ABSTRACT
SAS/Data Integration Studio (DI Studio) transformations are packaged tasks that give data architects the tools needed to construct ETL job flows. While DI Studio has built in transforms that satisfy the majority of an architect’s needs, there are occasions when the user must create a custom routine to meet a non-standard requirement. The DI Studio generated transformation is a perfect solution to fill this void when these non-standard requirements are repeatable.

DATA AND ENVIRONMENT
The following examples are based on data set SASHELP.STOCKS. This table contains stock prices for three companies with information which ranges from 1996 to 2005. SAS/Data Integration 4.21 and SAS 9.2 are used for this paper.

USER WRITTEN CODE TRANSFORMATIONS
The User Written Code transformation allows the data architect to run non-standard code. While other DI Studio transformations generate code for the user, the User Written Code transformation requires the end user to supply the majority of the logic. The DI Studio user should become familiar with this transformation before building their first Generated Transformation.

CONSTRUCTING THE USER WRITTEN CODE NODE
Drag the source table and the User Written code node onto the job canvas and connect the source table to the transformation.

Double click the User Written node, select the Mappings tab, and propagate the desired fields to the target table. All of the fields will already exist in the target if the job options which automatically propagate and map the job are active. These functions can be controlled in the job Settings by selecting the next to last icon at the top of the job.
Select the Code tab in the User Written Properties window. Confirm the Code generation mode is set to “User written body”. The user’s code is entered at the bottom of the window. As you scroll down, notice the macro variables which SAS/DAT Inte gration Studio creates. A few of the more prominent macro variables are:

_INPUT – the name of the input table(s)
_OUTPUT – the name of the output table(s)
_OUTPUT_keep – the list of fields which were defined in the target table via the Mapping tab

The next step is to enter the code. The following example sorts the STOCKS data set and uses first dot notation to select the date with the highest closing price for each company.
The result set can be loaded into a target table or can be processed with other transformations.

CONSTRUCTING A GENERATED TRANSFORMATION
Once a task is needed in multiple locations and therefore becomes repeatable, consider creating a Generated Transformation. The first step is to create a folder which can be used to store the transforms. Once a new folder is in place, right click on the folder to begin the process.
Give the transformation a name and confirm the transformation is stored in the desired metadata folder. Select a Transformation Category that best aligns with the function of the Generated Transformation so the transformation is easily found when constructing a job.

Enter the code that the transformation will execute. As with the User Written Code node, macro variables should be utilized. This example processes a sorted table and captures the data that meets the first dot or last dot if condition. Any records which do not meet the criteria are stored in an exception table. Notice the macro variables for the input and output tables, the by variable list, and the if statement.
After defining the source code, create prompts to link the designated macro variables which are not related to input and output tables. Defining these interfaces gives the user a way to enter values for these macro variables when building each DI Studio job.

The following prompt interfaces with the by_list macro variable.
The prompt which passes the list of by variables to the source code should be defined with a Prompt Type of Data Source Column. This allows the value to be selected by viewing the list of input source fields. Specify the option Select from source and deselect the boxes which Limit the number of selectable columns.
The second parameter in our example, the if_statement_value, is created in a similar fashion. The Prompt Type is Text and the method for populating the prompt is User enters values.

This example reads one input table and stores output in two output tables. The Transform Properties window is used to control the input and output properties of the transformation.

![New Transformation](image)

**Transform properties**

Specify additional properties for this transform.

**Inputs**

- Transform supports inputs
- Minimum number of inputs: 1
- Maximum number of inputs: 1

**Outputs**

- Transform supports outputs
- Minimum number of outputs: 2
- Maximum number of outputs: 2
- Automatically generate delete code for outputs
- Generate column mapping macros

**USING A GENERATED TRANSFORMATION**

Once the Generated Transform is complete, it can be used in a DI Studio job. Start with a blank canvas and add the source table holding stock information, a Sort transform, and the Capture First or Last Dot w Exceptions Generated Transformation.

Connecting the resulting table from the Sort transformation to the Generated Transform defines the input table. The transformation automatically creates two temporary tables as output tables. These output tables can flow through to additional processes or can be loaded directly into target tables.
Map the columns from the source to the target tables. Notice the columns are mapped twice since the Generated Transformation sends output to a primary target as well as an exception table.
To define the remaining macro variables, use the Options tab to specify the by variables and the if statement expression.

Once the transformation is complete and the job is submitted, the primary output table will contain the dates of the highest closing price.

The exceptions, which in this case include the remaining records, are available in the second table.
CONCLUSION
A Generated Transformation is a great way to surface non-standard code blocks that are used on a regular basis, in a repeatable fashion, to DI Studio. Becoming familiar with the User Written Code node is a great first step as it exposes the user to the link between macros variables and the required code. Once a task is needed in multiple locations, consider a Generated Transformation to easily surface this repeatable business logic.

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

Name: Jeff Dyson  
Enterprise: Financial Risk Group  
Address: 1251 NW Maynard Road STE 305  
City, State ZIP: Cary, NC 27513  
Work Phone: 919-449-6746  
E-mail: jeff.dyson@frgrisk.com  
Web: http://www.frgrisk.com/

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies.