Techniques for User Code in SAS® Data Integration Studio

Scenario

- A SAS program is developed to derive moving averages and other moving measures for transactional data.
- The program is generalized with macro variables as a general purpose utility.
- I will show 2 ways to implement this custom program as a step in a SAS Data Integration Studio job:
  - via a User Written transformation
  - via a user-defined custom transformation designed with the New Transformation wizard

This will serve as an example to demonstate how to implement any custom SAS code in SAS Data Integration Studio jobs.

Example SAS Program: What It Does

Stock	Date	AdjClose
IBM	01AUG1986	23.04
IBM	02SEP1986	22.33
IBM	01OCT1986	20.53
IBM	03NOV1986	21.29
IBM	01DEC1986	20.10
IBM	02JAN1987	21.57
IBM	02FEB1987	23.56
IBM	02MAR1987	25.35
IBM	01APR1987	27.04
IBM	01MAY1987	27.20
IBM	01JUN1987	27.63
IBM	01JUL1987	27.37

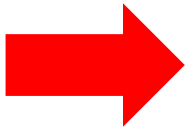
source data

Stock	Date	AdjClose	ave_adjClose	max_adjClose	min_adjClose	range_adjClose
IBM	01AUG1986	23.04	.	.	.	.
IBM	02SEP1986	22.33	.	.	.	.
IBM	01OCT1986	20.53	.	.	.	.
IBM	03NOV1986	21.29	.	.	.	.
IBM	01DEC1986	20.10	.	.	.	.
IBM	02JAN1987	21.57	21.477	23.04	20.10	2.94
IBM	02FEB1987	23.56	21.563	23.56	20.10	3.46
IBM	02MAR1987	25.35	22.067	25.35	20.10	5.25
IBM	01APR1987	27.04	23.152	27.04	20.10	6.94
IBM	01MAY1987	27.20	24.137	27.20	20.10	7.10
IBM	01JUN1987	27.63	25.392	27.63	21.57	6.06
IBM	01JUL1987	27.37	26.358	27.63	23.56	4.07

target data

Moving Measures

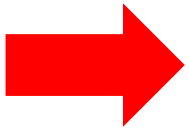
Stock	Date	AdjClose
IBM	01AUG1986	23.04
IBM	02SEP1986	22.33
IBM	01OCT1986	20.53
IBM	03NOV1986	21.29
IBM	01DEC1986	20.10
IBM	02JAN1987	21.57
IBM	02FEB1987	23.56
IBM	02MAR1987	25.35
IBM	01APR1987	27.04
IBM	01MAY1987	27.20
IBM	01JUN1987	27.63
IBM	01JUL1987	27.37



Stock	Date	AdjClose	ave_adjClose	max_adjClose	min_adjClose	range_adjClose
IBM	01AUG1986	23.04	.	.	.	.
IBM	02SEP1986	22.33	.	.	.	.
IBM	01OCT1986	20.53	.	.	.	.
IBM	03NOV1986	21.29	.	.	.	.
IBM	01DEC1986	20.10	.	.	.	.
IBM	02JAN1987	21.57	21.477	23.04	20.10	2.94
IBM	02FEB1987	23.56	21.563	23.56	20.10	3.46
IBM	02MAR1987	25.35	22.067	25.35	20.10	5.25
IBM	01APR1987	27.04	23.152	27.04	20.10	6.94
IBM	01MAY1987	27.20	24.137	27.20	20.10	7.10
IBM	01JUN1987	27.63	25.392	27.63	21.57	6.06
IBM	01JUL1987	27.37	26.358	27.63	23.56	4.07

Moving Measures

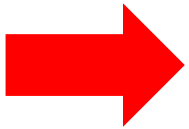
Stock	Date	AdjClose
IBM	01AUG1986	23.04
IBM	02SEP1986	22.33
IBM	01OCT1986	20.53
IBM	03NOV1986	21.29
IBM	01DEC1986	20.10
IBM	02JAN1987	21.57
IBM	02FEB1987	23.56
IBM	02MAR1987	25.35
IBM	01APR1987	27.04
IBM	01MAY1987	27.20
IBM	01JUN1987	27.63
IBM	01JUL1987	27.37



