

Breaking Boundaries:

Working across different workspaces in LSAF

Kai Wanke, OCS Life Sciences

Application Development – AD09

PHUSE EU Connect November 2020

Background

- Our client is a large pharmaceutical company that regularly performs a large number of clinical trials

Background

- Our client is a large pharmaceutical company that regularly performs a large number of clinical trials
- The patient data of clinical trials is received from vendors on a regular basis and converted into SDTM

Background

- Our client is a large pharmaceutical company that regularly performs a large number of clinical trials
- The patient data of clinical trials is received from vendors on a regular basis and converted into SDTM
- We develop various solutions to automate and accelerate this process as much as possible using LSAF Jobs

General Introduction

Repository

Repository

Hands-On Demos

Training Organization

Users

Text to filter

3 items

☐	T ↑	Name ↑
☐		Hands-On Demos
☐		Training Organization
☐		Users

- Permanent
- Version controlled
- Accessible but not modifiable

General Introduction

Repository

Repository

Hands-On Demos

Training Organization

Users

Text to filter

3 items

☐	T ↑	Name	↑
☐		Hands-On Demos	
☐		Training Organization	
☐		Users	

- Permanent
- Version controlled
- Accessible but not modifiable

User Workspace

Workspace

Hands-On Demos

Text to filter

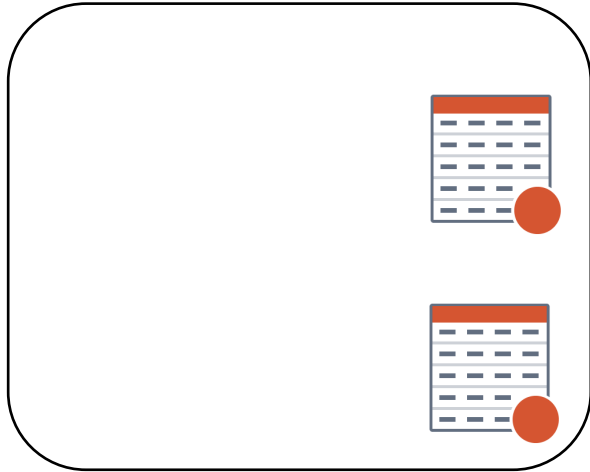
1 items

☐	T ↓	Name	
☐		Hands-On Demos	

- Changeable
- User created
- 'Sandbox' environment

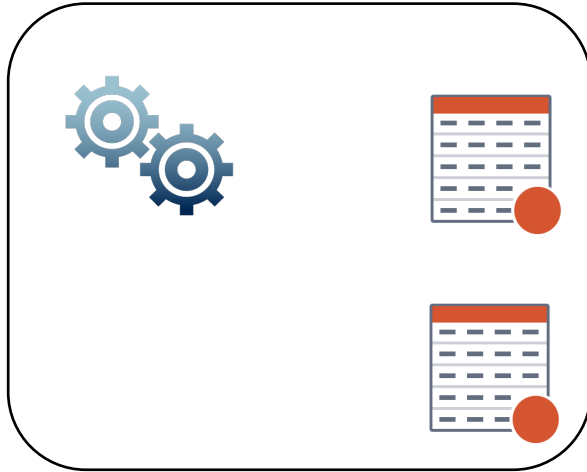
General Introduction

Repository

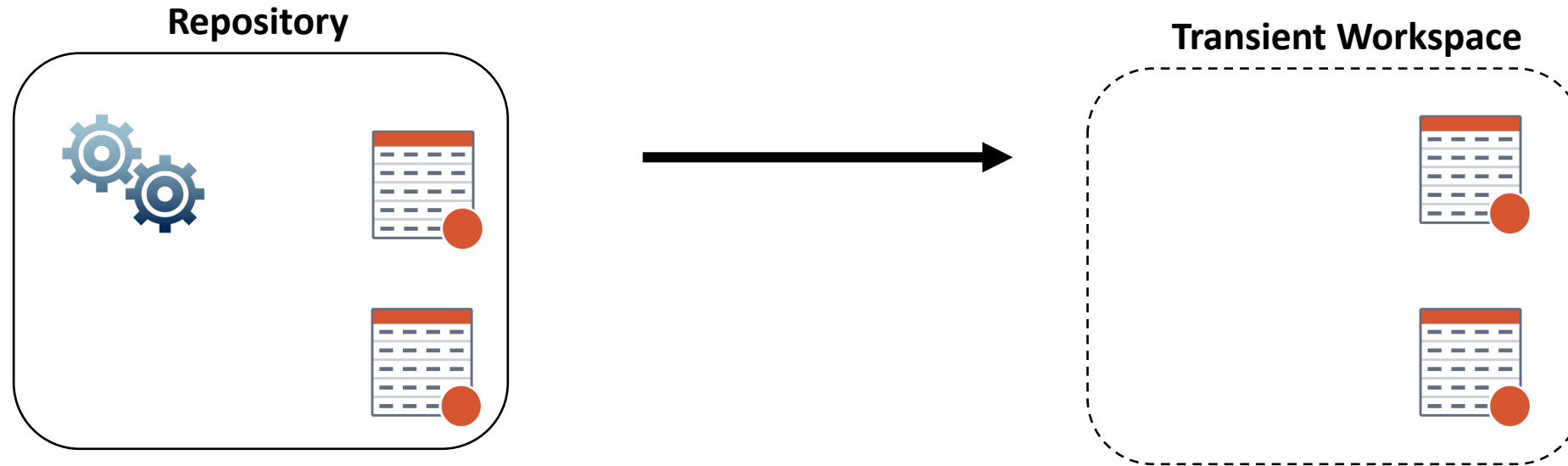


General Introduction

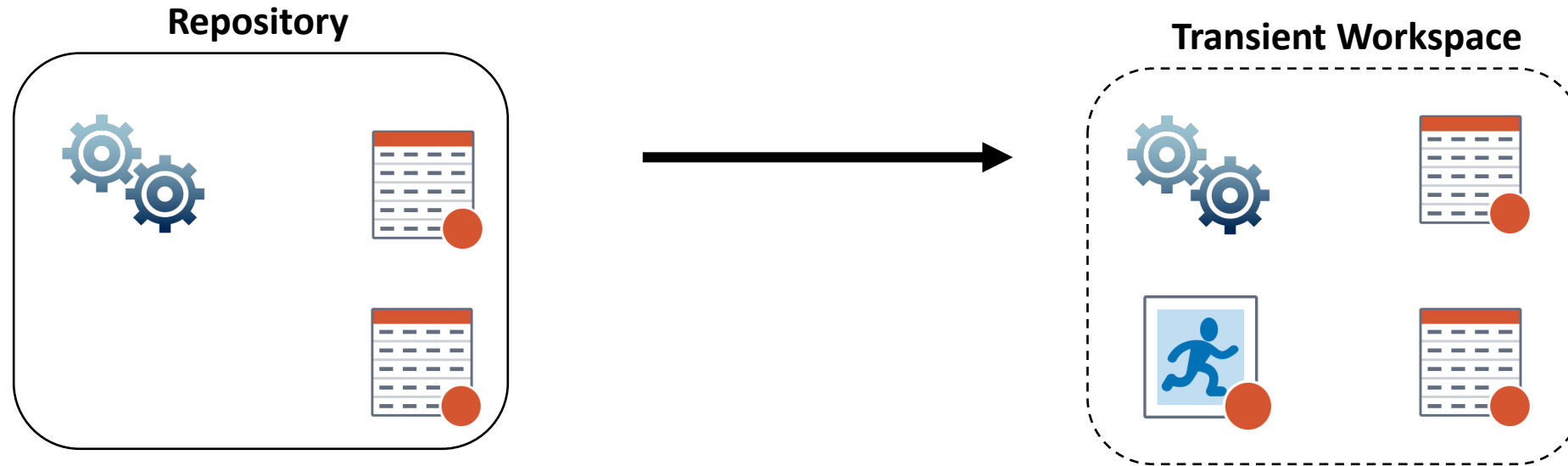
Repository



General Introduction

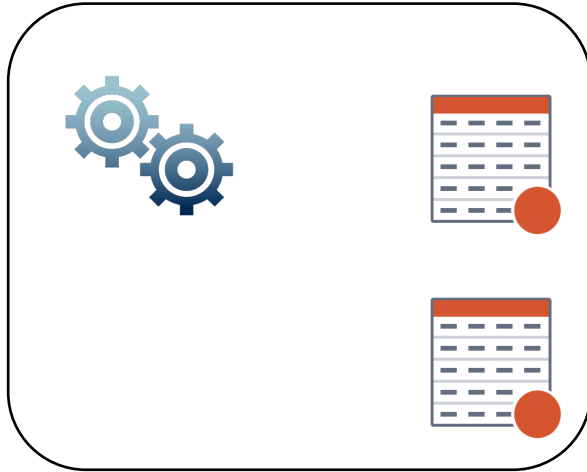


General Introduction

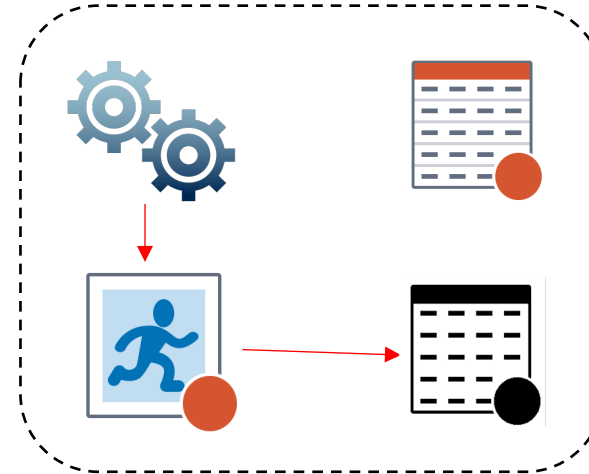


General Introduction

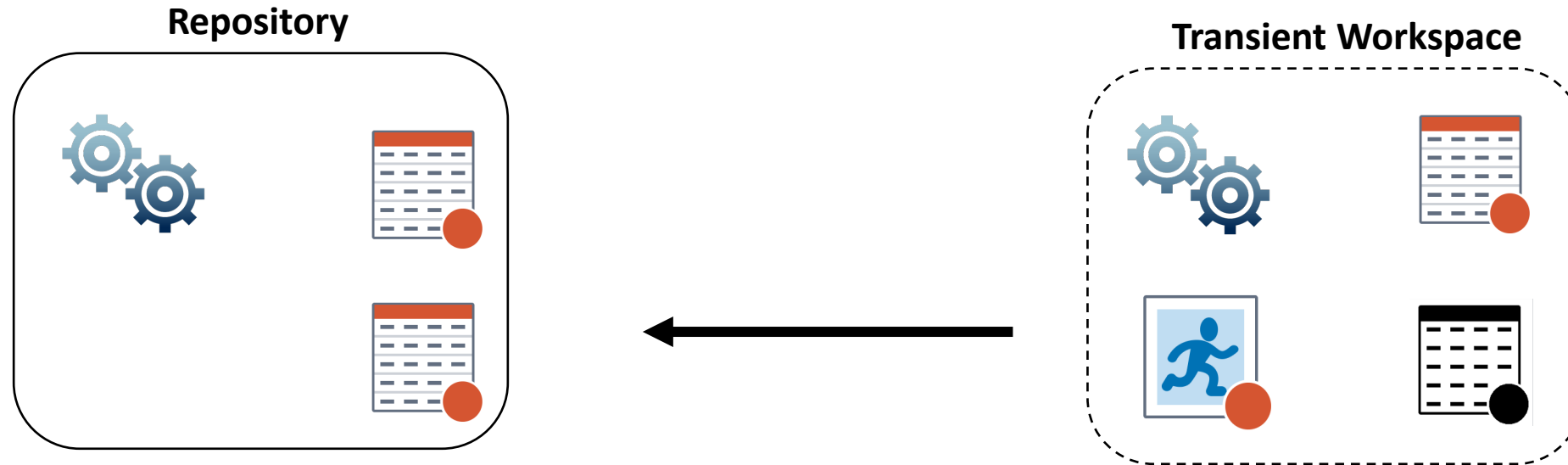
Repository



Transient Workspace

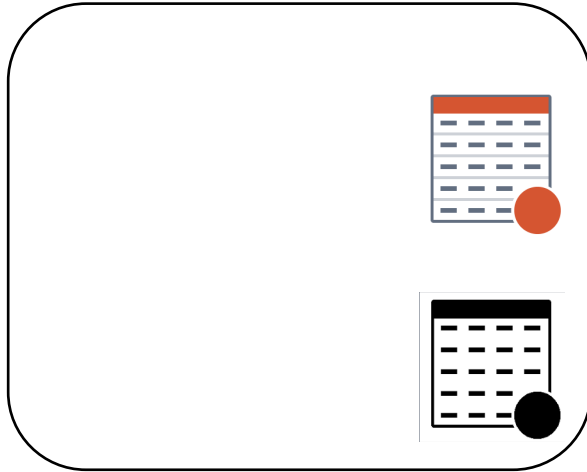


General Introduction



General Introduction

Repository



Automatic Dataflow Framework

Framework for automatic processing of clinical trial data

Automatic Dataflow Framework

Framework for automatic processing of clinical trial data

- One configurable parent job that executes a range of child jobs

Automatic Dataflow Framework

Framework for automatic processing of clinical trial data

