HAZELTREE

Sync Manager ETL Template

Confidential

ETL Template

ETL Template is a custom algorithm that operates with data of Feed files in three steps

Extract
Transform
Load

HAZELTREE SYNCMANAGER

Dashboard
- All Applications
- All Connections
- All Workflow Templates
- All ETL Templates

Monitoring
- Workflow Instances
- Workflow Recovery
- Etl Results
- Etl Process Results
- Etl - Files Queue
- Scripts Instances
- Processing - Executions
- Processing - Diagnostics
- Scheduler Log

Etl Templates

Template Id

Application Select...

Create template | Delete | Clone

Name	Created By	Created On	Modified By
CS ETL Template	HTFSSupport	2018/06/05 16:57:13	HTFSSupport
Super Template	HTFSSupport	2018/06/05 13:56:26	HTFSSupport
TestEtl	HTFSSupport	2018/05/28 11:17:29	HTFSSupport

ETL Template

Etl Templates

Template Id

Application htfs

Create template | Delete | Clone

Name	Created By	Created On	Modified By	Modified On
Super Template	HTFSSupport	2018/06/05 13:56:26	HTFSSupport	2018/06/05 14:46:55

TemplateId 803ea6ed-9c1b-4e02-afe7-c0d9215dcd9b

Create template

Template Name * CS ETL Template

Application * htfs

Ok | Cancel

CS ETL Template

Data Source | Transformations | Test Run

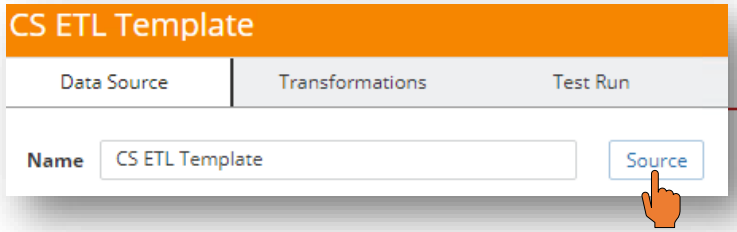
Name CS ETL Template | Source

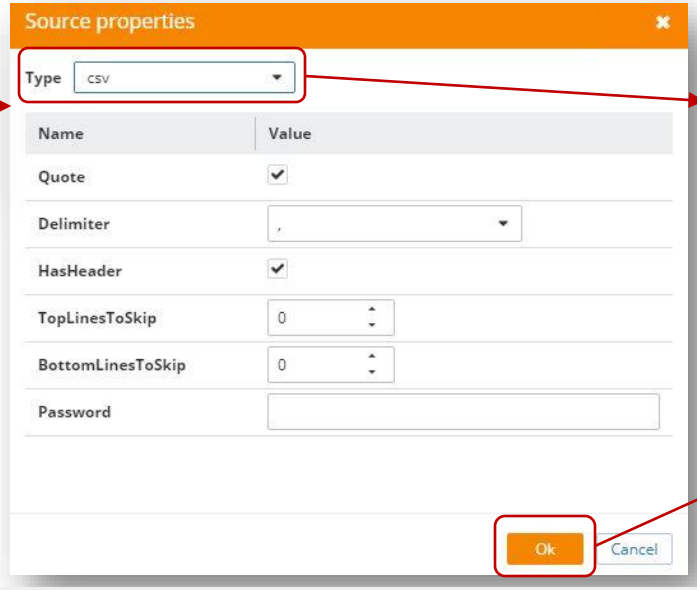
Data Source – where data from the file is extracted and preprocessed (file schema is edited, filters are applied, rows/columns are cut, etc.)

Transformations – where extracted data is rearranged according to provided spec

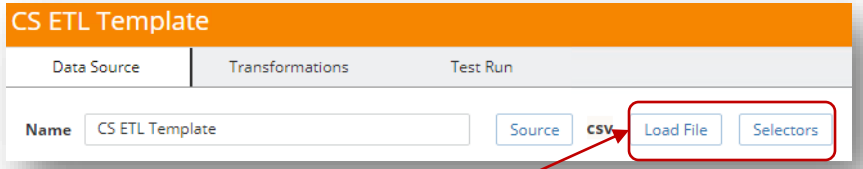
Test Run – where the ETL template is given a test execution

Data Source





- excel
- csv
- dat



Once Source properties are specified, two new buttons appear next to Source button:

- Load File
- Selectors

Press Source to specify the properties of extractor:

- Type of the Feed file
- If values are locked in ""
- Delimiter
- Is there is any header
- Top/Bottom lines to skip
- Password is file is protected

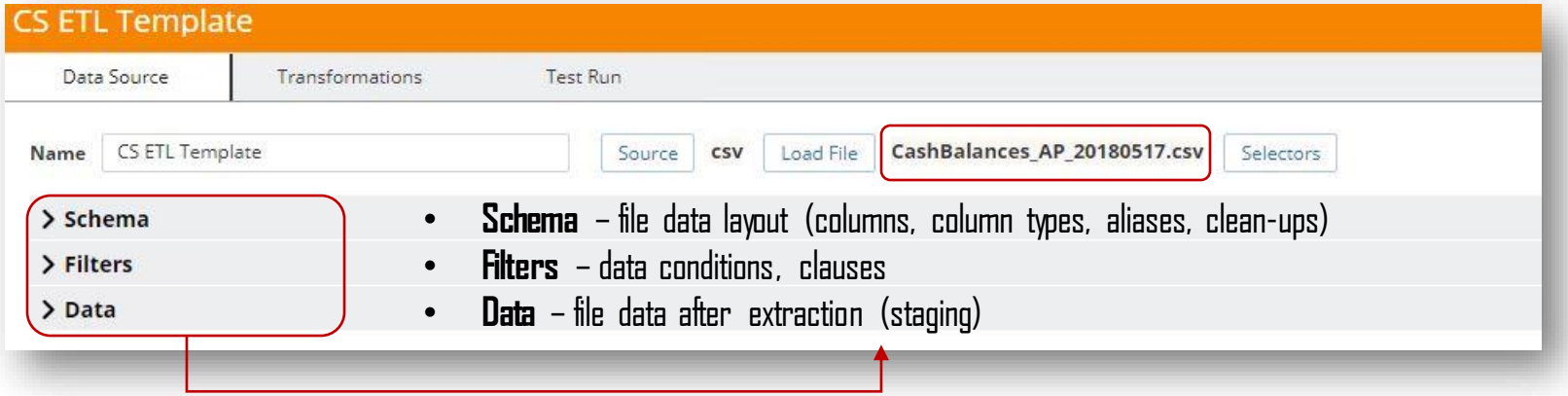
CashBalances_AP_20180517.csv

```
"Account Name / Account","Account Type","CCY","Trade Date Balance","Settle Date Balance","Exchange Rate","Trade Date Balance (USD)","Settle Date Balance (USD)","Account Type"
"CALEDONIA FUND / 0JTV","Margin","AUD",8485398.76000000,8485398.76000000,0.75165000,6378049.98000000,6378049.98000000,"Margin"
"CALEDONIA FUND / 0JTV","Margin","CAD",-4052.26000000,-4052.26000000,1.27880000,-3168.80000000,-3168.80000000,"Margin"
"CALEDONIA FUND / 0JTV","Margin","NZD",-4105.83000000,-4105.83000000,0.68975000,-2832.00000000,-2832.00000000,"Margin"
"CALEDONIA FUND / 0JTV","Margin","USD",-5453809.61000000,-5453809.61000000,1.00000000,-5453809.61000000,-5453809.61000000,"Margin"
"CALEDONIA ZILLOW FUND / 0JUA","Margin","CAD",192524.60000000,192524.60000000,1.27880000,150550.99000000,150550.99000000,"Margin"
"CALEDONIA ZILLOW FUND / 0JUA","Margin","USD",0.00000000,0.00000000,1.00000000,0.00000000,0.00000000,"Margin"
"CJH FAMILY TRUST / 0QLE","Margin","AUD",-57538233.32000000,-57538233.32000000,0.75165000,-43248613.07000000,-43248613.07000000,"Margin"
"CJH FAMILY TRUST / 0QLE","Margin","CAD",-2129829.68000000,-2129829.68000000,1.27880000,-1665490.84000000,-1665490.84000000,"Margin"
"CJH FAMILY TRUST / 0QLE","Margin","GBP",9753630.60000000,9753630.60000000,1.34835000,13151307.82000000,13151307.82000000,"Margin"
"CJH FAMILY TRUST / 0QLE","Margin","JPY",-242196479.00000000,-242196479.00000000,110.26750000,-2196444.82000000,-2196444.82000000,"Margin"
"MANDERRAH PL ATF GJJ FAM TRUST / 09VJ","Margin","AUD",-204195207.35000000,-204195207.35000000,0.75165000,-153483327.60000000,-153483327.60000000,"Margin"
"MANDERRAH PL ATF GJJ FAM TRUST / 09VJ","Margin","CAD",-4969331.55000000,-4969331.55000000,1.27880000,-3885933.34000000,-3885933.34000000,"Margin"
"MANDERRAH PL ATF GJJ FAM TRUST / 09VJ","Margin","GBP",17024675.71000000,17024675.71000000,1.34835000,22955221.49000000,22955221.49000000,"Margin"
"MANDERRAH PL ATF GJJ FAM TRUST / 09VJ","Margin","JPY",-356430298.00000000,-356430298.00000000,110.26750000,-3232414.79000000,-3232414.79000000,"Margin"
"MANDERRAH PL ATF GJJ FAM TRUST / 09VJ","Margin","USD",2915612.57000000,2915612.57000000,1.00000000,2915612.57000000,2915612.57000000,"Margin"
"VELCARA PTY LIMITED / 09VL","Margin","AUD",-4367728.70000000,-4367728.70000000,0.75165000,-3283003.28000000,-3283003.28000000,"Margin"
"VELCARA PTY LIMITED / 09VL","Margin","USD",17679277.49000000,17679277.49000000,1.00000000,17679277.49000000,17679277.49000000,"Margin"
```

