Paper 092-2009

For Base SAS® Users: Welcome to SAS® Data Integration!

Jeff Stander, SAS Institute Inc., Lake Mary, Florida

ABSTRACT

Base SAS® offers a very powerful way to manage your data. This paper introduces using SAS® Data Integration Studio from a Base SAS programmer's perspective. We will examine what benefits SAS Data Integration Studio offers and how to easily start using SAS Data Integration Studio through the use of existing SAS code, the SAS Data Integration Studio GUI, and the SAS Code Importer.

INTRODUCTION

This paper consists of two sections. The first section examines some of the reasons why Base SAS programmers should consider using SAS Data Integration for their data integration projects. The second section provides a roadmap of the things you need to know to get started using SAS Data Integration and SAS Data Integration Studio.

BASE SAS

Base SAS gives SAS users tremendous flexibility when doing data management. For years, it has proven extraordinarily capable. This has resulted in a large number of successful projects in many organizations and it continues providing value today.

TRENDS AND CHALLENGES

Certain business trends, as well as new initiatives in our data environments, are driving change in the way we manage and process data. We see the following trends:

- Demand for transparency—Business owners want assurance that data is correct, and want to know their information processes are well documented and reliable. Code alone is not complete documentation.
- More data sources—The number of data sources available is rapidly increasing, the sizes of the data are growing rapidly, and we are now being asked to tap into more types of data including semi-structured and unstructured data types.
- Movement toward real time—We have less time in some cases for our data integration processes to run. This drives the need for more direct connectivity to systems and for capabilities such as changed data capture (CDC).
- Collaboration—We need to integrate with other developers and projects across the organization, work with Software as a Service (Saas) applications and enterprise resource planning (ERP) applications, interchange data with partners, and collaborate with outsourcing or offshore resources. All of these are driving the need for a better way to share work efforts.

CHANGE IS COMING

Now is the right time for some organizations to revisit their current way of working, which often relies solely on Base SAS programs, written by only a few individuals. Your organization should consider deploying the new techniques available today to meet these challenges.

CONSIDERATIONS FOR THE SAS PROGRAMMER

What is your situation?

You might be a seasoned SAS programmer with many years of experience. Imagine you manage much of your code in your own personal environment. Is some of your work too good to keep to yourself? Is there, say, 20% of your code that needs to be managed more closely? Is this, perhaps, being done by another person who knows your code less than you do?

How can you take your best code, along with your best practices in SAS, and apply them to new areas? Perhaps in your organization the IT landscape has hurdles to jump over to gain access to certain resources. How do you get certified access to those resources?

Do you have data integration scenarios where you have no common key to use to match records? Sometimes, coding by hand is not good enough, and your jobs could benefit from using fuzzy matches based on advanced data matching and data quality techniques.

Do you have any people who are occasional SAS developers who really don't spend 100% of their time doing SAS? How do you support them and keep their productivity high?

CONSIDERATIONS FOR THE PROJECT STAKEHOLDER OR MANAGER

What is your situation?

Maybe you feel you have a good handle on your data integration environment and all is well. What happens if a developer moves on to another project? How do you bring a new person up to speed? What skills do they require?

You rely on your SAS developers working together to achieve a result. These people might very well have deep expertise in a given area such as data access, DATA step or SQL for data integration, data quality techniques, complex business logic, and more. SAS Data Integration helps you leverage the skills on the team and manage the environment to achieve desired results, without having to be an expert on everything. You can also capture best practices in your group, boost productivity, and meet project deadlines by reusing current processes and previous investments.

Maybe you manage a data mart or data warehouse that is in production. Perhaps you inherited it when you took your new position in the organization. Do you have a good understanding of what is in the legacy SAS code?

Maybe you rely on an outside consultant to do some custom development. What happens when you outsource some new SAS work to a consultant? How do you preserve your investment? How do you get the most out of it? For example, if you hire someone to do some work for you, and you'd like to take the result and apply it to some other data sources, maybe with a slightly different data structure, how easily can you reuse the work, and map it to new sources of data in the enterprise?

WHAT SAS ENTERPRISE DATA INTEGRATION BRINGS TO BASE SAS PROGRAMMERS

If you have been successful doing data management using any combination of Base SAS, SQL, flat files, SAS/ACCESS® engines, and SAS/CONNECT®, why should you change the way you operate? What can you do in SAS Data Integration Studio that can't be done already in Base SAS?