- One configurable parent job that executes a range of child jobs
- Jobs perform tasks to migrate inbound datasets, process them, examine their consistency and to convert them into SDTM datasets

Automatic Dataflow Framework

Framework for automatic processing of clinical trial data

- One configurable parent job that executes a range of child jobs
- Jobs perform tasks to migrate inbound datasets, process them, examine their consistency and to convert them into SDTM datasets
- The final child job is dedicated to monitor the jobs and to report their status to the user

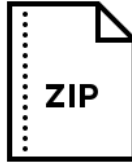


- Parent job sets up one transient workspace
- It will execute child jobs one-by-one



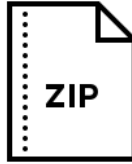
- The child job create a transient workspace to process study data

Collect study data



- The child job create a transient workspace to process study data

Collect study data

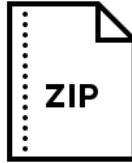


Job Report



- The child job create a transient workspace to process study data
- Each child job creates a report dataset

Collect study data

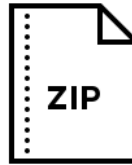


Job Report



- The child job finishes and moves the output to the repository

Collect study data



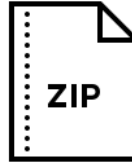
Job Report



Dataflow Report

- The child job finishes and moves the output to the repository
- The parent job creates a report which stays in its transient workspace

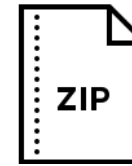
Collect study data



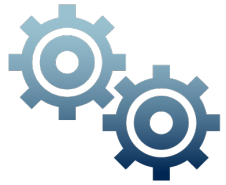
Job Report



Convert data



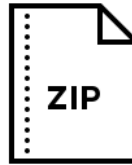
Job Report



Dataflow Report

- The following job is executed and accesses the output of the previous job from the repository in its transient workspace

Collect study data



Job Report



Convert data



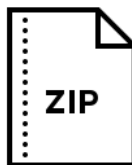
Job Report



Dataflow Report

- The following job is executed and accesses the output of the previous job from the repository in its transient workspace
- The reports are updated
- Everything is moved to the repository

Collect study data



Job Report



Convert data



Job Report



Convert data

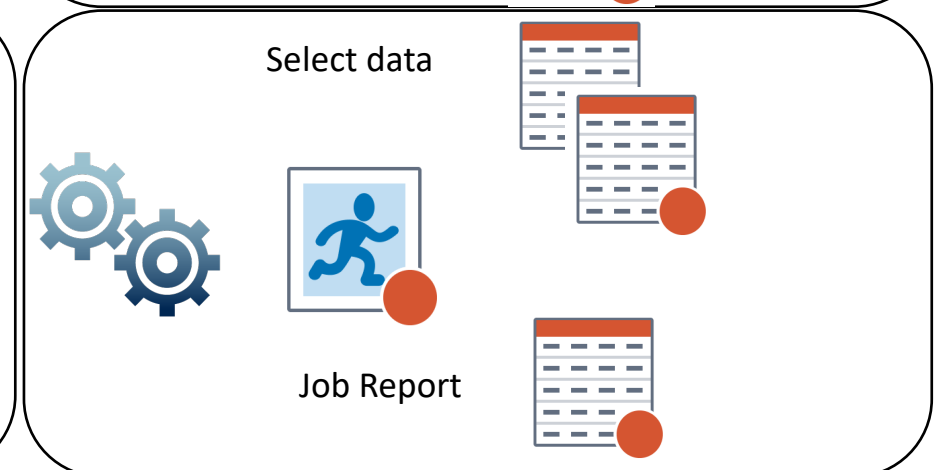
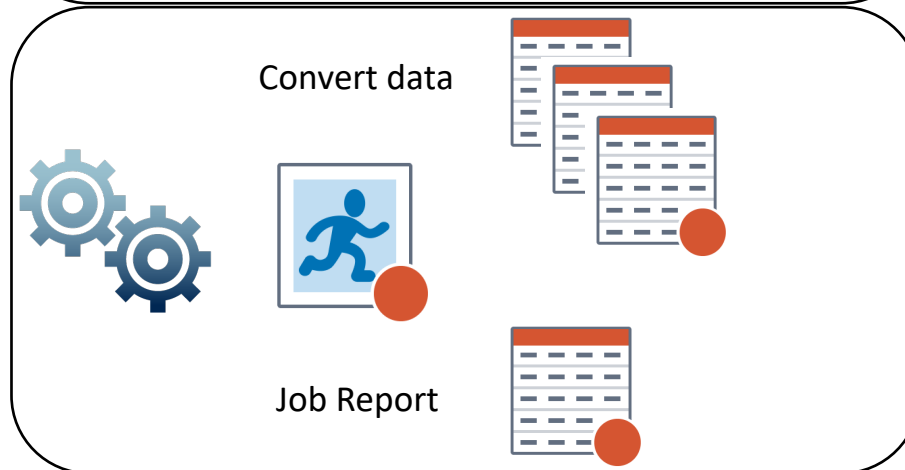
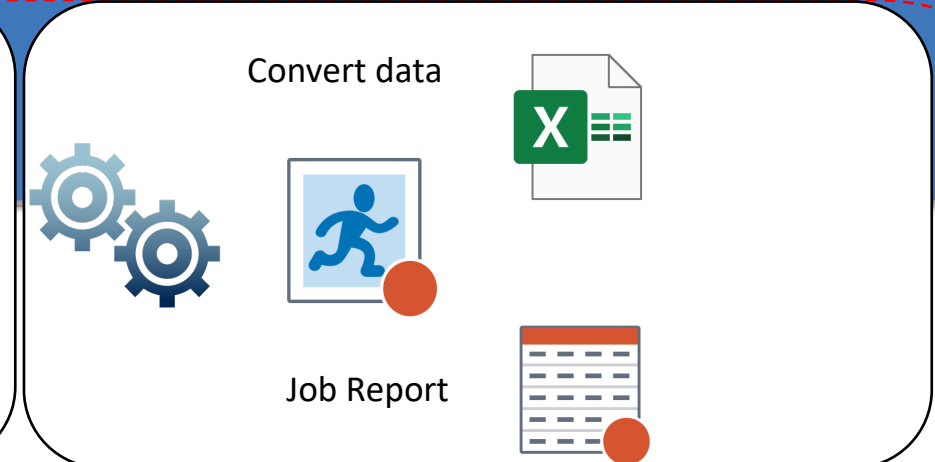
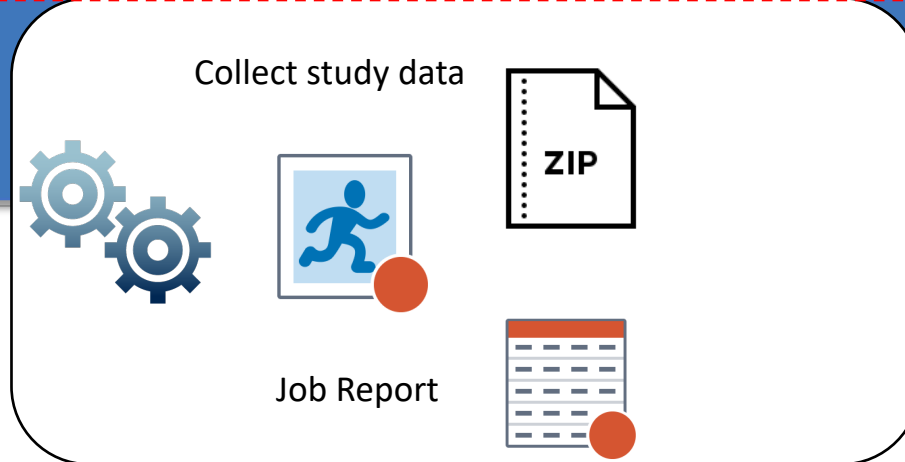


Job Report



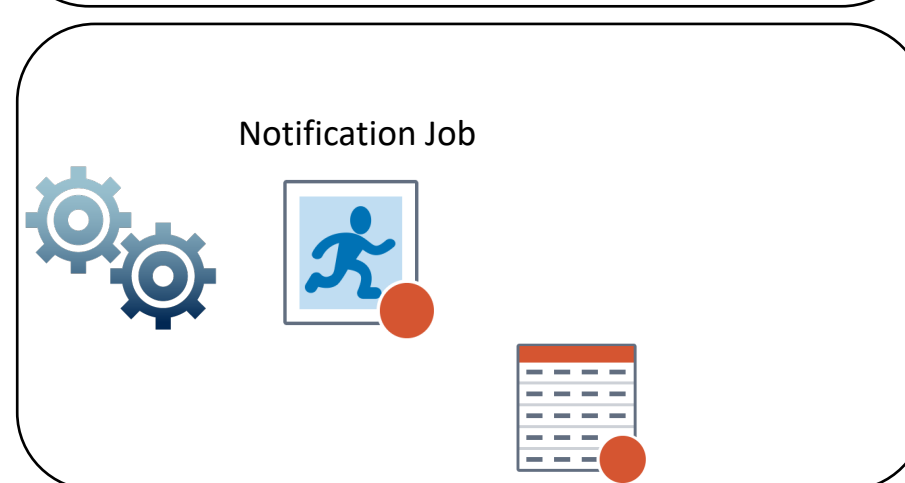
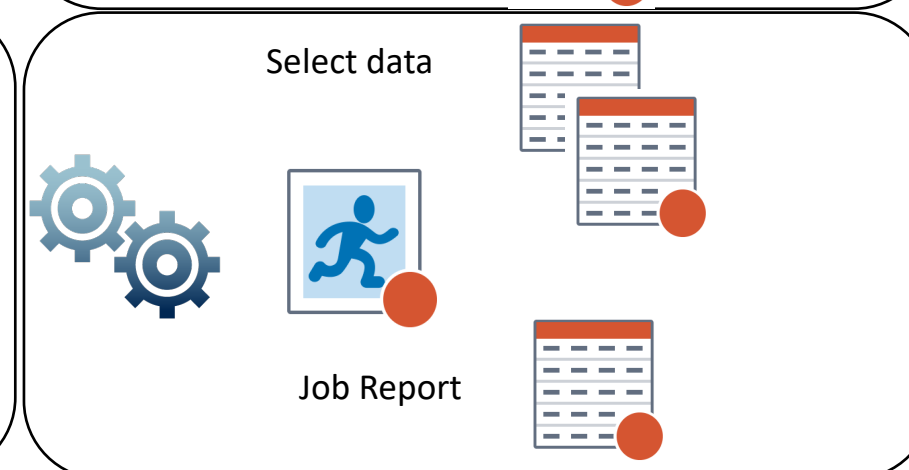
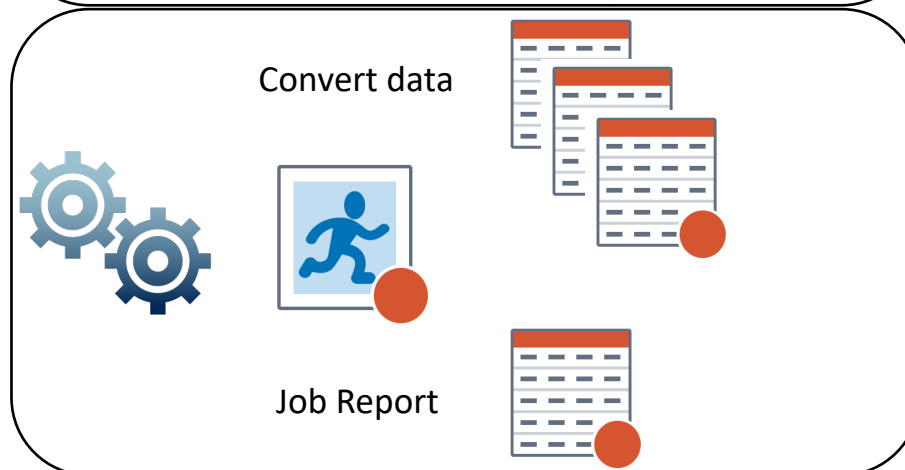
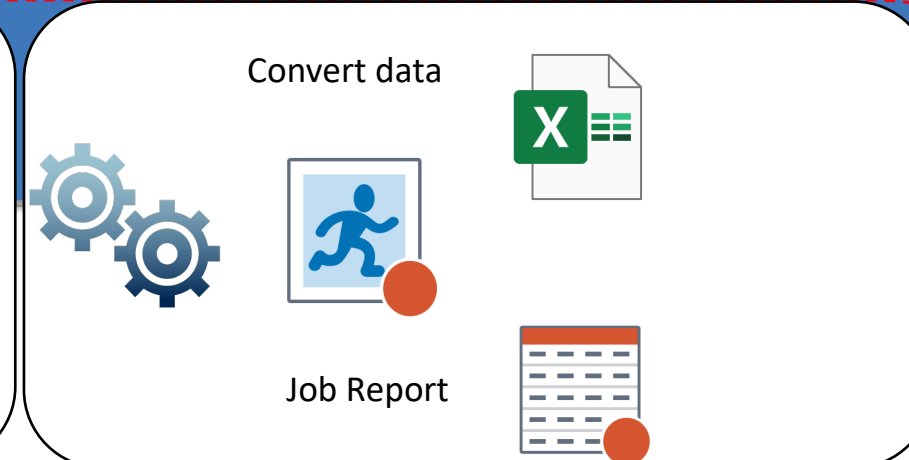
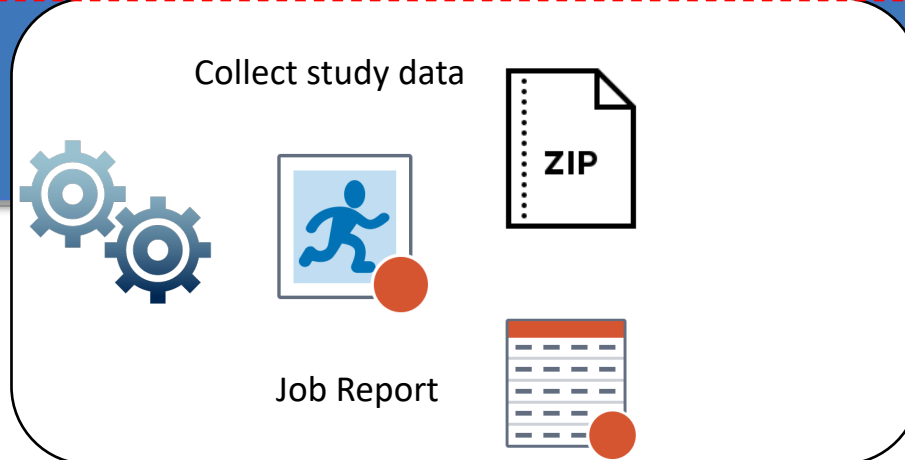
Dataflow Report