Stock	Date	AdjClose	ave_adjClose	max_adjClose	min_adjClose	range_adjClose
IBM	01AUG1986	23.04	.	.	.	.
IBM	02SEP1986	22.33	.	.	.	.
IBM	01OCT1986	20.53	.	.	.	.
IBM	03NOV1986	21.29	.	.	.	.
IBM	01DEC1986	20.10	.	.	.	.
IBM	02JAN1987	21.57	21.477	23.04	20.10	2.94
IBM	02FEB1987	23.56	21.563	23.56	20.10	3.46
IBM	02MAR1987	25.35	22.067	25.35	20.10	5.25
IBM	01APR1987	27.04	23.152	27.04	20.10	6.94
IBM	01MAY1987	27.20	24.137	27.20	20.10	7.10
IBM	01JUN1987	27.63	25.392	27.63	21.57	6.06
IBM	01JUL1987	27.37	26.358	27.63	23.56	4.07

Moving Measures

Stock	Date	AdjClose
IBM	01AUG1986	23.04
IBM	02SEP1986	22.33
IBM	01OCT1986	20.53
IBM	03NOV1986	21.29
IBM	01DEC1986	20.10
IBM	02JAN1987	21.57
IBM	02FEB1987	23.56
IBM	02MAR1987	25.35
IBM	01APR1987	27.04
IBM	01MAY1987	27.20
IBM	01JUN1987	27.63
IBM	01JUL1987	27.37



Stock	Date	AdjClose	ave_adjClose	max_adjClose	min_adjClose	range_adjClose
IBM	01AUG1986	23.04	.	.	.	.
IBM	02SEP1986	22.33	.	.	.	.
IBM	01OCT1986	20.53	.	.	.	.
IBM	03NOV1986	21.29	.	.	.	.
IBM	01DEC1986	20.10	.	.	.	.
IBM	02JAN1987	21.57	21.477	23.04	20.10	2.94
IBM	02FEB1987	23.56	21.563	23.56	20.10	3.46
IBM	02MAR1987	25.35	22.067	25.35	20.10	5.25
IBM	01APR1987	27.04	23.152	27.04	20.10	6.94
IBM	01MAY1987	27.20	24.137	27.20	20.10	7.10
IBM	01JUN1987	27.63	25.392	27.63	21.57	6.06
IBM	01JUL1987	27.37	26.358	27.63	23.56	4.07

The Program

```
%let source=stocks;
%let n = 6;
%let statvar=high;
%let groupvar=stock;
%let datevar=date;
```

These macro variables specify
- the source data
- the number of observations (time periods) for each summary calculation
- the numeric variable on which to calculate moving statistics
- the grouping variable
- the time variable.