To answer that question, let's see what the SAS® Enterprise Data Integration Server brings. It has additional data integration capabilities compared to Base SAS programming alone, and can help in several major areas including the following:

- Management
- Performance
- Development productivity
- Documentation
- Integration with the enterprise

Management
- Single point of control
- Managed environment
- Metadata-driven development
- Visual diagrams
- Documentation
- Centralized metadata repository
- Check-in/Check-out at a metadata object level
- User comment history changes made to objects checked-in
- Role-based authorization security

Performance
- Optimized code for multiprocessing and grid utilization
- SQL optimized for the database or data source
- Extract, load, and transform (ELT) support

- Display of performance statistics from SAS logs and visualization of performance metrics

Development Productivity

- Easy to understand help as you are building jobs, which reminds you what is missing
- Detects and finds errors without searching through logs
- Recommendations on how to correct errors you get
- Mapping interface accelerates mapping of data, which is much faster than coding it by hand
- Expression builder makes it easy to create derived columns
- Accurate fuzzy logic matching and standardization of unlike data
- Reuse of your code, which makes the environment work for you
- Copy and paste of existing objects
- Automated code generation for routine tasks
- Automated metadata management
- Visual debugging, step through job, view intermediate results, skip over steps
- Replace tables at source or target with another
- Point-and-click data mapping and data transformation
- SQL query designer saves time
- Automated documentation of data integration environment
- Includes SAS® Enterprise Miner™ analytic model results in your data integration jobs
- Data profiling tool
- Change data capture transformations
- Data quality services

Documentation

- Actively maintained metadata
- Job and table reports
- Data profiling
- Metadata discovery
- Difference reports on changing metadata and data sources

Integration with the Enterprise

- Supports collaboration with people working across the organization (multi-user)
- Import and export of metadata with various repositories
- Capable of accessing ERP and SaaS applications and other enterprise infrastructure

As you can see, SAS Enterprise Data Integration Server brings a lot of useful capabilities over and beyond Base SAS. Here are a few points to note:

- Using SAS Data Integration Studio doesn't mean you cannot code in Base SAS.
- You still have power and flexibility of code.
- SAS Data Integration allows you to automate routine tasks.
- You can extend Base SAS functionality.
- You can do more in less time.

OVERVIEW OF ARCHITECTURE

If you are doing data management using Base SAS, your investment in SAS programs will continue to be protected and preserved after upgrading to a SAS Data Integration solution. Because Base SAS is the foundation of SAS Enterprise Data Integration, you maintain the capability to run your SAS programs.

It's important to know what is included with SAS Enterprise Data Integration. Many of the capabilities SAS programmers are already familiar with are included with the SAS Enterprise Data Integration Server. Here is a partial list:

- Base SAS—commonly used via the Program Editor, the Base SAS language includes PROC SQL and the DATA step
- SAS/ACCESS—for getting access to data stored outside of SAS (for example, in relational databases)
- SAS/CONNECT—for connecting to remote SAS servers and partitioning SAS processing across SAS sessions

What you might not be familiar with are some of the additional components that SAS Enterprise Data Integration provides:

- SAS Data Integration Studio—a common development environment for designing and testing your data integration jobs and processes.
- SAS® Metadata Server—provides metadata services to support development of your data integration processes.
- SAS Workspace Server—execution environment for SAS Data Integration jobs (think of this as your traditional Base SAS environment as a server you can submit work to).
- SAS Metadata Bridges—engines that read and write metadata from various formats into something you can put to use.
- DataFlux dfPower Profile—profiles data sources for a complete data assessment (new in SAS 9.2 Data Integration Server).
- DataFlux dfPower Explorer—profiles metadata, helps identify relationships between tables, and more (new in SAS 9.2 Enterprise Data Integration Server).
- DataFlux Enterprise Integration Server for SAS—DataFlux data quality services server environment for use with SAS. This server is delivered with the SAS Enterprise Data Integration Server starting in 9.2 and works with new data quality transformations in SAS Data Integration Studio.

ROADMAP FOR GETTING STARTED

Below are some steps you can take to prepare to use SAS Data Integration. First, you should examine your environment and decide which SAS programs you'd like to bring into SAS Data Integration Studio.

Create an inventory of all things SAS, including the following items:

- SAS programs
- SAS libraries
- SAS data sets
- SAS formats
- SAS macros
- Project documentation, such as .doc, .ppt, .xls, .ddl, and so on

Also, take an inventory of data sources available to you today, including those you might not be using.