- The cycle continues
- Reports monitor the process



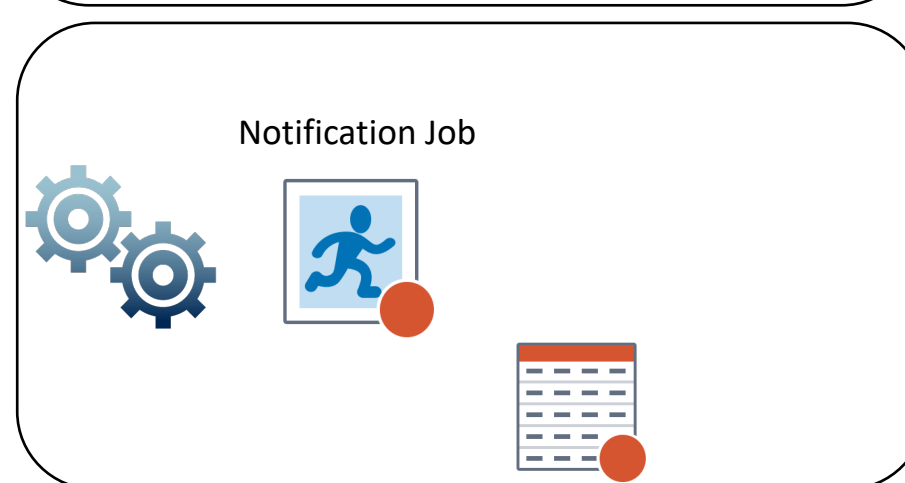
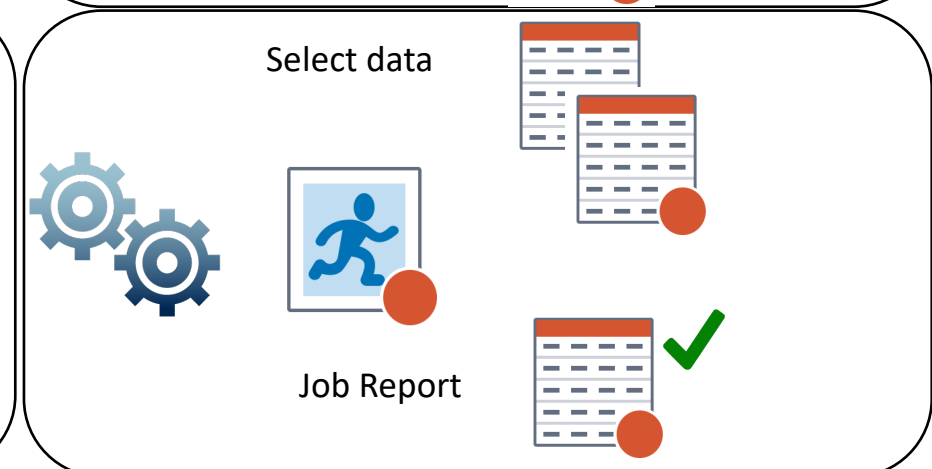
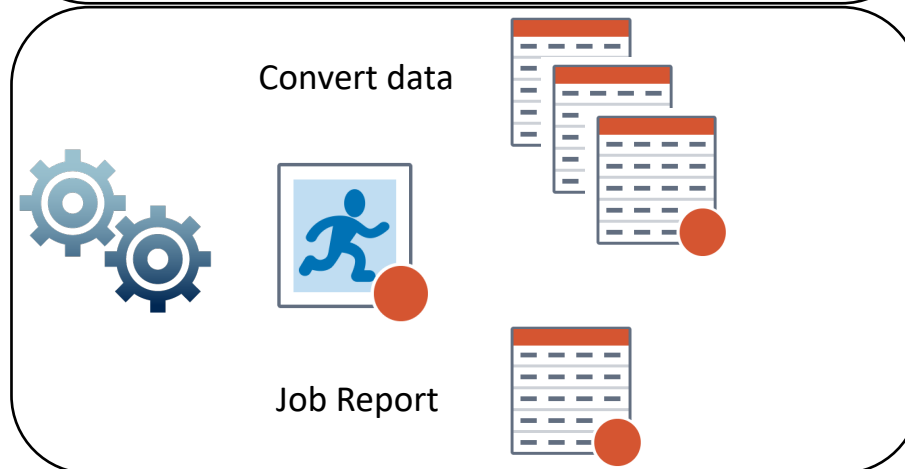
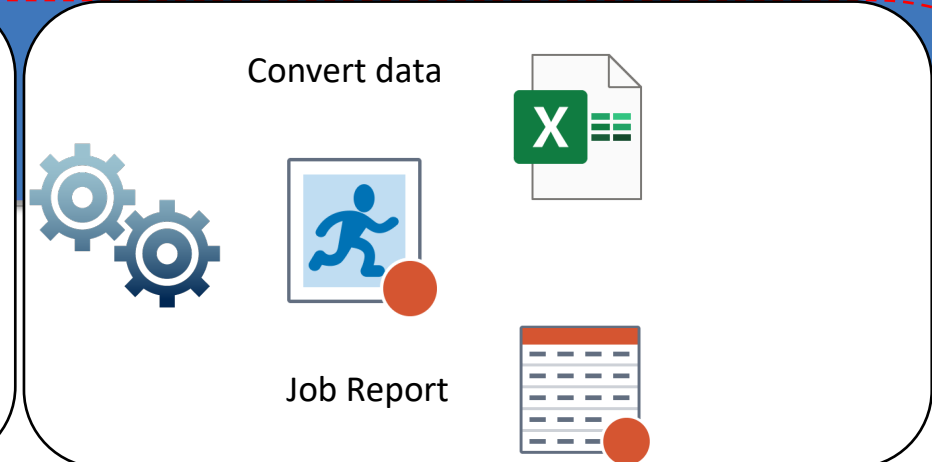
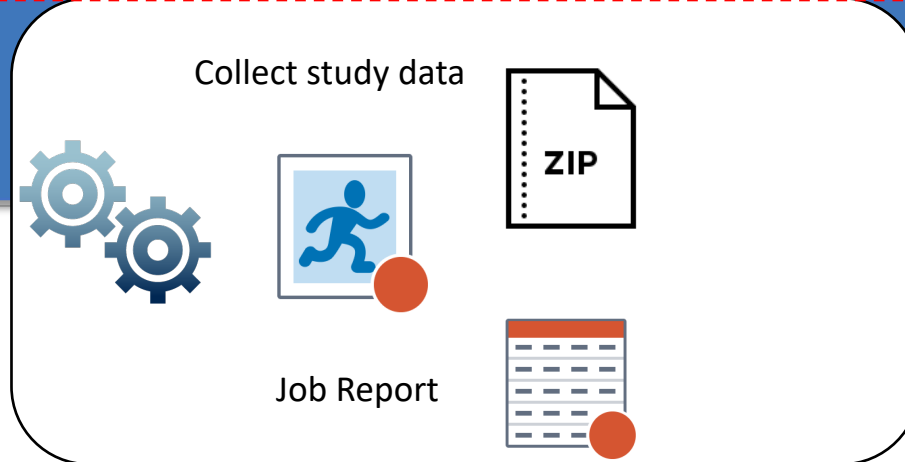
Dataflow Report

- The cycle continues
- Reports monitor the process



Dataflow Report

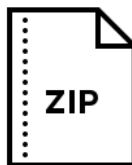
- The final child job will send the summarized report to the user
- The child job report is accessible in the repository but the parent job report is still in its own workspace



Dataflow Report

- The final child job will send the summarized report to the user
- The child job report is accessible in the repository but the parent job report is still in its own workspace

Collect study data



Job Report



Convert data



Job Report



Convert data



Job Report



Select data



Job Report



Dataflow Report

?

Notification Job



Goal:

- Combine the dataflow and job report in one dataset to create one email

Problem:

- The job report and the dataflow report are in two separate locations

Approaches

Option 1)

Generate the final report **outside** the workflow

- The last job could run independently after the workflow is finished
- It is possible to have two separate jobs that are scheduled to run after each other with **no changes**

Approaches

Option 1)

Generate the final report **outside** the workflow

- The last job could run independently after the workflow is finished
- It is possible to have two separate jobs that are scheduled to run after each other with **no changes**
- **Downsides:**
 - The jobs need to be timed right
 - It defeats the idea of having one singular dataflow
 - This may be more inconvenient for future users (e.g. addition of new jobs changes the timing; new user need to know about the relationship between jobs)

Approaches

Option 1)

Generate the final report **outside** the workflow

- The last job could run independently after the workflow is finished
- It is possible to have two separate jobs that are scheduled to run after each other with **no changes**
- **Downsides:**
 - The jobs need to be timed right
 - It defeats the idea of having one singular dataflow
 - This may be more inconvenient for future users (e.g. addition of new jobs changes the timing; new user need to know about the relationship between jobs)

Option 2)

Generate the final report **as part** of the workflow

- Allow the notification job to access the dataflow report during the **same dataflow**
- Grant the notification job access the parent jobs' transient workspace

Paths to the transient workspace

LSAF automatically generates various macro variables that point to file and / or workspace location:

```
/* SAS Program location */  
&_SASFILEPATH_;
```

```
/SAS/0000/Files/LSAF_Test/Example_Job/Programs/child_job.sas  
/* User transient workspace */  
&_SASUSRWS_;
```

```
/lsafshared/SASWorkspaces/.t.transient/child_job.job-840cc342-4fda-4dc7-8410-b03904f44582/Users/KW1  
/* Job transient workspace */  
&_SASWS_;
```

```
/lsafshared/SASWorkspaces/.t.transient/child_job.job-840cc342-4fda-4dc7-8410-b03904f44582
```

Paths to the transient workspace

LSAF automatically generates various macro variables that point to file and / or workspace location:

```
/* SAS Program location */
&_SASFILEPATH_;

/SAS/0000/Files/LSAF_Test/Example_Job/Programs/child_job.sas
/* User transient workspace */
&_SASUSRWS_;

/lsafshared/SASWorkspaces/.t.transient/child_job.job-840cc342-4fda-4dc7-8410-b03904f44582/Users/KW1
/* Job transient workspace */
&_SASWS_;

/lsafshared/SASWorkspaces/.t.transient/child_job.job-840cc342-4fda-4dc7-8410-b03904f44582
```

The location is always relative to the running (child) job

Paths to the transient workspace

LSAF automatically generates various macro variables that point to file and / or workspace location:

```
/* SAS Program location */
&_SASFILEPATH_;

/SAS/0000/Files/LSAF_Test/Example_Job/Programs/child_job.sas
/* User transient workspace */
&_SASUSRWS_;

/lsafshared/SASWorkspaces/.t.transient/child_job.job-840cc342-4fda-4dc7-8410-b03904f44582/Users/KW1
/* Job transient workspace */
&_SASWS_;

/lsafshared/SASWorkspaces/.t.transient/child_job.job-840cc342-4fda-4dc7-8410-b03904f44582
```

The location is always relative to the running (child) job

→ How can we use this to access the parent workspace location?

Paths to the transient workspace

LSAF automatically generates various macro variables that point to file and / or workspace location:

```
/* SAS Program location */
&_SASFILEPATH_;

/SAS/0000/Files/LSAF_Test/Example_Job/Programs/child_job.sas
/* User transient workspace */
&_SASUSRWS_;

/lsafshared/SASWorkspaces/.t.transient/child_job.job-840cc342-4fda-4dc7-8410-b03904f44582/Users/KW1
/* Job transient workspace */
&_SASWS_;

/lsafshared/SASWorkspaces/.t.transient/child_job.job-840cc342-4fda-4dc7-8410-b03904f44582
```

The location is always relative to the running (child) job

→ How can we use this to access the parent workspace location?