Data Source

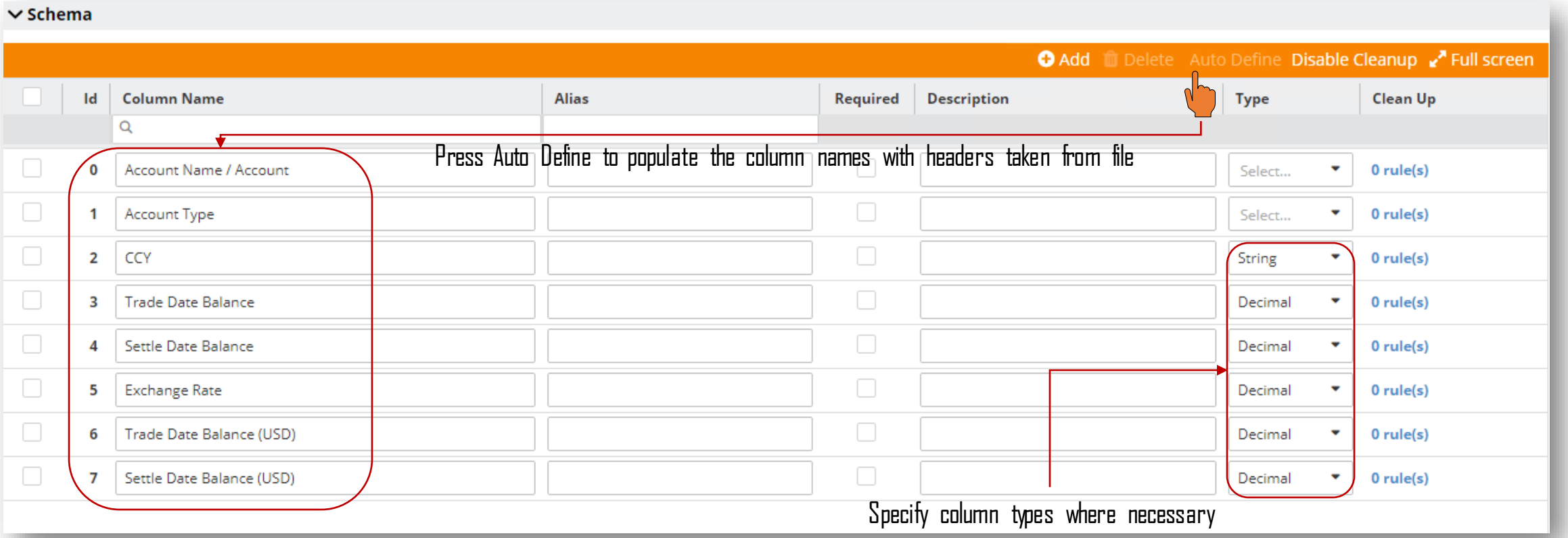
Load File

Press Load File to browse for the Feed file.
Once uploaded, its name appears next to the button



- **Schema** – file data layout (columns, column types, aliases, clean-ups)
- **Filters** – data conditions, clauses
- **Data** – file data after extraction (staging)

SCHEMA



Press Auto Define to populate the column names with headers taken from file

Specify column types where necessary

Data Source

Auto Define

Auto Define instrument is not a magic wand that creates headers all the time.
There are cases where headers should be defined manually:

- File is too complex
- No headers
- File format does not imply headers at all
- Etc....

Name: CS ETL Template | Source: csv | Load File: CashBalances_AP_20180517.csv | Selectors

Schema

Add | Delete | Auto Define | Disable Cleanup | Full screen

Id	Column Name	Alias	Required	Description	Type	Clean Up
0	Col0	Account Name / Account			String	0 rule(s)
1	Col1	Account Type			String	0 rule(s)
2	Col2	CCY			String	0 rule(s)
3	Col3	Trade Date Balance			Decimal	0 rule(s)
4	Col4	Settle Date Balance			Decimal	0 rule(s)
5	Col5	Exchange Rate			Decimal	0 rule(s)
6	Col6	Trade Date Balance (USD)			Decimal	0 rule(s)
7	Col7	Settle Date Balance (USD)			Decimal	0 rule(s)

For instance, if there are no headers in the files, the Auto Define instrument will insert Col1, Col2, Col3... in the quantity of file columns number.

Users can use Alias option to define reasonable column names. Aliases will be displayed on Transformations screen if defined.

Data Source

Filters

Filters

Add Delete

	Id	Name	Expression	Is Enabled
☐	1	AccType<>short	source['Account Type'] != 'Short'	☑

Press Add to add condition.

In this case, all records with Account Type 'Short' value are excluded from the extracted data.

Data

Data

Apply Filters ☑ Apply CleanUp ☑ Load Export Refresh Full screen

Account Name / Account	Account Type	CCY	Trade Date Balance	Settle Date Balance	Exchange Rate	Trade Date Balance (USD)	Settle Date Balance (USD)
CALEDONIA FUND / 0JTV	Margin	AUD	8485398.76	8485398.76	0.75165	6378049.98	6378049.98
CALEDONIA FUND / 0JTV	Margin	CAD	-4052.26	-4052.26	1.2788	-3168.8	-3168.8
CALEDONIA FUND / 0JTV	Margin	NZD	-4105.83	-4105.83	0.68975	-2832	-2832
CALEDONIA FUND / 0JTV	Margin	USD	-5453809.61	-5453809.61	1	-5453809.61	-5453809.61
CALEDONIA ZILLOW FUND / 0JUA	Margin	CAD	192524.6	192524.6	1.2788	150550.99	150550.99
CALEDONIA ZILLOW FUND / 0JUA	Margin	USD	0	0	1	0	0
CJH FAMILY TRUST / 0QLE	Margin	AUD	-57538233.32	-57538233.32	0.75165	-43248613.07	-43248613.07
CJH FAMILY TRUST / 0QLE	Margin	CAD	-2129829.68	-2129829.68	1.2788	-1665490.84	-1665490.84
CJH FAMILY TRUST / 0QLE	Margin	GBP	9753630.6	9753630.6	1.34835	13151307.82	13151307.82
CJH FAMILY TRUST / 0QLE	Margin	JPY	-242196479	-242196479	110.2675	-2196444.82	-2196444.82
MANDERRAH PL ATF GJJ FAM TRUST / 09VJ	Margin	AUD	-204195207.35	-204195207.35	0.75165	-153483327.6	-153483327.6

Press Load to load data from the Feed file. You can think of it as if the data is loaded into staging table on database.

In case of any errors, the data section remains empty. Error and error description will pop up at the bottom of the screen.

Data Source

Error! What do we do?

Data

Apply Filters ☑ Apply CleanUp ☑ Load

Account Name / Account	Account Type	CCY	Trade Date Balance	Settle Date Balance	Exchange Rate	Trade Date Bala
			No data			

10 25 50 100

Error	Description
Internal exception	String was not recognized as a valid Boolean.

Error	Description
Internal exception	String was not recognized as a valid Boolean.

Apparently, the data type of some column was not set correctly and extractor could not parse the file...

Let's return to the Schema and check out the Boolean type the error description pointed out

Schema

Id	Column Name	Type
0	Account Name / Account	Select...
1	Account Type	Select...
2	CCY	String
3	Trade Date Balance	Boolean
4	Settle Date Balance	Decimal
5	Exchange Rate	Decimal
6	Trade Date Balance (USD)	Decimal
7	Settle Date Balance (USD)	Decimal

D
Trade Date Balance
-8485398.76
-4052.26
-4105.83
-5453809.61
192524.6
0
-57538233.32

Data Source

Clean ups – transformations of data values on Schema section. Use Clean ups to modify the data when necessary

Trade Date Balance
NaN
NaN
NaN

Example: in Feed file zeros are defined as NaN, but not 0.
We need to replace NaN with simple "0" value.
Otherwise, error will be shown since NaN can't be decimal.