- Relational databases, such as Oracle, DB2,Teradata, and SQL Server
- Files, such as .csv, .xls, .txt, and .xml
- Message queues, such as MQ Series and MSMQ

Having this information available helps you as you get started using SAS Data Integration Studio.

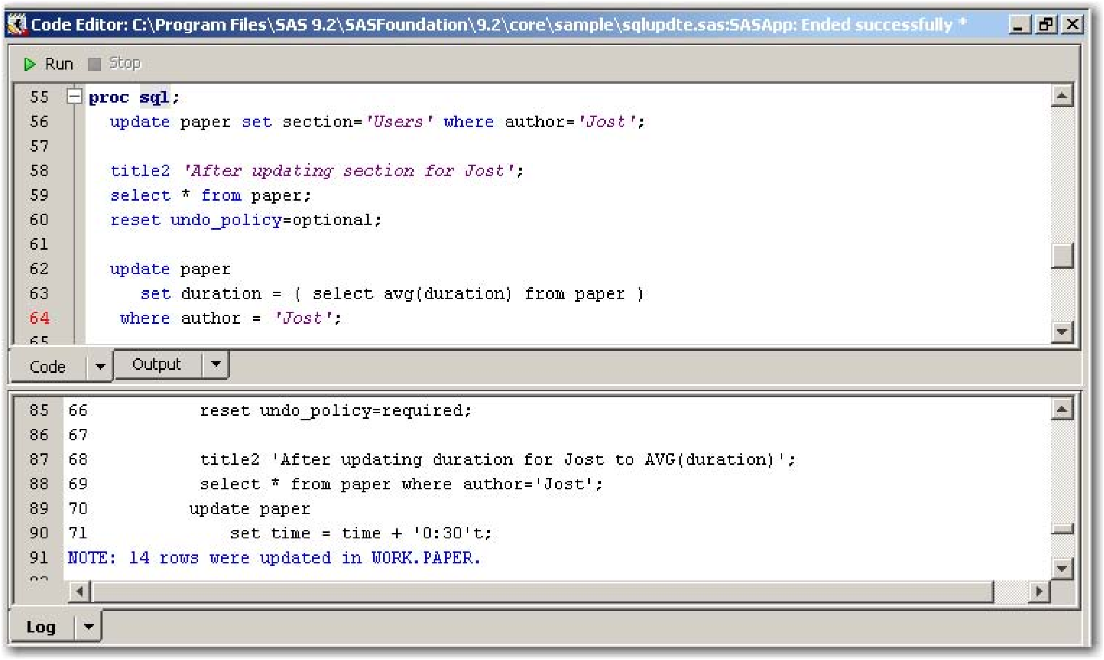
You can keep your existing SAS programs. They can continue to run as-is using Base SAS. You can develop new data integration processes using SAS Data Integration Studio. Over time, you can choose to gradually bring your existing SAS programs into the more managed environment that this application provides.

GETTING STARTED FROM A BASE SAS USER'S PERSPECTIVE

In the second half of the paper, we walk through some examples. We start with our Base SAS environment and bring existing code into SAS Data Integration Studio. You can do all or some of these steps. We look at the range of activities you can follow to use your existing SAS projects in SAS Data Integration Studio. We show some of the tools available to you.

CODE EDITOR

SAS Data Integration Studio is a development environment for designing and testing your data integration jobs. From the **Tools** menu, there is a code editor, along with a Log and Output window available for your use. The Code Editor is shown in Figure 1.


Code Editor: C:\Program Files\SAS 9.2\SASFoundation\9.2\core\sample\sqlupdte.sas:SASApp: Ended successfully *
Run
Stop
Code
Output
Log

```
proc sql;
  update paper set section='Users' where author='Jost';

  title2 'After updating section for Jost';
  select * from paper;
  reset undo_policy=optional;

  update paper
     set duration = ( select avg(duration) from paper )
   where author = 'Jost';
```