→ Inherit them as job parameters in the child job

Job Parameters

Child Job:

⊖ Jobs

⚙ child_job.job

×

✔ Job saved successfully.

Details				
SAS Programs				
Parameters				
Inputs				
Outputs				
	☐	Variable Name *	Label *	Type
	☐	output	Output Location	Folder ▼
	☐	_saspjws_	Parent job transient workspace	Character ▼
				Default Value
				./Output ✎

Job Parameters

Child Job:

Jobs

child_job.job x

Job saved successfully.

Details

SAS Programs

Parameters

Inputs

Outputs

☐	Variable Name *	Label *	Type	Default Value
☐	output	Output Location	Folder	./Output
☐	_saspjws_	Parent job transient workspace	Character	

- The new macro variable is created as a parameter in the **child** job

- The parameter should be an **empty** character macro variable

→ The value will be added by the parent job

Job Parameters

Child Job:

Jobs

child_job.job x

Job saved successfully.

Details
SAS Programs
Parameters
Inputs
Outputs

☐	Variable Name *	Label *	Type	Default Value
☐	output	Output Location	Folder	./Output
☐	_saspjws_	Parent job transient workspace	Character	

- The new macro variable is created as a parameter in the **child** job

- The parameter should be an **empty** character macro variable

→ The value will be added by the parent job

Parent Job:

Jobs

parent_job.job x

Job saved successfully.

Details
SAS Programs
Parameters
Inputs
Outputs

☐	Variable Name *	Label *	Type	Default Value
☐	output	Output Location	Folder	./Output
☐	jobpath	Location of Jobs	Folder	./

Output: Without Access to Parent Transient Workspace

```
⊖ %macro child_job();  
  
    /* Assign child output library */  
    libname childout "&output.";  
  
    /* Update the parent report */  
    data childout.report;  
        set childout.report;  
        output;  
        message = "Start of child job";  
        transient_workspace = "&_SASWS.";  
        output;  
    run;  
  
%mend child_job;
```

Output: Without Access to Parent Transient Workspace

⊖ %macro child_job();

/* Assign child output library */

libname childout "&output.";

/* Update the parent report */

data childout.report;

set childout.report;

output;

message = "Start of child job";

transient_workspace = "&_SASWS.";

output;

run;

%mend child_job;

NOTE: Libref OUTPUT was successfully assigned as follows:

Engine: V9

Physical Name:

/lsafshared/SASWorkspaces/.transient/parent_job.job-a7e60181-1a86-40dd-85ba-76c59d29fd59/SAS/0000/DRM_WAVE2_001/Files/10_UAT/L

SAF_TestCases/00_DATA_FLOW/Example Job/Output

ERROR: File CHILDOUT.REPORT.DATA does not exist.

NOTE: The SAS System stopped processing this step because of errors.

WARNING: The data set CHILDOUT.REPORT may be incomplete. When this step was stopped there were 0 observations and 2 variables.

NOTE: DATA statement used (Total process time):

real time 0.00 seconds

cpu time 0.00 seconds

Output: Without Access to Parent Transient Workspace

```
⊖ %macro child_job();  
  
  /* Assign child output library */  
  libname childout "&output.";  
  
  /* Update the parent report */  
  data childout.report;  
    set childout.report;  
    output;  
    message = "Start of child job";  
    transient_workspace = "&_SASWS.";  
    output;  
  run;  
  
%mend child_job;
```

NOTE: Libref OUTPUT was successfully assigned as follows:
Engine: V9
Physical Name:
/lsafshared/SASWorkspaces/.transient/parent_job.job-a7e60181-1a86-40dd-85ba-76c59d29fd59/SAS/0000/DRM_WAVE2_001/Files/10_UAT/L
SAF_TestCases/00_DATA_FLOW/Example Job/Output
ERROR: File CHILDOUT.REPORT.DATA does not exist.

NOTE: The SAS System stopped processing this step because of errors.
WARNING: The data set CHILDOUT.REPORT may be incomplete. When this step was stopped there were 0 observations and 2 variables.
NOTE: DATA statement used (Total process time):
real time 0.00 seconds
cpu time 0.00 seconds

Repository.REPORT.DATASET × +

Search

#	message	transient_workspace
1	Start of parent job	/lsafshared/SASWorkspaces/.transient/parent_job.job-a7e60181-1a86-40dd-8...

Parent Job SAS Code

```
%macro parent_job();

/* Assign report library */
libname output "&output.";

/* Create initial report */
data output.report;
  length message $ 50 transient_workspace $ 300;
  message = "Start of parent job";
  transient_workspace = "&_SASWS.";
  output;
run;

/* Define the path to the child job (repository) */
%let jobpath=&jobpath.;
%let jobpathrepository=%substr(&jobpath., %index(&jobpath., %str(/SAS/)));

%put [Parent Job] original parent transient workspace _SASWS_;;
%put [Parent Job] &_SASWS_;;

/* Access child job parameters */
%LSAF_getJobParameters(LSAF_path=&jobpathrepository./child_job.job
                      ,LSAF_version=
                      ,sas_dsname=originalJobParameters);

/* Populate the empty _SASWSPJW_ parameter */
data work.JobParameters;
  set work.originalJobParameters (rename=(defaultValue = value));
  if name="_saspjws_" then Value="&_SASWS.";
run;

/* Submit the modified job */
%LSAF_submitjob(LSAF_path=&jobpathrepository./child_job.job
               ,sas_dsname=work.JobParameters);

%put %sysfunc(sleep(10,1));

/* Obtain submission status */
%LSAF_GETSUBMISSIONSTATUS(LSAF_JOBSUBMISSION_ID=&_lsafJobSubmissionId_);
%mend parent_job;
```

Parent Job SAS Code

⊖ %macro parent_job();

```
/* Assign report library */  
libname output "&output.";
```

```
/* Create initial report */
```

```
data output.report;  
  length message $ 50 transient_workspace $ 300;  
  message = "Start of parent job";  
  transient_workspace = "&_SASWS.";  
  output;  
run;
```

```
/* Define the path to the child job (repository) */
```

```
%let jobpath=&jobpath.;  
%let jobpathrepository=%substr(&jobpath., %index(&jobpath., %str(/SAS/)));
```

```
%put [Parent Job] original parent transient workspace _SASWS_;  
%put [Parent Job] &_SASWS_;
```

```
/* Access child job parameters */
```

```
%LSAF_getJobParameters(LSAF_path=&jobpathrepository./child_job.job  
                      ,LSAF_version=  
                      ,sas_dsname=originalJobParameters);
```

```
/* Populate the empty _SASWSPJW_ parameter */
```

```
data work.JobParameters;  
  set work.originalJobParameters (rename=(defaultValue = value));  
  if name="_saspjws_" then Value="&_SASWS_";  
run;
```

```
/* Submit the modified job */
```

```
%LSAF_submitjob(LSAF_path=&jobpathrepository./child_job.job  
               ,sas_dsname=work.JobParameters);
```

```
%put %sysfunc(sleep(10,1));
```

```
/* Obtain submission status */
```

```
%LSAF_GETSUBMISSIONSTATUS(LSAF_JOBSubmission_ID=&_lsafJobSubmissionId_);
```

```
%mend parent_job;
```

Assign output library

Create initial report in parent transient workspace

Parent Job SAS Code

⊖ %macro parent_job();

```
/* Assign report library */  
libname output "&output.";
```

Assign output library

```
/* Create initial report */  
data output.report;  
  length message $ 50 transient_workspace $ 300;  
  message = "Start of parent job";  
  transient_workspace = "&_SASWS_";  
  output;  
run;
```

Create initial report in parent transient workspace

```
/* Define the path to the child job (repository) */  
%let jobpath=&jobpath.;  
%let jobpathrepository=%substr(&jobpath., %index(&jobpath., %str(/SAS/)));  
  
%put [Parent Job] original parent transient workspace _SASWS_;  
%put [Parent Job] &_SASWS_;
```

Obtain paths for transient workspace and repository

Print paths to the log

```
/* Access child job parameters */  
%LSAF_getJobParameters(LSAF_path=&jobpathrepository./child_job.job  
  ,LSAF_version=  
  ,sas_dsname=originalJobParameters);
```

```
/* Populate the empty _SASWSPJW_ parameter */  
data work.JobParameters;  
  set work.originalJobParameters (rename=(defaultValue = value));  
  if name="_saspjws_" then Value="&_SASWS_";  
run;
```

```
/* Submit the modified job */  
%LSAF_submitjob(LSAF_path=&jobpathrepository./child_job.job  
  ,sas_dsname=work.JobParameters);
```

```
%put %sysfunc(sleep(10,1));
```

```
/* Obtain submission status */  
%LSAF_GETSUBMISSIONSTATUS(LSAF_JOBSUBMISSION_ID=&_lsafJobSubmissionId_);  
%mend parent_job;
```

Parent Job SAS Code

```
%macro parent_job();
```

```
/* Assign report library */  
libname output "&output.";
```

```
/* Create initial report */  
data output.report;  
  length message $ 50 transient_workspace $ 300;  
  message = "Start of parent job";  
  transient_workspace = "&_SASWS_";  
  output;  
run;
```

```
/* Define the path to the child job (repository) */  
%let jobpath=&jobpath.;  
%let jobpathrepository=%substr(&jobpath., %index(&jobpath., %str(/SAS/)));
```

```
%put [Parent Job] original parent transient workspace _SASWS_;  
%put [Parent Job] &_SASWS_;
```

```
/* Access child job parameters */  
%LSAF_getJobParameters(LSAF_path=&jobpathrepository./child_job.job  
                      ,LSAF_version=  
                      ,sas_dsname=originalJobParameters);
```

```
/* Populate the empty _SASWSPJW_ parameter */  
data work.JobParameters;  
  set work.originalJobParameters (rename=(defaultValue = value));  
  if name="_saspjws_" then Value="&_SASWS_";  
run;
```

```
/* Submit the modified job */  
%LSAF_submitjob(LSAF_path=&jobpathrepository./child_job.job  
               ,sas_dsname=work.JobParameters);
```

```
%put %sysfunc(sleep(10,1));
```

```
/* Obtain submission status */  
%LSAF_GETSUBMISSIONSTATUS(LSAF_JOBSUBMISSION_ID=&_lsafJobSubmissionId_);  
%mend parent_job;
```