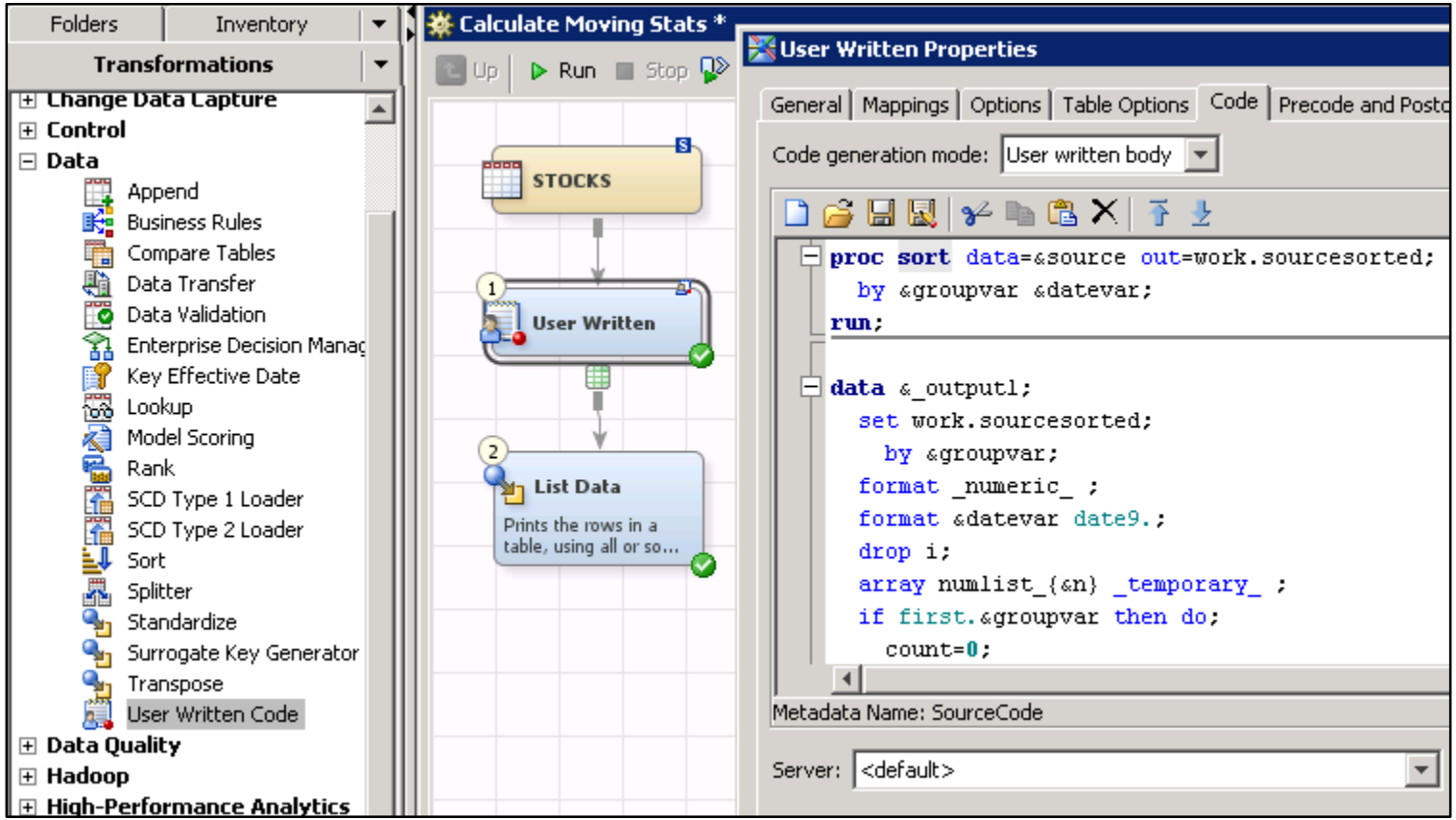
```
data movingstatsfor_&source;
  set work.sourcesorted;
    by &groupvar;
  drop i;
  array numlist_{&n} _temporary_ ;
  if first.&groupvar then do;
    count=0;
    do i=1 to &n;
      numlist_{i}=.;
    end;
  end;
  count+1;
  do i=&n to 2 by -1;
     numlist_{i}=numlist_{i-1};
  end;
  numlist_{1}=&statvar;
  if count GE &n then do;
    ave_&statvar=sum(of numlist_{*}) / &n;
    min_&statvar=min(of numlist_{*});
    max_&statvar=max(of numlist_{*});
    range_&statvar=max_&statvar-min_&statvar;
  end;
run;
```

Scenario

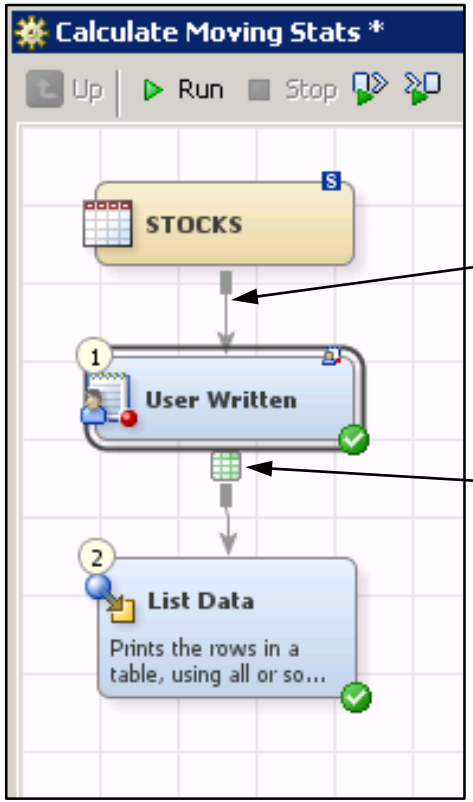
There are two ways to implement this code utility in a Data Integration Studio job:

1. Use the User Written transformation.
   - Source tables can be attached as input. Users still need to modify the %LET statements as needed. The ***update metadata*** utility is applied by the user to the target table within the job to make the target table usable by other transformations in the job.
2. Create a new custom transformation.
   - This enables users to use the transformation in the same way they use other SAS Data Integration Studio transformations. Users do not alter the underlying code and instead use transformation properties to control how the transformation works.

User Written Transformation



User Written Transformation



You can connect source tables to the User Written transformation.

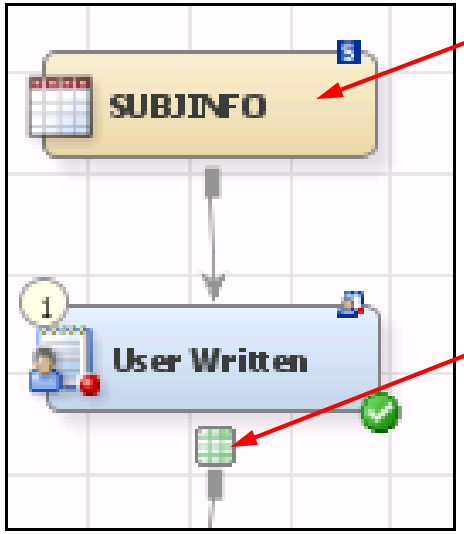
Temporary or permanent target tables can be created and connected to subsequent transformations.

User Written Transformation

Here is what you can do with the User Written transformation:

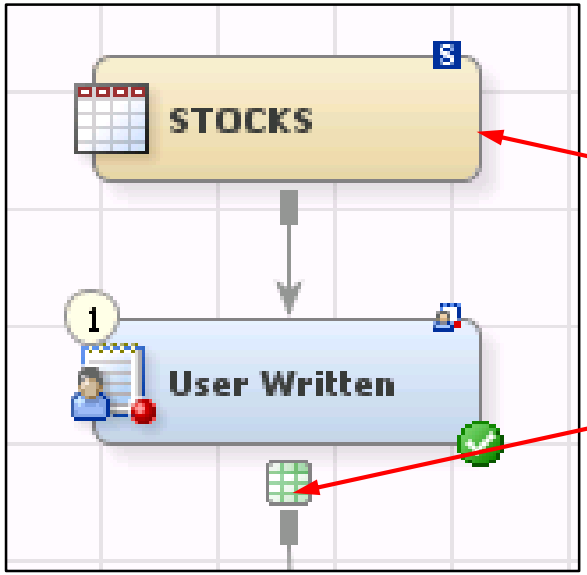
- add your own SAS code to a job flow
- reference one or more source table with &_input(n)
- reference one or more target tables with &_output(n)
- leverage existing code within a job
- add a customization that is not available in the interface