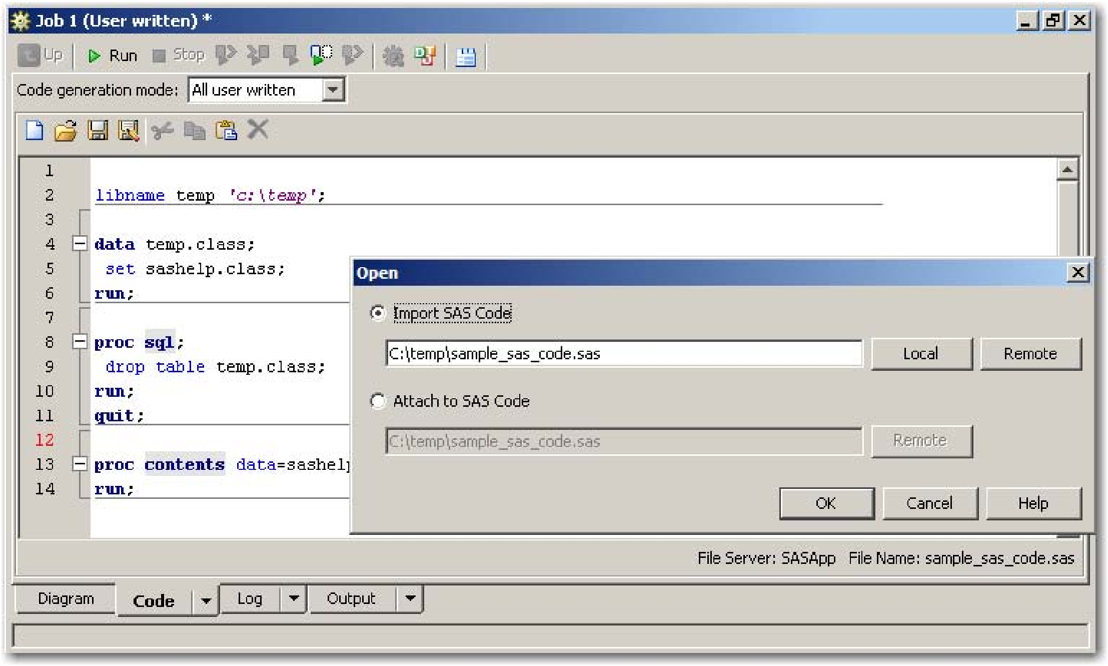
```
66           reset undo_policy=required;
67
68           title2 'After updating duration for Jost to AVG(duration)';
69           select * from paper where author='Jost';
70          update paper
71              set time = time + '0:30't;
NOTE: 14 rows were updated in WORK.PAPER.
```

Figure 1. The Code Editor in SAS Data Integration Studio

Here you can open existing SAS code and make edits to it, and you can write new SAS code. You can submit your code to any SAS Workspace Server, and view the Log and Output windows afterward.

CREATE A SAS DATA INTEGRATION STUDIO JOB USING ONLY EXISTING CODE

You can create a job by simply using existing code. This results in your SAS code having complete control over the job. You can copy and paste or directly enter your SAS code into the code window for the job. Alternatively, you can read in a file containing SAS statements. You can link or attach to a file containing SAS code. In this case, if you change the file later, the changes are reflected in the code in the job the next time you open it. Note that **Code generation mode** is set to **All user written** in Figure 2.



Job 1 (User written) *

Code generation mode: All user written

```
libname temp 'c:\temp';

data temp.class;
 set sashelp.class;
run;

proc sql;
 drop table temp.class;
run;
quit;

proc contents data=sashel
run;
```