Assign output library

Create initial report in parent transient workspace

Obtain paths for transient workspace and repository

Print paths to the log

Use LSAF to obtain child job parameters

Modify the _SASPJWS_ parameter by adding the value for _SASWS_

Parent Job SAS Code

```
%macro parent_job();
```

```
/* Assign report library */  
libname output "&output.";
```

```
/* Create initial report */  
data output.report;  
  length message $ 50 transient_workspace $ 300;  
  message = "Start of parent job";  
  transient_workspace = "&_SASWS.";  
  output;  
run;
```

```
/* Define the path to the child job (repository) */  
%let jobpath=&jobpath.;  
%let jobpathrepository=%substr(&jobpath., %index(&jobpath., %str(/SAS/)));
```

```
%put [Parent Job] original parent transient workspace _SASWS_;  
%put [Parent Job] &_SASWS_;
```

```
/* Access child job parameters */  
%LSAF_getJobParameters(LSAF_path=&jobpathrepository./child_job.job  
                      ,LSAF_version=  
                      ,sas_dsname=originalJobParameters);
```

```
/* Populate the empty _SASWSPJW_ parameter */  
data work.JobParameters;  
  set work.originalJobParameters (rename=(defaultValue = value));  
  if name="_saspjws_" then Value="&_SASWS.";  
run;
```

```
/* Submit the modified job */  
%LSAF_submitjob(LSAF_path=&jobpathrepository./child_job.job  
               ,sas_dsname=work.JobParameters);
```

```
%put %sysfunc(sleep(10,1));
```

```
/* Obtain submission status */  
%LSAF_GETSUBMISSIONSTATUS(LSAF_JOBSubmission_ID=&_lsafJobSubmissionId_);
```

```
%mend parent_job;
```

Assign output library

Create initial report in parent transient workspace

Obtain paths for transient workspace and repository

Print paths to the log

Use LSAF to obtain child job parameters

Modify the _SASPJWS_ parameter by adding the value for _SASWS_

Submit the modified job and monitor its status

Child Job SAS Code

```
⊖ %macro child_job();  
  
  %put [Child Job] Parent workspace;;  
  %put [Child Job] &_SASPJWS_.;  
  %put [Child Job] Child transient workspace;;  
  %put [Child Job] &_SASWS_.;  
  
  /* Assign child output library */  
  libname childout "&output.";   
  
  /* Assign parent output library */  
  %let pj_output = %substr(&output.,%index(&output.,/SAS/));  
  %let pj_output = &_SASPJWS_.&pj_output.;  
  
  libname output "&pj_output.";   
  
  /* Update the parent report */  
  data output.report;  
    set output.report;  
    output;  
    message = "Start of child job";  
    transient_workspace = "&_SASWS_.";  
    output;  
  run;  
  
%mend child_job;
```

Child Job SAS Code

⊖ %macro child_job();

```
%put [Child Job] Parent workspace;;  
%put [Child Job] &_SASPJWS_.;  
%put [Child Job] Child transient workspace;;  
%put [Child Job] &_SASWS_.;
```

Print path to child and parent transient workspace to the log

```
/* Assign child output library */  
libname childout "&output.";   
  
/* Assign parent output library */  
%let pj_output = %substr(&output.,%index(&output.,/SAS/));  
%let pj_output = &_SASPJWS_.&pj_output.;  
  
libname output "&pj_output.";   
  
/* Update the parent report */  
data output.report;  
  set output.report;  
  output;  
  message = "Start of child job";  
  transient_workspace = "&_SASWS_.";  
  output;  
run;  
  
%mend child_job;
```

Child Job SAS Code

⊖ %macro child_job();

```
%put [Child Job] Parent workspace;;  
%put [Child Job] &_SASPJWS_.;  
%put [Child Job] Child transient workspace;;  
%put [Child Job] &_SASWS_.;
```

Print path to child and parent transient workspace to the log

```
/* Assign child output library */  
libname childout "&output.";
```

Assign output library for the child transient workspace

```
/* Assign parent output library */  
%let pj_output = %substr(&output.,%index(&output.,/SAS/));  
%let pj_output = &_SASPJWS_.&pj_output.;
```

```
libname output "&pj_output.";
```

```
/* Update the parent report */  
data output.report;  
  set output.report;  
  output;  
  message = "Start of child job";  
  transient_workspace = "&_SASWS_.";  
  output;  
run;
```

```
%mend child_job;
```

Child Job SAS Code

⊖ %macro child_job();

```
%put [Child Job] Parent workspace;;  
%put [Child Job] &_SASPJWS_.;  
%put [Child Job] Child transient workspace;;  
%put [Child Job] &_SASWS_.;
```

```
/* Assign child output library */  
libname childout "&output.";
```

```
/* Assign parent output library */  
%let pj_output = %substr(&output.,%index(&output.,/SAS/));  
%let pj_output = &_SASPJWS_.&pj_output.;  
  
libname output "&pj_output.";
```

```
/* Update the parent report */  
data output.report;  
  set output.report;  
  output;  
  message = "Start of child job";  
  transient_workspace = "&_SASWS_.";  
  output;  
run;
```

```
%mend child_job;
```

Print path to child and parent transient workspace to the log

Assign output library for the child transient workspace

Overwrite the child transient workspace location with the parent one

Use the new path to assign an output library

Child Job SAS Code

⊖ %macro child_job();

```
%put [Child Job] Parent workspace;;  
%put [Child Job] &_SASPJWS_.;  
%put [Child Job] Child transient workspace;;  
%put [Child Job] &_SASWS_.;
```

```
/* Assign child output library */  
libname childout "&output.";
```

```
/* Assign parent output library */  
%let pj_output = %substr(&output.,%index(&output.,/SAS/));  
%let pj_output = &_SASPJWS_.&pj_output.;
```

```
libname output "&pj_output.";
```

```
/* Update the parent report */  
data output.report;  
  set output.report;  
  output;  
  message = "Start of child job";  
  transient_workspace = "&_SASWS_.";  
  output;  
run;
```

```
%mend child_job;
```

Print path to child and parent transient workspace to the log

Assign output library for the child transient workspace

Overwrite the child transient workspace location with the parent one

Use the new path to assign an output library

Modify the report in the parent transient workspace library

Output: Log Files

parent_job.log (Read-Only) x +

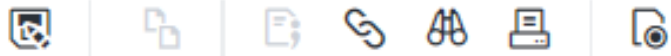


Code

```
127      69
128      70      %parent_job();
129      [Parent Job] original parent transient workspace _SASWS_:
130      [Parent Job] /lsafshared/SASWorkspaces/.transient/parent_job.job-7dd0d3ac-7853-4cb0-a231-51cb6eb0829d
```

Output: Log Files

parent_job.log (Read-Only) x +



Code

```
127      69
128      70      %parent_job();
129      [Parent Job] original parent transient workspace _SASWS_:
130      [Parent Job] /lsafshared/SASWorkspaces/.transient/parent_job.job-7dd0d3ac-7853-4cb0-a231-51cb6eb0829d
```

child_job.log (Read-Only) x +



Code

```
104      48
105      49      %child_job();
106      [Child Job] Parent workspace:
107      [Child Job] /lsafshared/SASWorkspaces/.transient/parent_job.job-7dd0d3ac-7853-4cb0-a231-51cb6eb0829d
108      [Child Job] Child transient workspace:
109      [Child Job] /lsafshared/SASWorkspaces/.transient/child_job.job-906022a0-be6b-492b-b4fd-b7ae6168464a
```

Output: Report Dataset

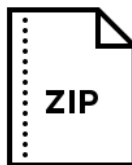
 Repository.REPORT.DATASET × +

 Search

#	 message	 transient_workspace
1	Start of parent job	/lsafshared/SASWorkspaces/.transient/parent_job.job-227b97d5-39ad-4880-a193-9484770c3...
2	Start of child job	/lsafshared/SASWorkspaces/.transient/child_job.job-04b64bae-5e2c-4ed0-984f-2ddb3485a078