Schema

Add | Delete | Auto Define | Disable Cleanup | Full screen

	Id	Column Name	Alias	Required	Description	Type	Clean Up
	0	Account Name / Account				Select...	0 rule(s)
	1	Account Type				Select...	0 rule(s)
	2	CCY				String	0 rule(s)
	3	Trade Date Balance				Decimal	2 rule(s)

Press 0 rules to open up the Clean up rules pop-up

Clean up rules for item: Col3

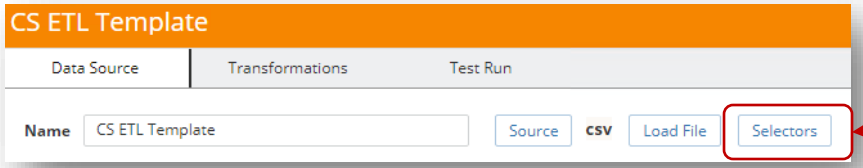
Add | Delete

	Enabled	Order	Condition	Condition Argument	Action	Expression
	✔	1	Equal	NaN	ReplaceMatched	0

Ok | Cancel

Press Add to add a condition.
In this case, we replace NaN with '0'

Data Source



Let's get back to the upper toolbar and review the button Selectors

Selector is a tool that helps take particular value from specified file. Taken value can be used in further transformations.

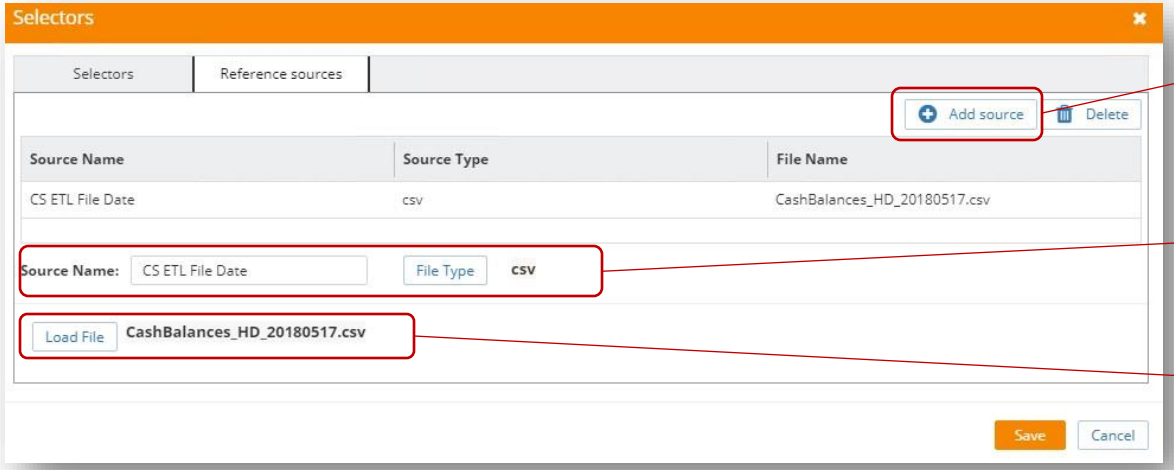
For example, the first string of the Feed file is Date.
We need this date in further transformations.
Let's grasp the value from column B2.

	A	B	C	D	E	F
1	Date	20180605				
2	Account Name / Account	Account Type	CCY	Trade Date Balance	Settle Date Balance	Exchange Rate
3	CALEDONIA FUND / 0JTV	Margin	AUD	-8485398.76	8485398.76	0.75165
4	CALEDONIA FUND / 0JTV	Margin	CAD	-4052.26	-4052.26	1.2788
5	CALEDONIA FUND / 0JTV	Margin	NZD	-4105.83	-4105.83	0.68975
6	CALEDONIA FUND / 0JTV	Margin	USD	-5453809.61	-5453809.61	1
7	CALEDONIA ZILLOW FUND / 0JUA	Margin	CAD	192524.6	192524.6	1.2788

Press button to open up the pop up screen Selectors.

Data Source

Tab **Reference sources** – specification of the file from which value will be taken

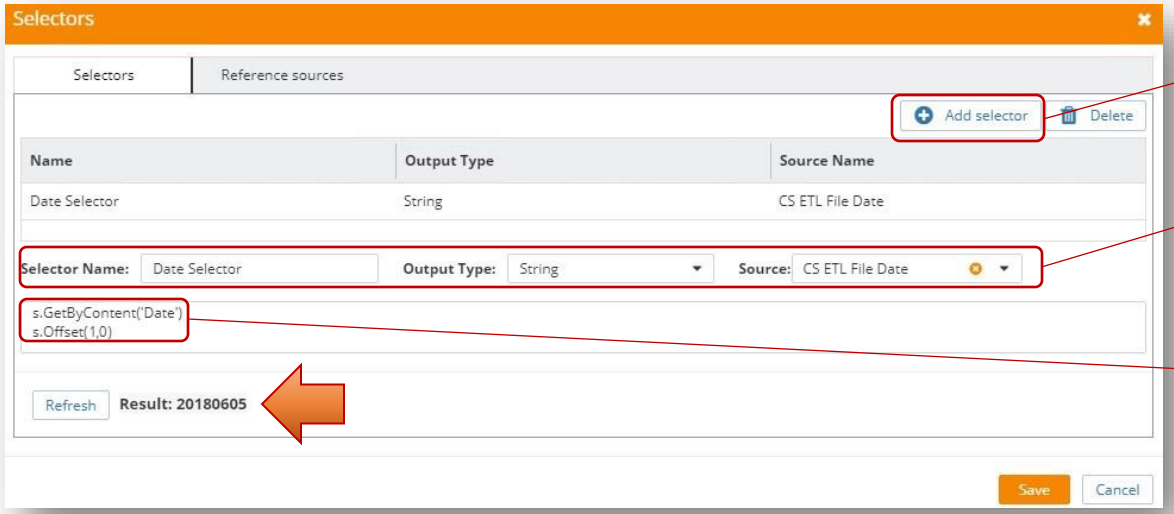


Press **Add source** to create new source for Selector

Specify **Source name** and **Type**

Press **Load** to browse for file. Its name appears next to the button

Tab **Selectors** – specification of the value taken from the Feed file



Press **Add selector** to create new Selector

Specify **Selector Name**, **Output Type** and **Source** (created in tab Reference sources)

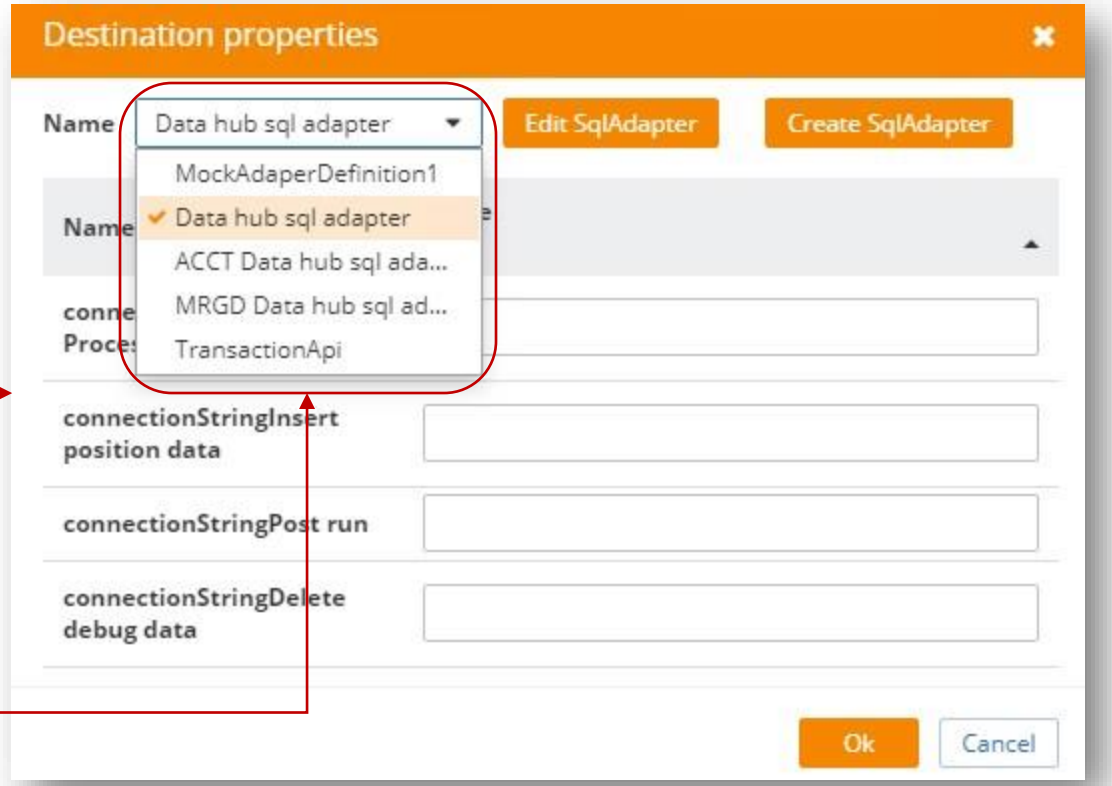
Write the **expression**.
In this case, we search for the value Date and step 1 cell right:

```
s.GetByContent('Date')
s.Offset(1,0)
```

Transformations

Transformations tab - tool that allows users to set up values extracted from the Feed file against system arguments.
The setup is executed according to the provided spec.