User Written Transformation




```
proc sort data = &_input1
        out = sortsubjinfo(keep = sdyid subjid usubjid     tobacco
alcohol);
     by sdyid usubjid;
run;
data &_output1;
     set sortsubjinfo;
     measure = 'Tobacco';
     value = tobacco;
     output;
     measure = 'Alcohol';
     value = alcohol;
     output;
     drop tobacco alcohol;
run;
```

User Written Transformation

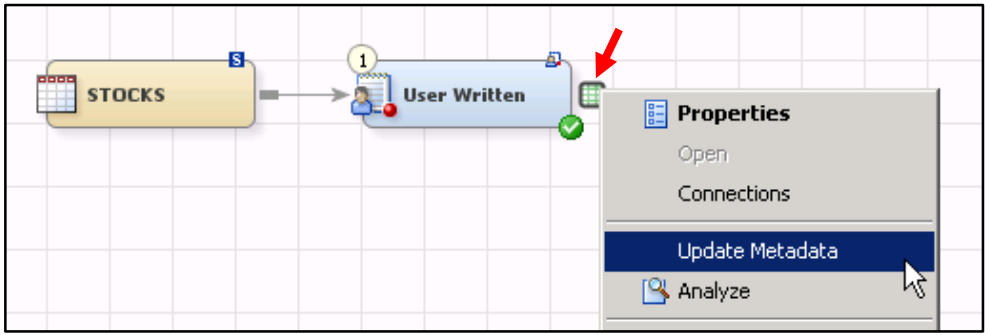



```
%let n = 6;
%let statvar=adjClose;
%let groupvar=stock;
%let datevar=date;
%let source=&_input1;

proc sort data=&source out=work.sourcesorted;
  by &groupvar &datevar;
run;

data &_output1;
  set work.sourcesorted;
    by &groupvar;
  format _numeric_ ;
  format &datevar date9.;
  drop i;
  array numlist_{&n} _temporary_ ;
  if first.&groupvar then do;
    count=0;
          do i=1 to &n;
          …
   …
run;
```

Update Metadata

The easiest method to define the column metadata for the target table for user-written code or custom transformations is often the following:

1. Execute the transformation to create the physical target table.
2. Use **Update Metadata** for the physical target table.



With **Update Metadata**, the structure of the ***physical*** table is read by SAS and is used to define the table metadata within the SAS Data Integration Studio job.

User-Defined Transformations

The New Transformation Wizard can be used to convert user-written SAS code into custom transformations that work like any other transformation.

- drag and drop from the Transformation tab.
- connect desired source tables and target tables.
- assign source or target columns (or both) to roles in transformation properties.
- specify other properties as required. User choices for these properties are assigned to macro variable values used in the code.

Implementing a SAS Program in a SAS Data Integration Studio Job

Key Steps

User Written Transformation

1. Specify macro variables for input/output data sets.
2. In a job, add the SAS program to the User Written transformation.
3. Connect the source and target tables to the User Written transformation.
4. Run the transformation.
5. Update the target table to match the physical table created by the User Written code.

Custom Transformation

1. Specify macro variables for the input/output data sets ***and*** for all other code elements that will be modifiable by transformation users via parameters.
2. Add SAS program to the custom transformation.
3. Define the parameters that will be available for users as transformation options.
4. In a job, add the custom transformation.
5. Connect the desired course and target tables to the custom transformation.
6. Define parameters as desired.
7. Run the transformation.
8. Update the target table to match the physical table created by the custom transformation.

User Written Code versus Custom Transformation

User Written	Custom
Job designer must add code to the transformation.	Job designer adds a transformation with existing code embedded in the transformation
Input and output data sets can be added through the visual interface.	Any options can be added via the pre-built parameter interface.
The best use is for one-time application of code in a specific job for a specific data source.	The best use is for general application of code utility for various data sources in multiple jobs.