Collect study data



Job Report



Convert data



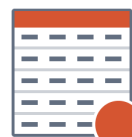
Job Report



Convert data



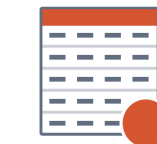
Job Report



Select data



Job Report



Notification Job



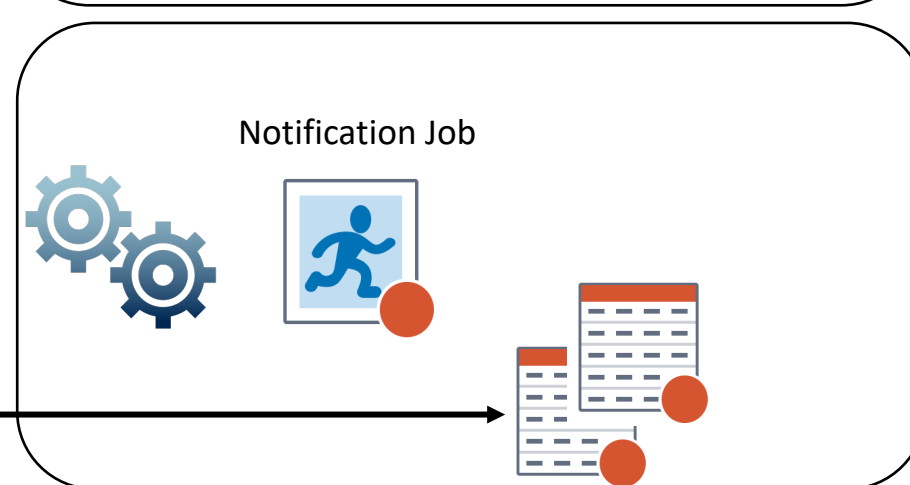
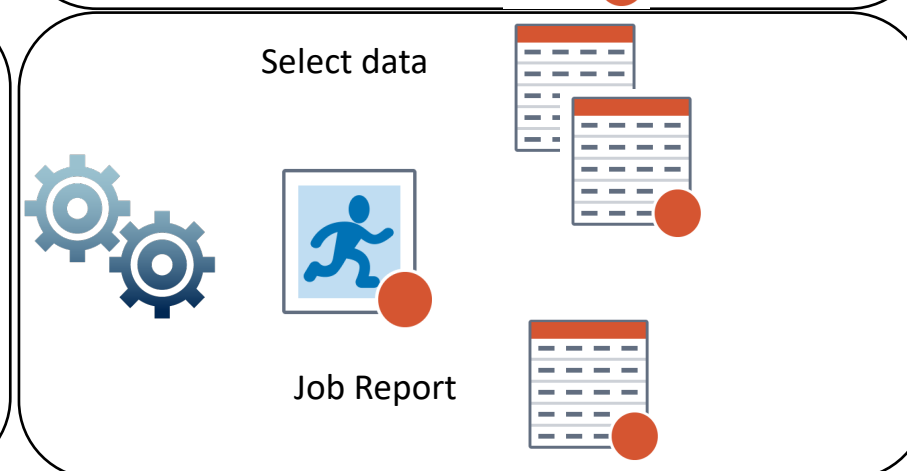
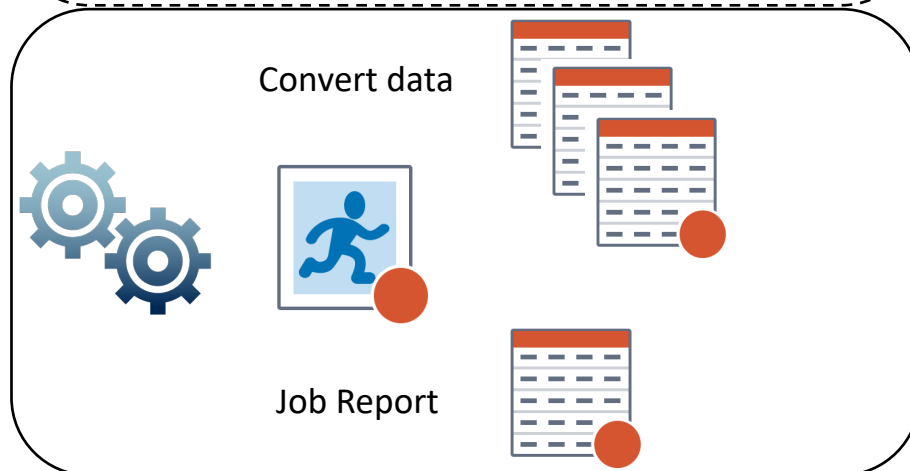
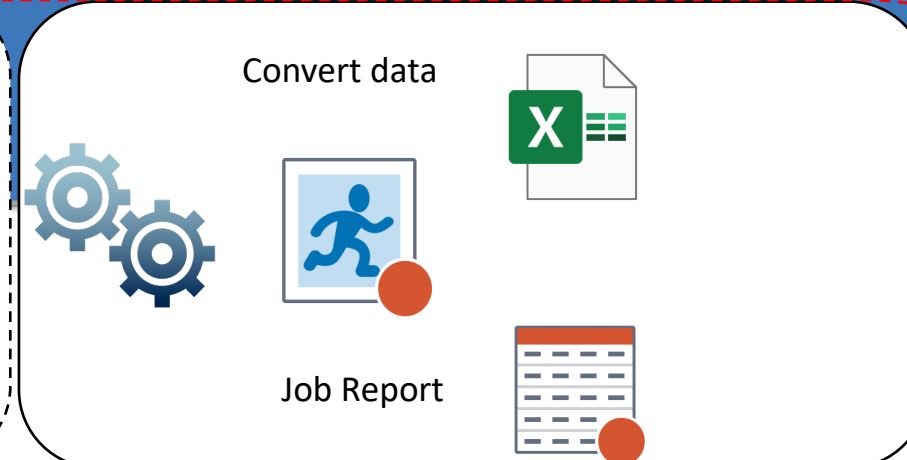
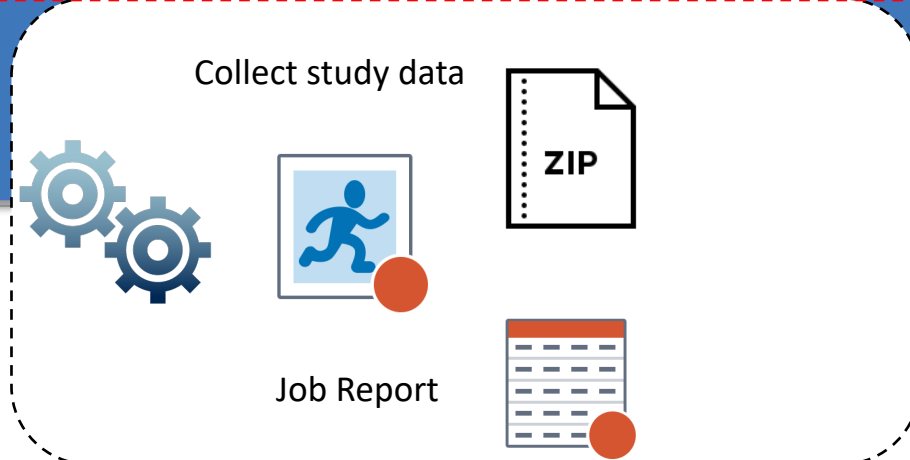
Dataflow Report



?

Solution:

- Create a job parameter that points to the parent job transient workspace

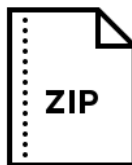


Dataflow Report

Solution:

- Create a job parameter that points to the parent job transient workspace
- Both reports can be combined in one datasets before the parent job ends