Press Setup destination to open up Destination properties pop-up

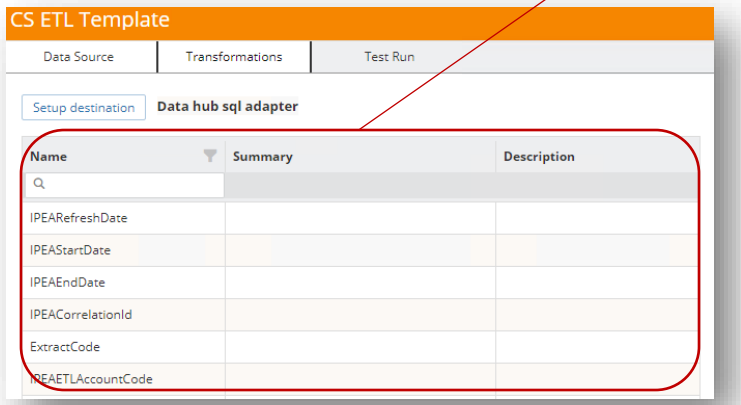


Select the SQL adapter.

SQL adapter is a **preinstalled** set of SQL scripts that operate with extracted data. We will explicitly analyze this entities later. For now we select the **Data hub sql adapter**, which has a set of fields the extracted data will be mapped on to.

Transformations

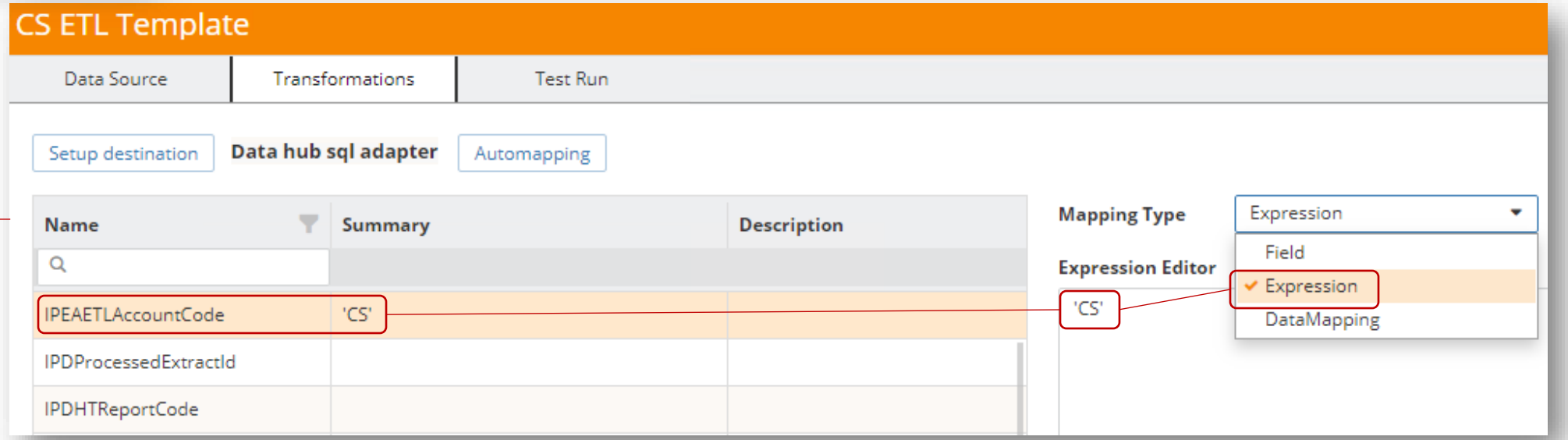
After the **SQL adapter** is selected, Argument table appears below. We will map values from extracted Feed data onto these Arguments according to the provided spec.



Let's say we've got the spec:

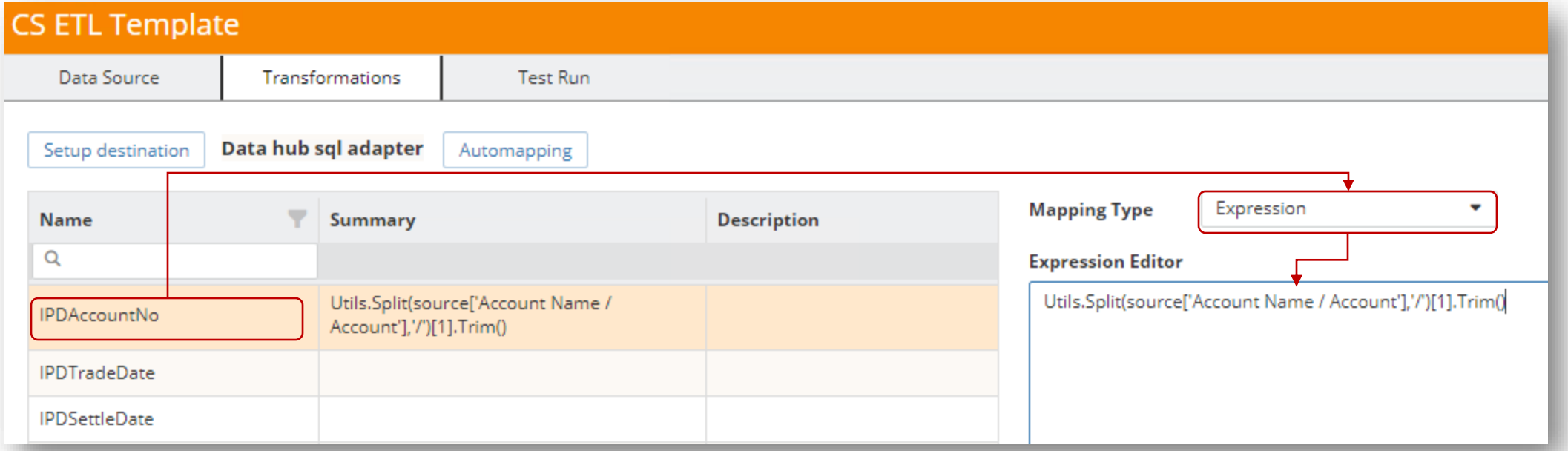
- Account Code = constant 'CS'
- Account No = **0JTV** from values like 'CALEDONIA FUND / 0JTV' from column Account Name / Account
- Settle Date = date from the file name in yyyyMMdd format
- Trade Date = same as Settle Date
- Security CCY Code = value of CCY column
- FX Rate = value of Exchange Rate column

Let's proceed to the mapping of Account Code.
It's just constant value 'CS'



Transformations

Account No = **OJTV** from values like **'CALEDONIA FUND / OJTV'** from column Account Name / Account



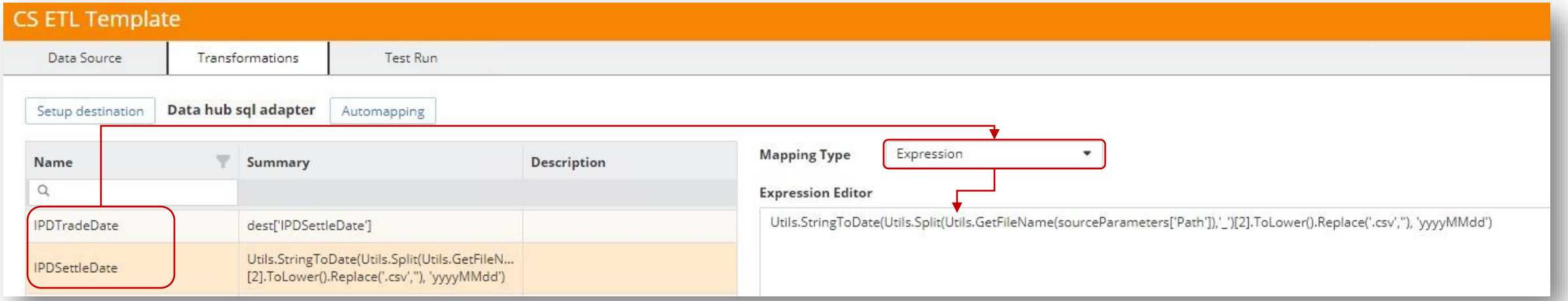
Write the following expression into the Expression Editor

```
Utils.Split(source['Account Name / Account'],'/')[1].Trim()
```

Transformations

Settle Date = date from the file name in **yyyyMMdd** format
Trade Date = same as **Settle Date**