Open

(●) Import SAS Code

C:\temp\sample_sas_code.sas

( ) Attach to SAS Code

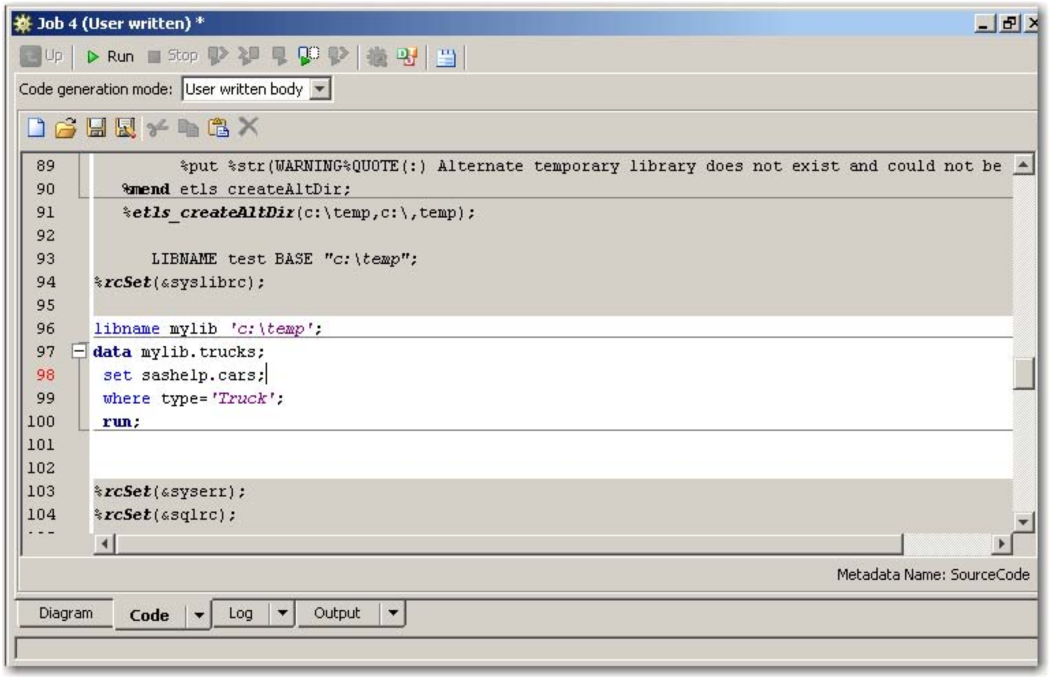
C:\temp\sample_sas_code.sas

File Server: SASApp File Name: sample_sas_code.sas

Figure 2. A Job Using All User-Written Code

CREATE A SAS DATA INTEGRATION STUDIO JOB USING A USER-WRITTEN BODY (AND AUTOGENERATED CODE)

If you want, you can tell SAS Data Integration Studio to generate some wrapper code around your SAS program. To do this, you switch **Code generation mode** to **User written body**. (See Figure 3.) The wrapper code is in gray around the body code. The benefit here is you can provide the main part of the code and allow SAS Data Integration Studio to generate code for things such as status handling and error controls.



Job 4 (User written) *

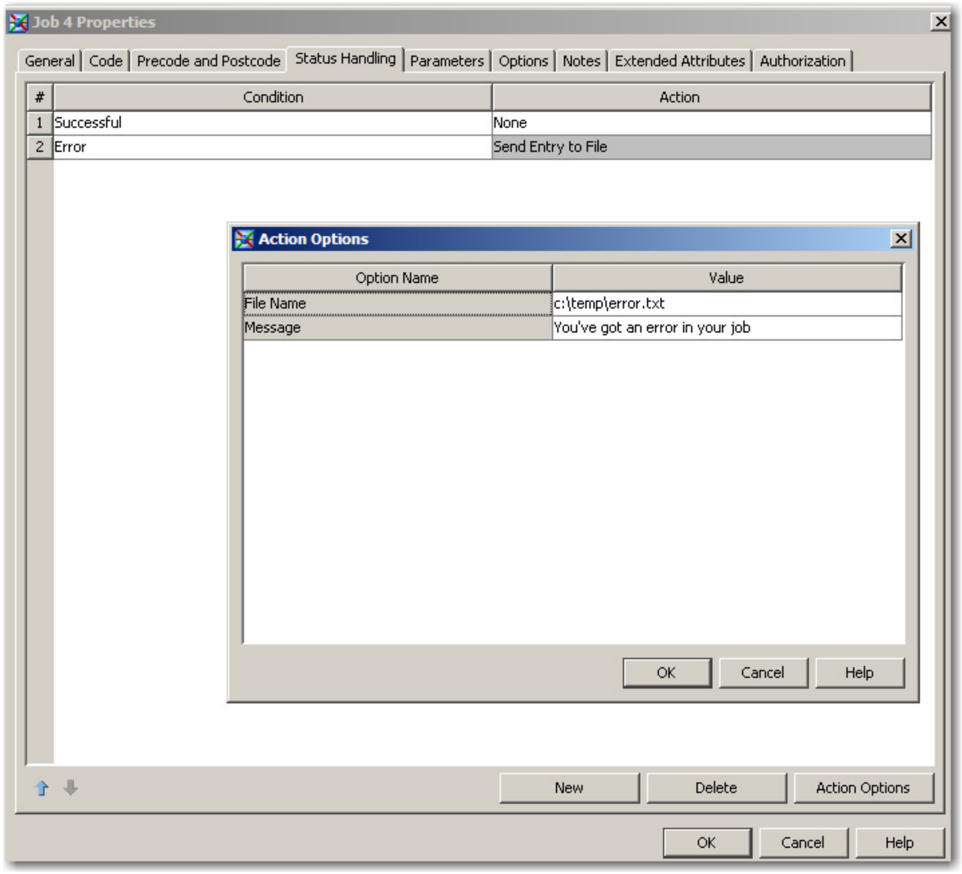
Code generation mode: User written body

```
            %put %str(WARNING%QUOTE(:) Alternate temporary library does not exist and could not be
   %mend etls createAltDir;
   %etls_createAltDir(c:\temp,c:\,temp);

      LIBNAME test BASE "c:\temp";
%rcSet(&syslibrc);

libname mylib 'c:\temp';
data mylib.trucks;
 set sashelp.cars;
 where type='Truck';
 run;


%rcSet(&syserr);
%rcSet(&sqlrc);
```