Collect study data



Job Report



Convert data



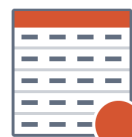
Job Report



Convert data



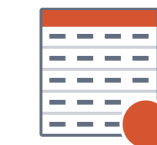
Job Report



Select data



Job Report



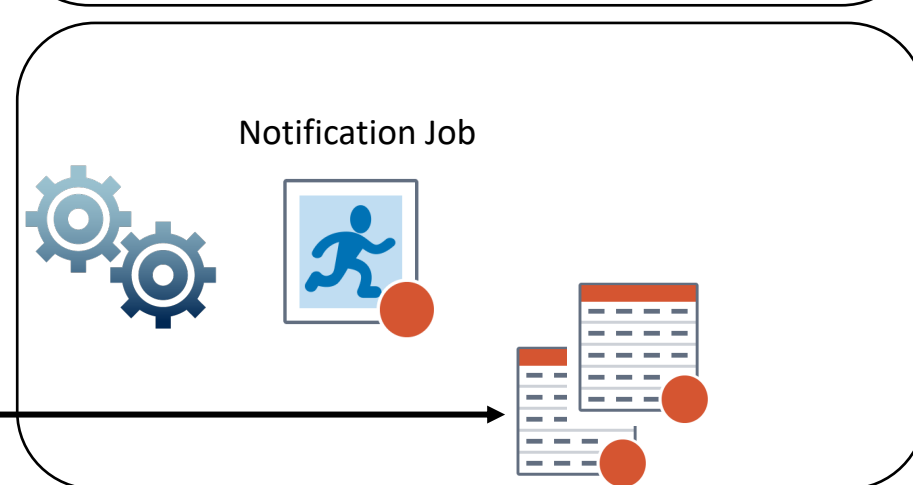
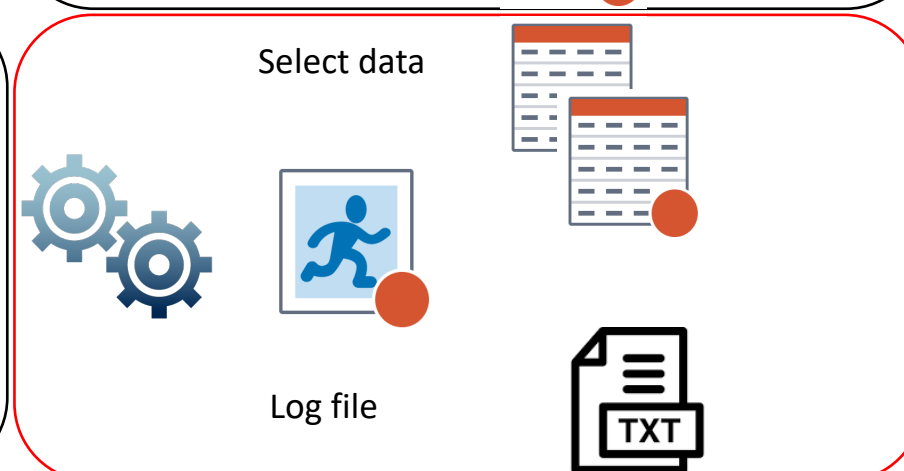
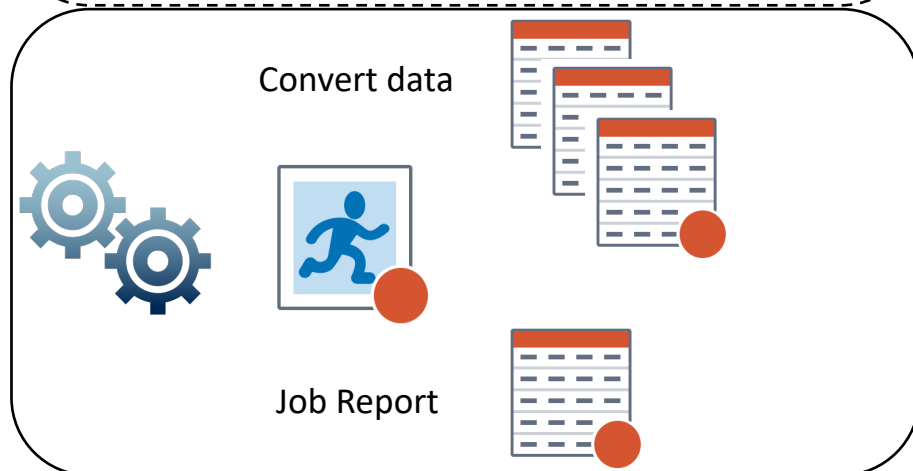
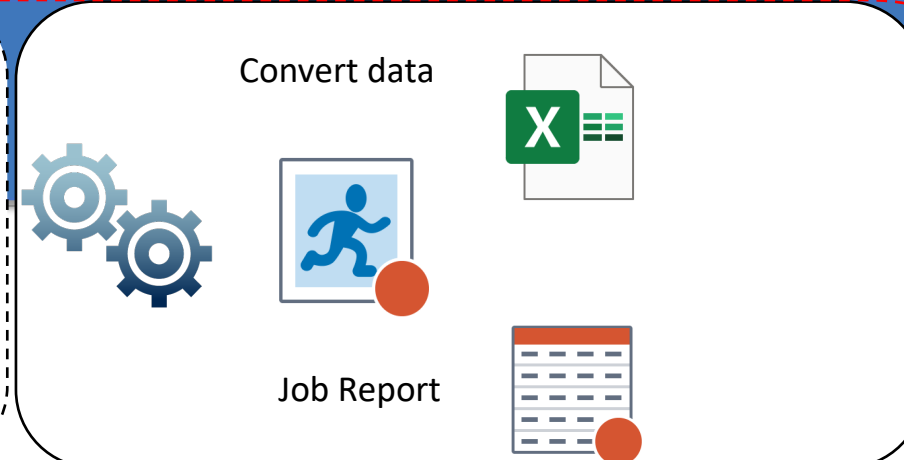
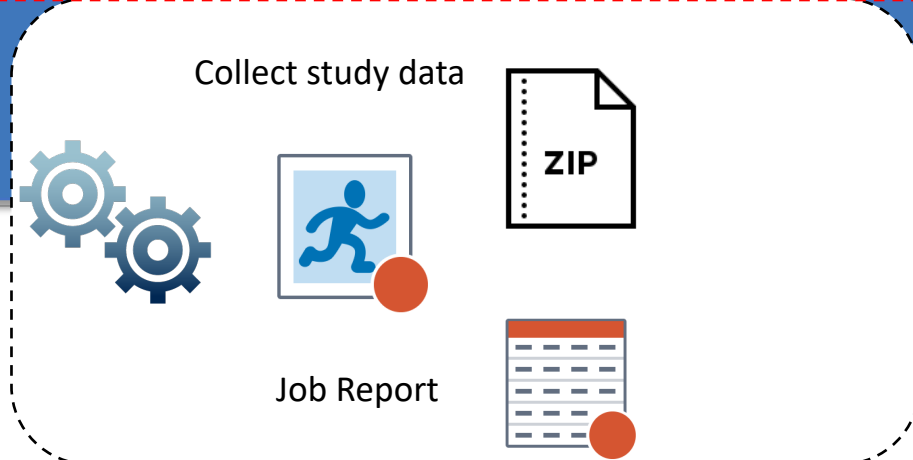
Dataflow Report

Notification Job

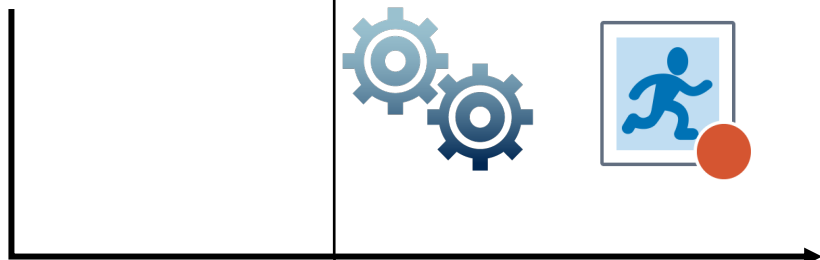


Other application:

- Monitor the dataflow and execute parts of a job conditionally



Dataflow Report



Other application:

- Monitor the dataflow and execute parts of a job conditionally
- E.g. in presence of an unexpected job

Other Application: Respond to other jobs' status

```
/* Only execute the program if the parent job ran */
%let continue = N;
data _null_;
  set output.report;
  if message = "Start of parent job" then call symput("continue","Y");
run;

%if &continue. = Y %then %do;
  %put Parent job execution detected.;
  %put Child job will continue.;

  /* Update the parent report */
  data output.report;
    set output.report;
    output;
    message = "Child job is starting after the parent job";
    transient_workspace = "&_SASWS_.";
    output;
  run;
%end; %else %do;
  %put Parent job execution not detected.;
  %put Child job will make no changes.;

  /* Update the parent report */
  data output.report;
    set output.report;
  run;
%end;
```

Other Application: Respond to other jobs' status

```
/* Only execute the program if the parent job ran */
%let continue = N;
data _null_;
  set output.report;
  if message = "Start of parent job" then call symput("continue","Y");
run;
```

The child job can access information from the parent job

```
%if &continue. = Y %then %do;
  %put Parent job execution detected.;
  %put Child job will continue.;

/* Update the parent report */
data output.report;
  set output.report;
  output;
  message = "Child job is starting after the parent job";
  transient_workspace = "&_SASWS_.";
  output;
run;

%end; %else %do;
  %put Parent job execution not detected.;
  %put Child job will make no changes.;

/* Update the parent report */
data output.report;
  set output.report;
run;
%end;
```

The code of the child job is conditional to the parent jobs' status

Other Application: Respond to other jobs' status

```
/* Only execute the program if the parent job ran */
%let continue = N;
data _null_;
  set output.report;
  if message = "Start of parent job" then call symput("continue","Y");
run;

%if &continue. = Y %then %do;
  %put Parent job execution detected.;
  %put Child job will continue.;

  /* Update the parent report */
  data output.report;
    set output.report;
    output;
    message = "Child job is starting after the parent job";
    transient_workspace = "&_SASWS.";
    output;
  run;
%end; %else %do;
  %put Parent job execution not detected.;
  %put Child job will make no changes.;

  /* Update the parent report */
  data output.report;
    set output.report;
  run;
%end;
```

Log File:

Parent job execution detected.
Child job will continue.

NOTE: There were 1 observations read from the data set OUTPUT.REPORT.

NOTE: The data set OUTPUT.REPORT has 2 observations and 2 variables.

NOTE: DATA statement used (Total process time):

real time	0.00 seconds
cpu time	0.01 seconds

Other Application: Respond to other jobs' status

```
/* Only execute the program if the parent job ran */
%let continue = N;
data _null_;
  set output.report;
  if message = "Start of parent job" then call symput("continue","Y");
run;

%if &continue. = Y %then %do;
  %put Parent job execution detected.;
  %put Child job will continue.;

  /* Update the parent report */
  data output.report;
    set output.report;
    output;
    message = "Child job is starting after the parent job";
    transient_workspace = "&_SASWS_.";
    output;
  run;
%end; %else %do;
  %put Parent job execution not detected.;
  %put Child job will make no changes.;

  /* Update the parent report */
  data output.report;
    set output.report;
  run;
%end;
```

Log File:

Parent job execution detected.
Child job will continue.

NOTE: There were 1 observations read from the data set OUTPUT.REPORT.
NOTE: The data set OUTPUT.REPORT has 2 observations and 2 variables.
NOTE: DATA statement used (Total process time):
real time 0.00 seconds
cpu time 0.01 seconds

Report Dataset:



Search

#	message
1	Start of parent job
2	Child job is starting after the parent job

Considerations

- The default separation of different transient workspaces also presents a **safeguard** mechanism:

Considerations

- The default separation of different transient workspaces also presents a **safeguard** mechanism:
 - SAS ERRORS will prevent the migration of data into the repository
 - No faulty data will be **introduced**
 - Existing data will not be **damaged**
 - Data is overwritten when expected (at the end of a job)

Considerations

- The default separation of different transient workspaces also presents a **safeguard** mechanism:
 - SAS ERRORS will prevent the migration of data into the repository
 - No faulty data will be **introduced**
 - Existing data will not be **damaged**
 - Data is overwritten when expected (at the end of a job)
- This **workaround** should be used for report datasets etc. and **not for actual data**

Considerations

- The default separation of different transient workspaces also presents a **safeguard** mechanism:
 - SAS ERRORS will prevent the migration of data into the repository
 - No faulty data will be **introduced**
 - Existing data will not be **damaged**
 - Data is overwritten when expected (at the end of a job)
- This **workaround** should be used for report datasets etc. and **not for actual data**
- **Reports** should be located in a **separate folder** in the **LSAF repository**

Considerations

- The default separation of different transient workspaces also presents a **safeguard** mechanism:
 - SAS ERRORS will prevent the migration of data into the repository
 - No faulty data will be **introduced**
 - Existing data will not be **damaged**
 - Data is overwritten when expected (at the end of a job)
 - This **workaround** should be used for report datasets etc. and **not for actual data**
 - **Reports** should be located in a **separate folder** in the **LSAF repository**
- Improve the interaction of jobs without risking data integrity

Conclusion

Accessing different job transient workspaces can be an **creative** solution for different issues.

Conclusion

Accessing different job transient workspaces can be an **creative** solution for different issues.

The interaction between transient workspace can increase the flexibility during the interactions of different jobs and can further improve their potential for **automation**.

Questions?

Contact:

Kai Wanke

OCS Life Sciences

sasquestions@ocs-consulting.com

<https://ocs-lifesciences.com/>