CashBalances_AP_20180517.csv



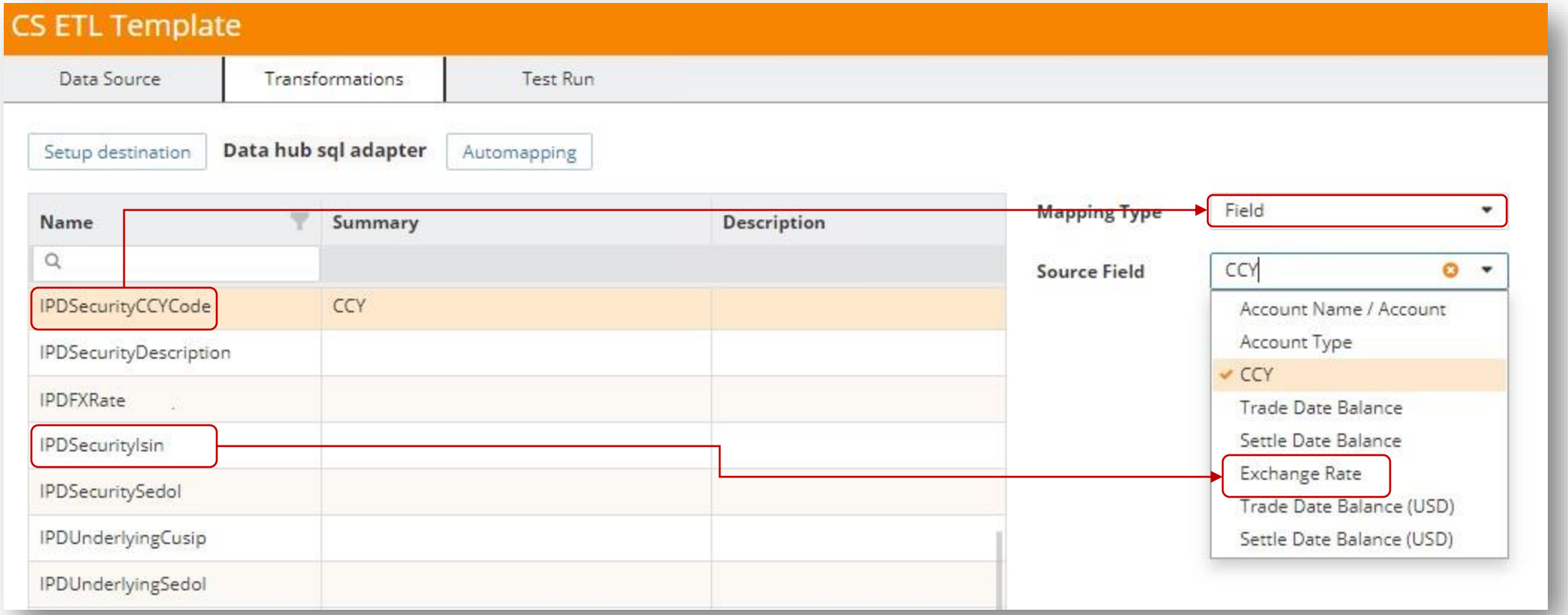
Write the following expressions into the Expression Editor

```
Utils.StringToDate(Utils.Split(Utils.GetFileName(sourceParameters['Path']),'_')[2].ToLower().Replace('.csv',''), 'yyyyMMdd')
```

```
dest['IPDSettleDate']
```

Transformations

Security CCY Code = value of **CCY** column
FX Rate = value from **Exchange Rate** column



Transformations

Didn't you forget about the Date value we extracted with the help of Selector? Just a reminder here

Say, we need this date to be mapped to Argument **Start Date**. Let's do it with a help of Expression

Selectors

Selectors | Reference sources

Add selector | Delete

Name	Output Type	Source Name
Date Selector	String	CS ETL File Date

Selector Name: Date Selector Output Type: String Source: CS ETL File Date

```
s.GetByContent('Date')
s.Offset(1,0)
```

Refresh Result: 20180605

Save | Cancel

Write the following expression into the Expression Editor

```
ref['Date Selector']
```

CS ETL Template

Data Source | Transformations | Test Run

Setup destination | Data hub sql adapter | Automapping

Name	Summary	Description
IPEAStartDate	ref['Date Selector']	

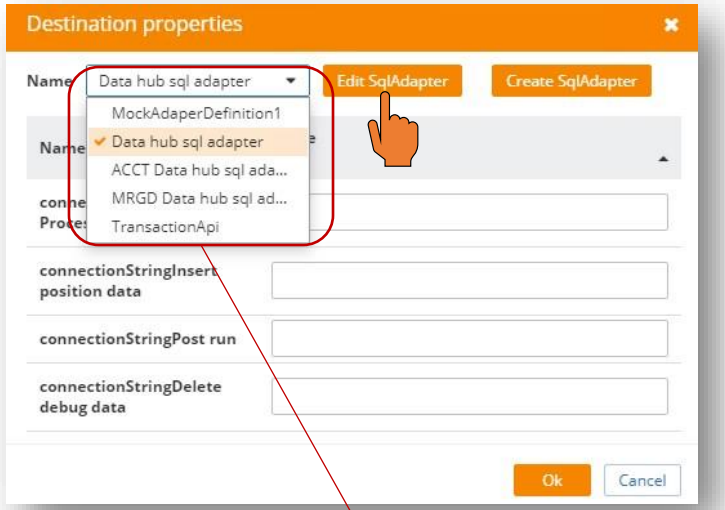
Mapping Type: Expression

Expression Editor

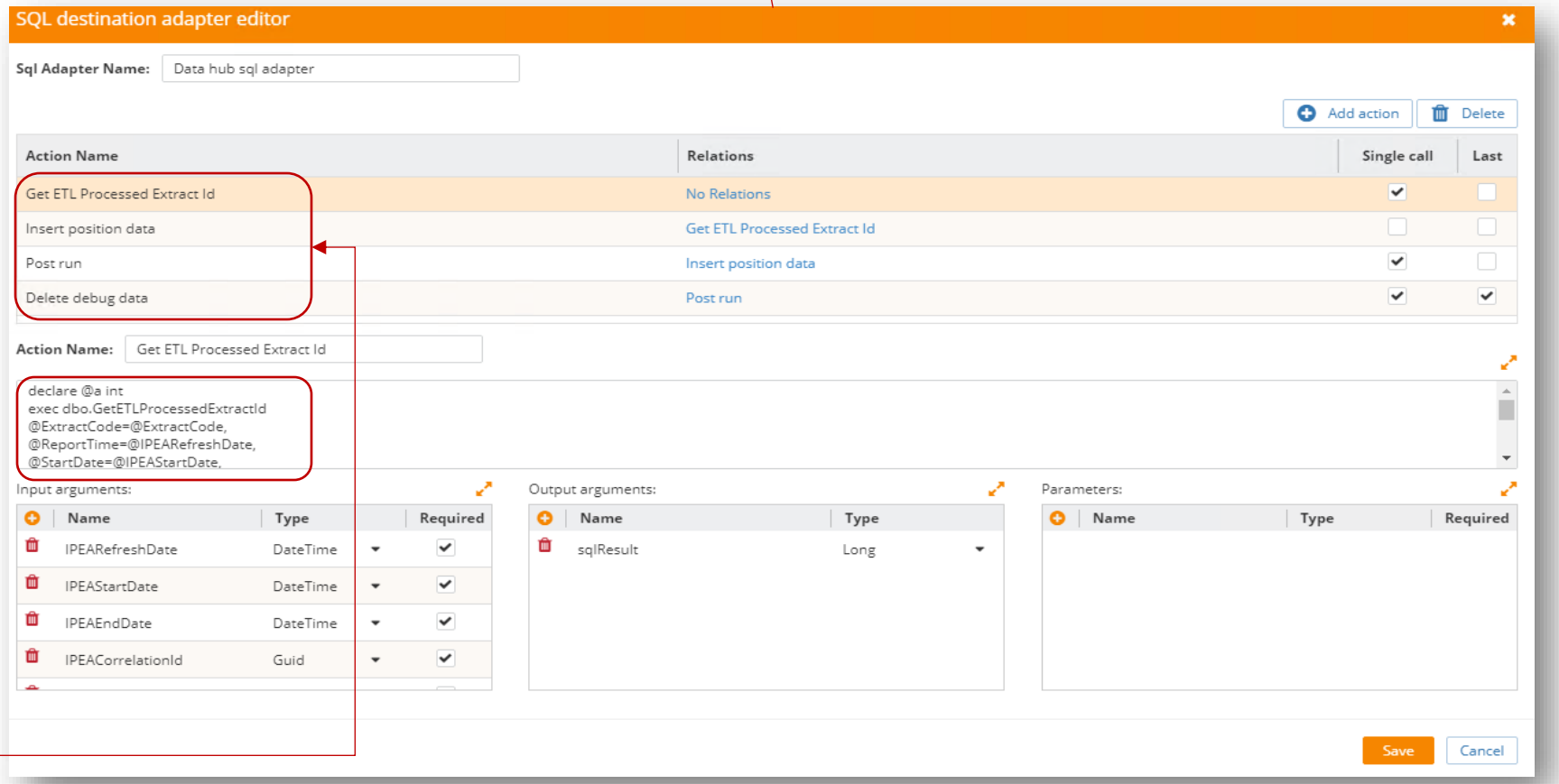
ref['Date Selector']

Transformations

Some Arguments are required by **SQL adapter** selected in Destination properties pop-up. They are used in Actions SQL adapter and need to be specified even if they are absent in spec. Let's get back to the Destination properties screen and explain the term of SQL adapter.



