued and encouraged. Contact the author at:

- **Author Name:** Luis Andre Soriano
- **Company:** OCS Consulting B.V.
- **Address:** Ruwekampweg 2 g, 5222 AT, 's-Hertogenbosch, The Netherlands
- **Work Phone:** +31 (0)73 523 6000
- **Email:** luisandre.soriano@ocs-consulting.com
- **Website:** www.ocs-lifesciences.com