Metadata Name: SourceCode

Figure 3. A Job With User-Written Body

Now that we have the wrapper code being generated by SAS Data Integration Studio, we can use automatic status handling by setting a few metadata properties for the job. In this case, a record is written to a file if there is an error. This is one of many capabilities that we can ask the application to do for us. Figure 4 shows the metadata attributes of job status handling.



#	Condition	Action
1	Successful	None
2	Error	Send Entry to File

Option Name	Value
File Name	c:\temp\error.txt
Message	You've got an error in your job

Figure 4. A Job Properties Sheet Showing Settings for Automatic Status Handling

CREATE A SAS DATA INTEGRATION STUDIO JOB WITH A USER-WRITTEN CODE NODE

Another way to use existing SAS code is to add a data transformation called **User Written Code** into a job. (See Figure 5.) This transformation allows you to insert any code and treat it as an object in a job. The code can be in a file or in metadata. You can register metadata attributes to document further what the code does. User-written transformations can be inserted anywhere in a SAS Data Integration Studio job. This allows us to use some of the standard transformations in SAS Data Integration Studio with our own code. Having your code in a node allows you to reorder the sequence in which it runs by using the **Control Flow** control.

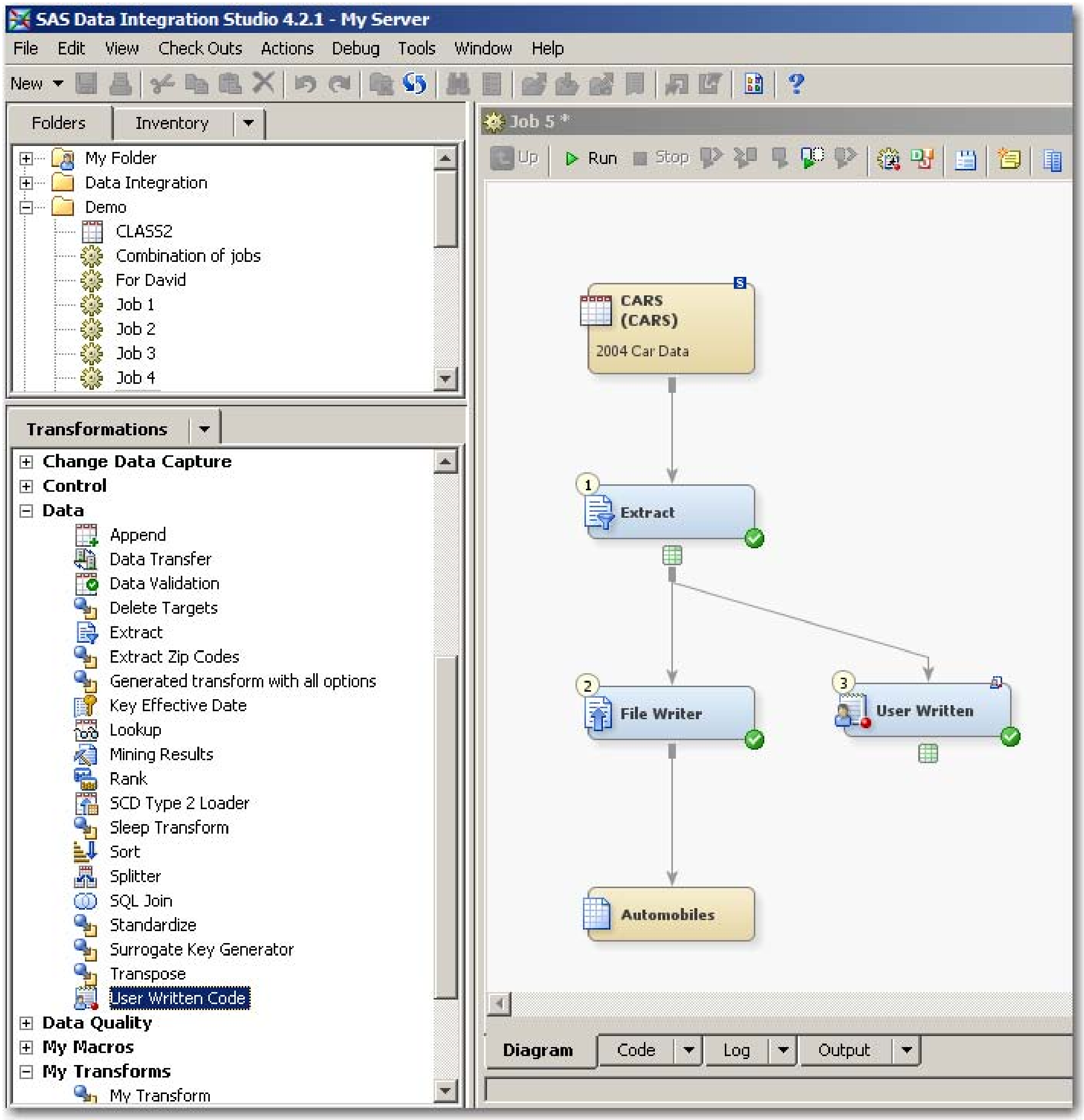
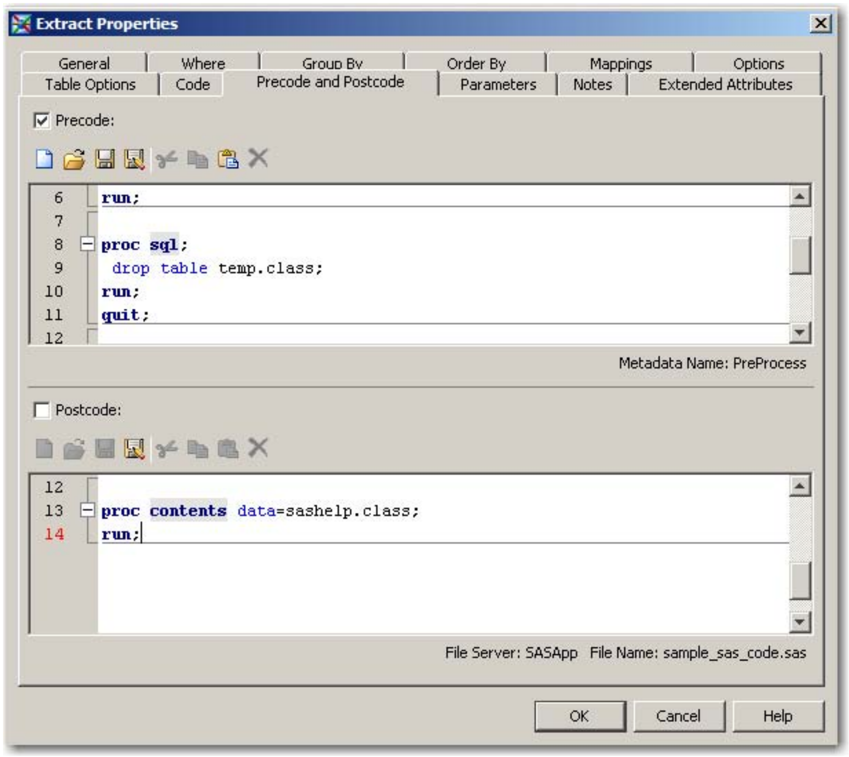


Figure 5. User Written Code Transformation Inserted in a Job

USING PRECODE AND POSTCODE

Precode and postcode can be defined for jobs and for most transformations. This is another place where you can insert code to perform standard tasks to set up your environment, to restore an environment, or to backup the results of a particular step. This is also a useful place to insert code for debugging purposes. Precode and postcode can be toggled on and off when in the development environment. (See Figure 6.)



☑ Precode:

```
run;

proc sql;
 drop table temp.class;
run;
quit;
```

Metadata Name: PreProcess

☐ Postcode:

```

proc contents data=sashelp.class;
run;
```

File Server: SASApp File Name: sample_sas_code.sas

Figure 6. Example of Precode and Postcode Being Used in the Extract Transformation

REPLACE AUTOMATICALLY GENERATED CODE WITH YOUR OWN

Every transformation has a Code tab that allows you to override the automatically generated code. This allows you to insert code, perhaps already existing code, to accomplish the task. You can also take the automatically generated code and modify it after saving a copy of it.