In our case we have Data hub sql adapter selected. Press Edit Sql adapter button to get to the editor



SQL Adapter – collection of ordered Actions. SQL Adapters are preinstalled in Sync Manager and should be modified by Development team.

Action – SQL script

Transformations

Gets Processed Extract Id – unique identifier for the set of data extracted from the Feed document

Inserts extracted data into Position using the value mapping created

Runs stored procedure ETLPostRunStoreProc

Cleans up debug data created during Test Run

Arguments used in the SQL script. They need to be specified in Arguments table

Order of execution

Checked: Action is applied once for the data array
Not checked: Action is applied to all rows in the data array

Last to execute

SQL destination adapter editor

Sql Adapter Name: Data hub sql adapter

Add action | Delete

Action Name	Relations	Single call	Last
Get ETL Processed Extract Id	No Relations	☑	☐
Insert position data	Get ETL Processed Extract Id	☐	☐
Post run	Insert position data	☑	☐
Delete debug data	Post run	☑	☑

Action Name: Get ETL Processed Extract Id

```
declare @a int
exec dbo.GetETLProcessedExtractId
@ExtractCode=@ExtractCode,
@ReportTime=@IPEARefreshDate,
@StartDate=@IPEAStartDate,
```

Input arguments:

Name	Type	Required
IPEARefreshDate	DateTime	☑
IPEAStartDate	DateTime	☑
IPEAEndDate	DateTime	☑
IPEACorrelationId	Guid	☑

Output arguments:

Name	Type
sqlResult	Long

Parameters:

Name	Type	Required

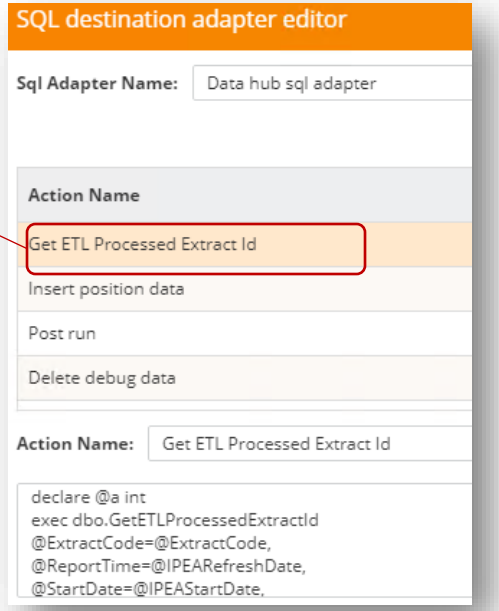
Save | Cancel

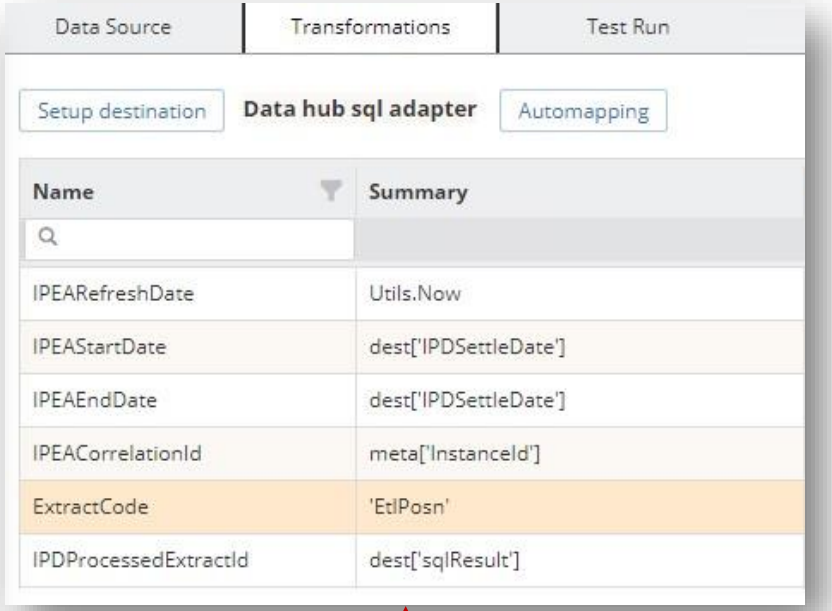
Transformations

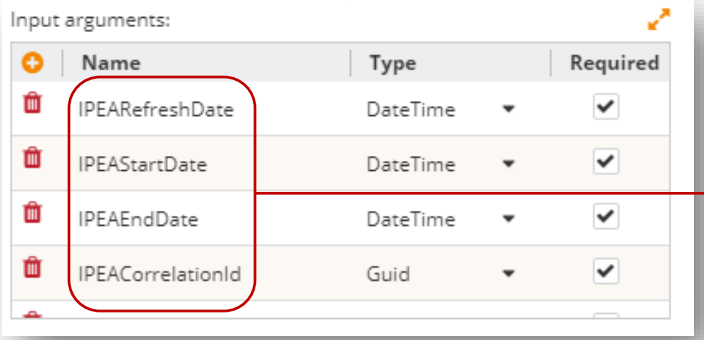
We said that some Arguments are required by **SQL adapter** selected in Destination properties pop-up. The only Action in **Data hub sql adapter** that requires Arguments is Action **Get ETL Processed Extract Id**

Let's see the SQL script it runs:

```
declare @a int
exec dbo.GetETLProcessedExtractId
@ExtractCode=@ExtractCode,
@ReportTime=@IPEARefreshDate,
@StartDate=@IPEAStartDate,
@EndDate=@IPEAEndDate,
@AllAccounts=1,
@AccountCode=@IPEAETLAccountCode,
@Debug=@isDebug,
@ProcessedExtractID=@a output
select @a
```







Specify the Arguments in the table accordingly

Transformations

IMPORTANT!

Sometimes Actions do not only use arguments with mapped values but produce output arguments, that, in turn, are used in other Actions.

In this case, action **Get ETL Processed Extract Id** produces Output argument **sqlResult**.
It stores the number of Processed Extract Id and should be applied to every row in the loaded data.

SQL destination adapter editor

Sql Adapter Name: Data hub sql adapter

Add action | Delete

Action Name	Relations	Single call	Last
Get ETL Processed Extract Id	No Relations	✓	
Insert position data	Get ETL Processed Extract Id		
Post run	Insert position data	✓	
Delete debug data	Post run	✓	✓

Action Name: Get ETL Processed Extract Id

```
declare @a int
exec dbo.GetETLProcessedExtractId
@ExtractCode=@ExtractCode,
@ReportTime=@IPEARefreshDate,
@StartDate=@IPEAStartDate,
```

Input arguments:

Name	Type	Required
IPEARefreshDate	DateTime	✓
IPEAStartDate	DateTime	✓
IPEAEndDate	DateTime	✓
IPEACorrelationId	Guid	✓

Output arguments:

Name	Type
sqlResult	Long

Parameters:

Name	Type	Required

Save | Cancel

Value of **sqlResult** argument is used in the next action **Insert position data**
Let's take a closer look

Single call setting shows if the Action is executed once or against every row in the loaded data.

Transformations

Value of **sqlResult** argument is used in the action **Insert position data**

SQL destination adapter editor

Sql Adapter Name: Data hub sql adapter

Action Name
Get ETL Processed Extract Id
Insert position data
Post run
Delete debug data

Action Name: Insert position data

```
insert into Etl_Posn
(ProcessedExtractId,
HTReportCode,
ClientCode,
ClientName,
```

Input arguments:

Name	Type	Required
IPDProcessedExtractId	Long	
IPDHTReportCode	String	
IPDClientCode	String	
IPDClientName	String	

Setup destination | **Data hub sql adapter** | Automapping

Name	Summary	Description
IPDProcessedExtractId	dest['sqlResult']	

Since we specified the mapping of argument Processed Extract Id to get data from argument sqlResult, the Action Insert position data calls execution of action Get ETL Processed Extract Id.

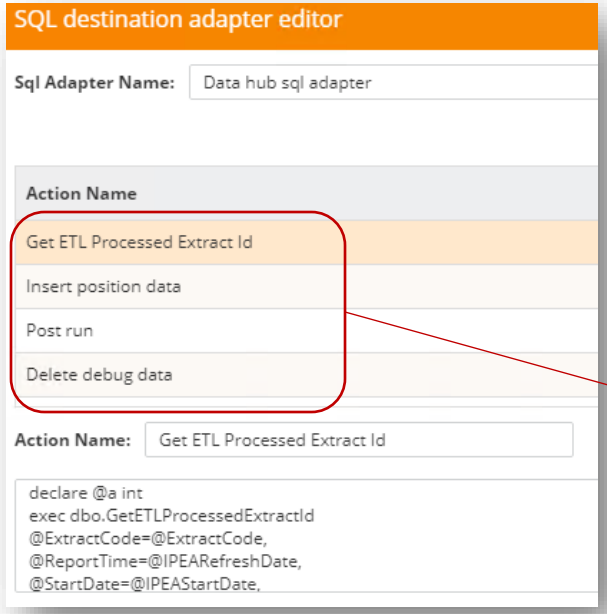
Now remember!

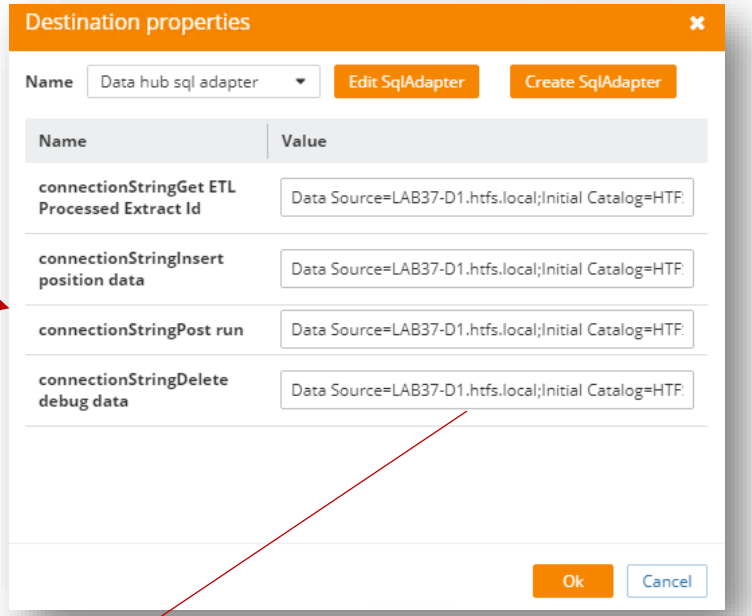
Action Get ETL Processed Extract Id is specified to be Single call only! In this case, action Insert position data that processes every row of loaded data, will use one value of Processed Extract Id argument on every affected row.

Action Name	Relations	Single call	Last
Get ETL Processed Extract Id	No Relations	✓	
Insert position data	Get ETL Processed Extract Id		

Transformations

Every Action in SQL adapter has to have a connection string. Connection string is a pointer to the location where the Action SQL script will be executed





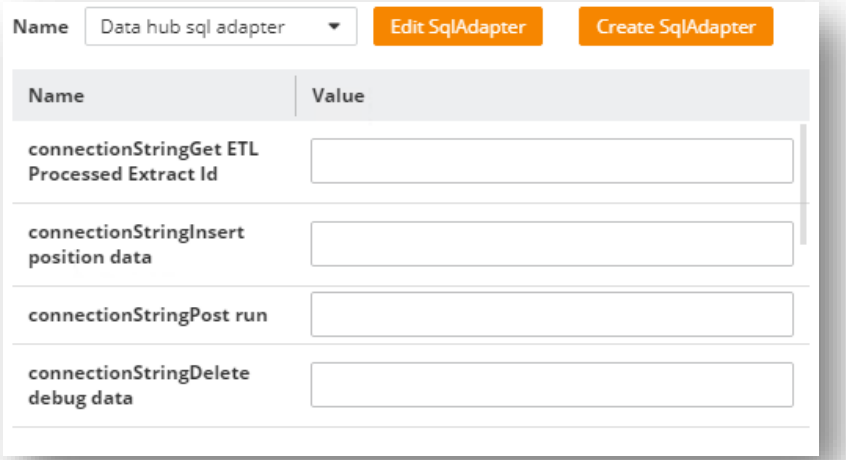

```
Data Source=LAB37-D1.htfs.local;Initial Catalog=HTFSExtracts;Integrated Security=SSPI
```

IMPORTANT!

Values of connection strings on the screen Destination properties better be not specified.

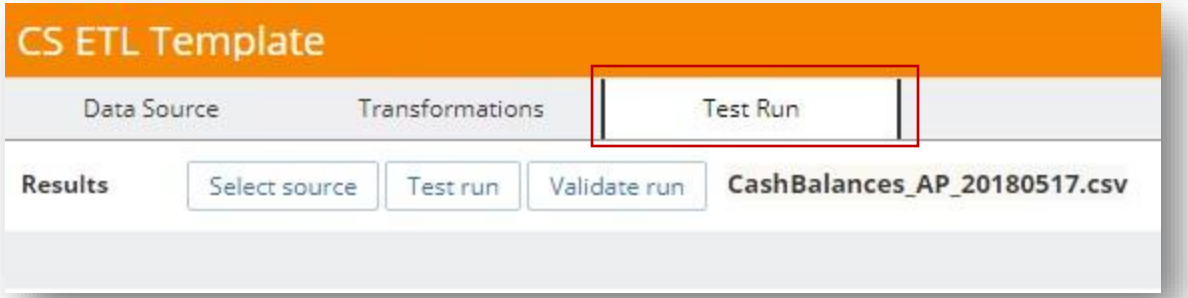
WHY?

Because all ETL arguments are specified either on Test Run, or on Scheduler or in Workflow Template.

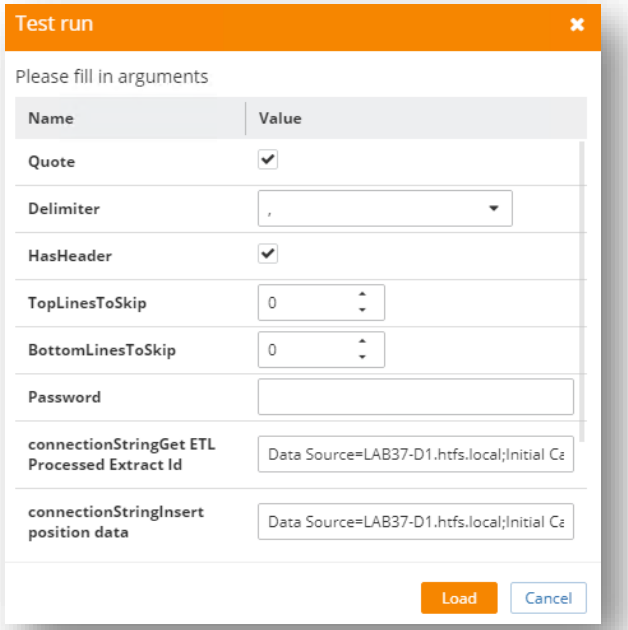


Test Run

Test Run is a tool that allows to provide a test execution of created ETL template



When starting a Test Run, specify all ETL Arguments including connection strings



Select source — Browse for the file that is processed in the ETL Template.

Test run — Test Run actually processes the Feed file when runs the ETL template.

Validate run — Validation table demonstrates result as if the file is processed. File is not actually processed. Flag IsDebug set to true , so the file is not processed.

Results | Select source | Test run | Validate run | CashBalances_AP_20180517.csv | Export to csv | Columns chooser

	IPEARefreshDate	IPDSettleDate	IPEAStartDate	IPEAEndDate	IPEACorrelationId	ExtractCode	IPEAETLAccountCode		sqlResult
>	2018-06-07T12:14:44.364618+00:00	2018-05-17T00:00:00	2018-05-17T00:00:00	2018-05-17T00:00:00	408f9f1d-046b-4e48-9033-52fbd84c9111	EtlPosn	CS	30039	30039
>	2018-06-07T12:14:44.364618+00:00	2018-05-17T00:00:00	2018-05-17T00:00:00	2018-05-17T00:00:00	408f9f1d-046b-4e48-9033-52fbd84c9111	EtlPosn	CS	30039	30039
>	2018-06-07T12:14:44.3656298+00:00	2018-05-17T00:00:00	2018-05-17T00:00:00	2018-05-17T00:00:00	408f9f1d-046b-4e48-9033-52fbd84c9111	EtlPosn	CS	30039	30039

Workflow Template

Configured ETL Template appears on the ETL Templates grid. You can modify it any time.

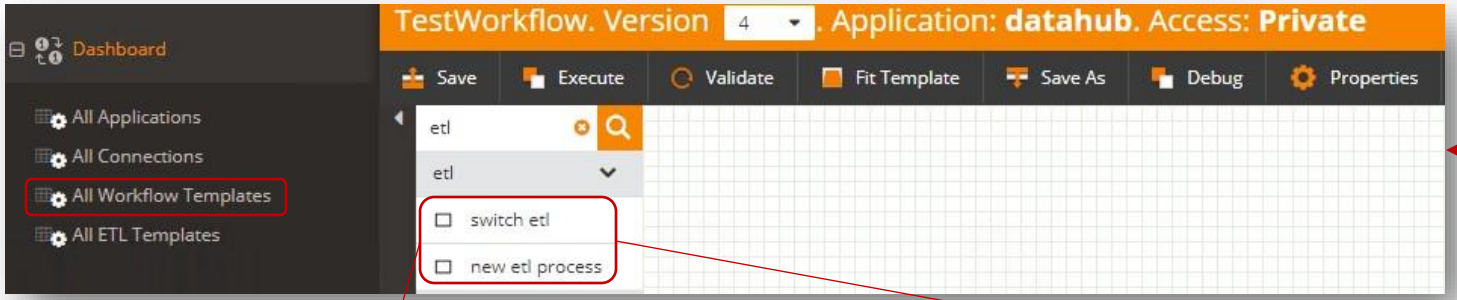
Etl Templates | Template Id | Application Select...

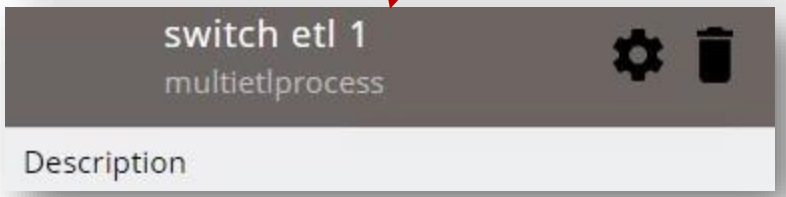
Create template | Delete | Clone

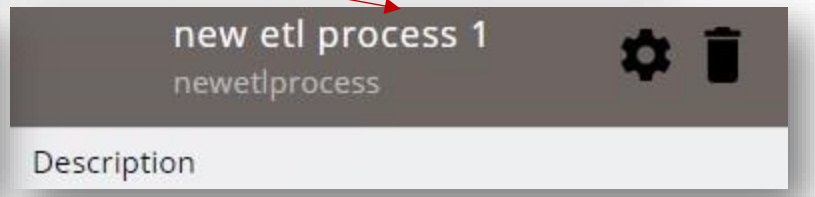
Name	Created By	Created On	Modified By	Modified On
CS ETL Template	HTFSSupport	2018/06/05 16:57:13	HTFSSupport	2018/06/05 16:57:13
Super Template	HTFSSupport	2018/06/05 13:56:26	HTFSSupport	2018/06/05 14:46:55
TestEtl	HTFSSupport	2018/05/28 11:17:29	HTFSSupport	2018/05/30 08:45:57

Created ETL Template can be scheduled for execution in Sync Manager Scheduler.

However, more frequently they will be used in Workflow Templates. Let's take a quick look on the Workflow Template **activities** that use ETL Templates

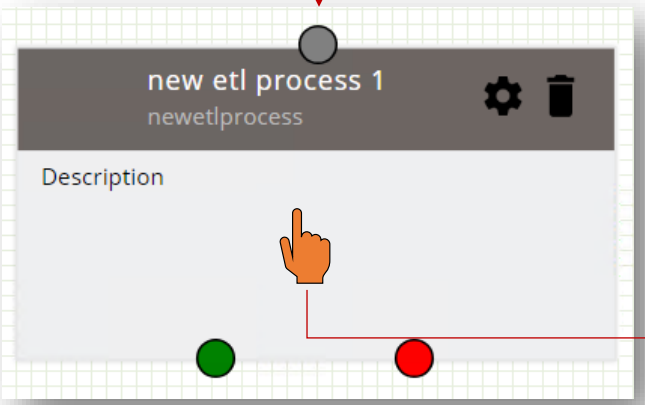






Workflow Template

New ETL Process activity – simple execution of one ETL Template

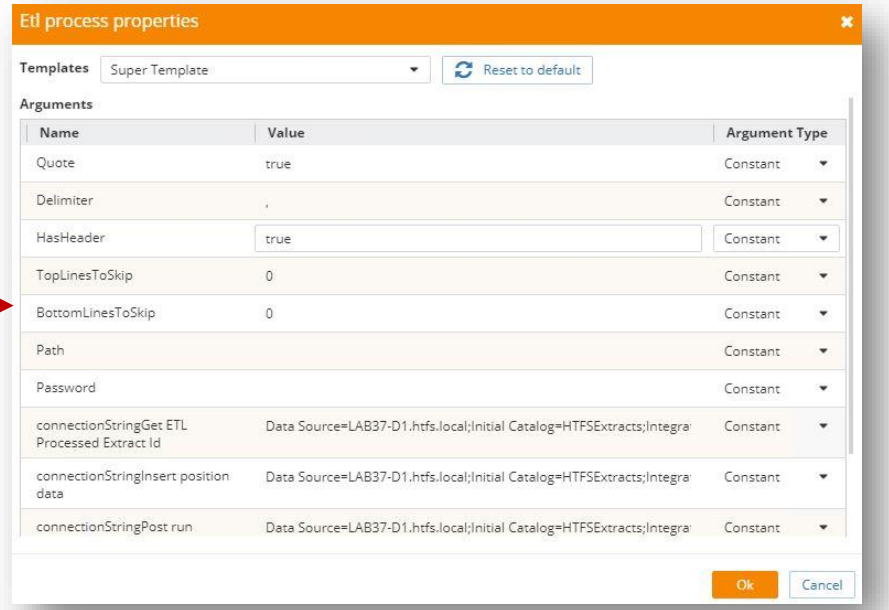


Double click on the activity to open Properties pop-up.
Select ETL template from the Templates dropdown

Once ETL Template is selected, specify the Arguments, including connection strings.

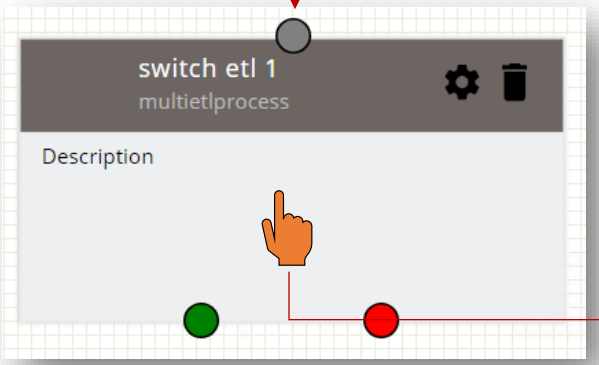
NOTE!

This is why the connection strings Arguments were not specified in the ETL template. Agent can use one ETL template is multiple Workflow Templates with different set of Arguments for every Workflow Template.



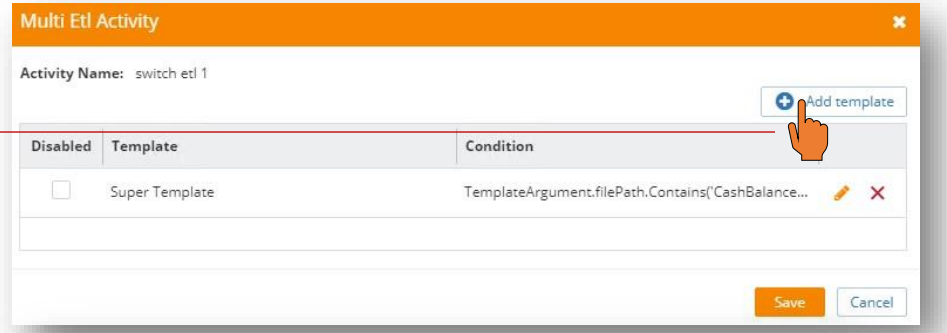
Workflow Template

Switch ETL activity – can execute multiple ETL Templates checking them against specified conditions.



Double click on the activity to open Multi Etl Activity properties pop-up.

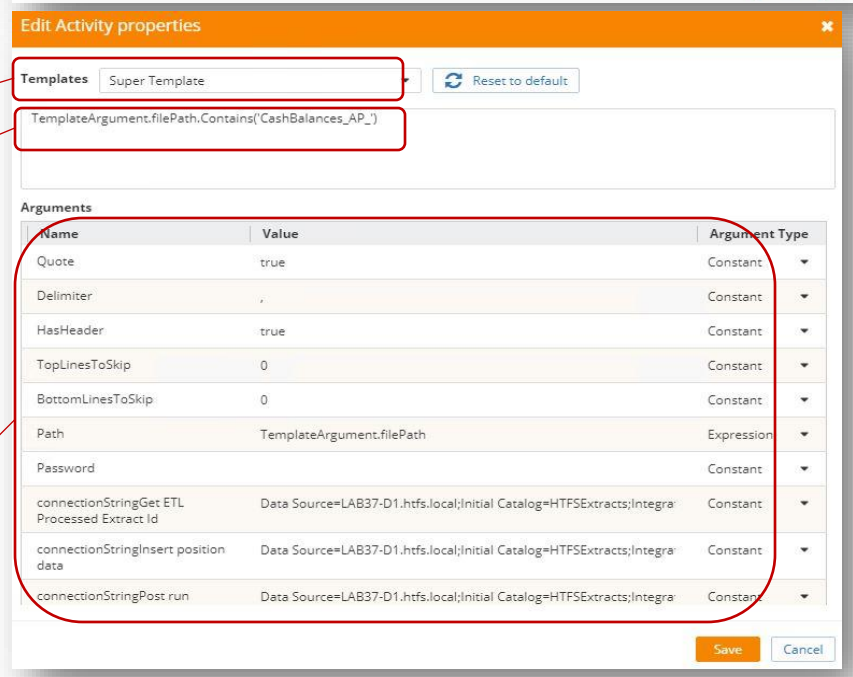
Press Add Template to open Activity properties.



Select ETL Template from dropdown

Insert expression of a condition when ETL Template is executed

Specify the Arguments, including connection strings



Thank you for your attention!