REUSE YOUR CODE BY CREATING YOUR OWN TRANSFORMATIONS

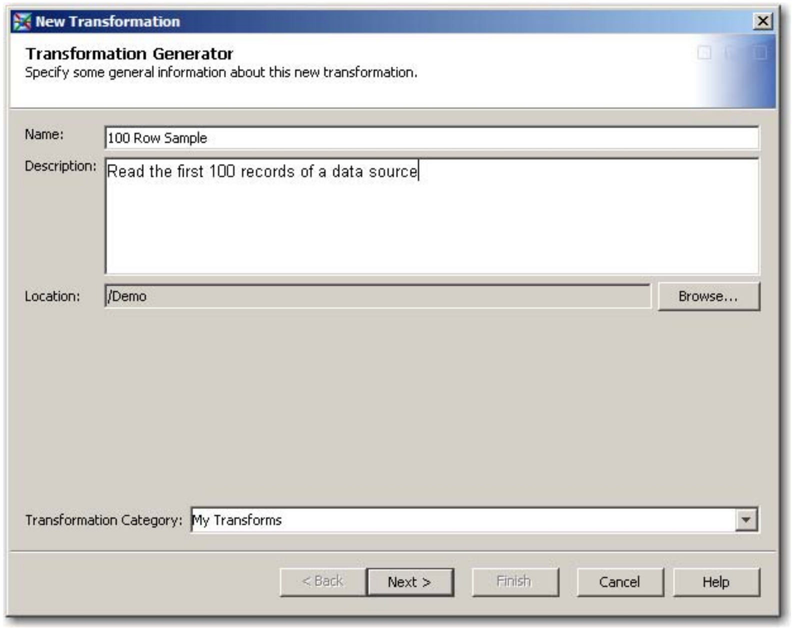
SAS Data Integration Studio provides a standard set of transformations. You might have code that you find extremely useful that you want to use in multiple places and perhaps share with other developers. You might have a standard set of SAS macros you've created that you like to use. Here are some ways to use these items within SAS Data Integration Studio:

- Embed macro code in an individual job
- Encapsulate your code in a user-generated SAS Data Integration transformation
- Store your macros in an autocall macro library for global usage

Making a transformation to hold your code allows you to do neat things such as find the programs that use it by running an impact analysis report in SAS Data Integration Studio. Let's look at making a simple user-generated transformation to sample a data table. We are going to read the first 100 records of a table and create an output table. We will let the user select which columns to keep. Here is the SAS code we will use:

```
data &_OUTPUT0 (keep=&columnswanted);
  set &syslast;
  if _n_ < 101;
 run;
```

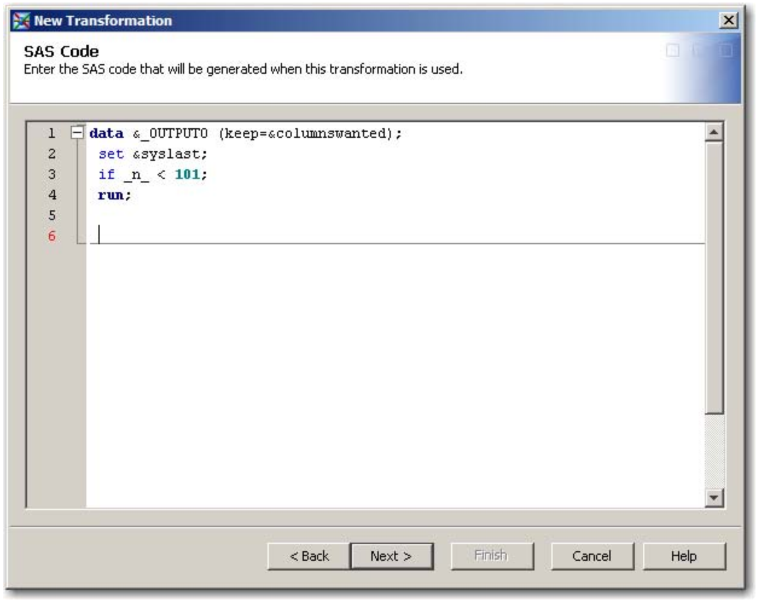
The Transformation Generator wizard steps you through, adding a new transformation using your code. Figures 7 through 10 illustrate using the Transformation Generator to create a new transformation.


New Transformation

Transformation Generator
Specify some general information about this new transformation.

Name: 100 Row Sample
Description: Read the first 100 records of a data source
Location: /Demo
Transformation Category: My Transforms

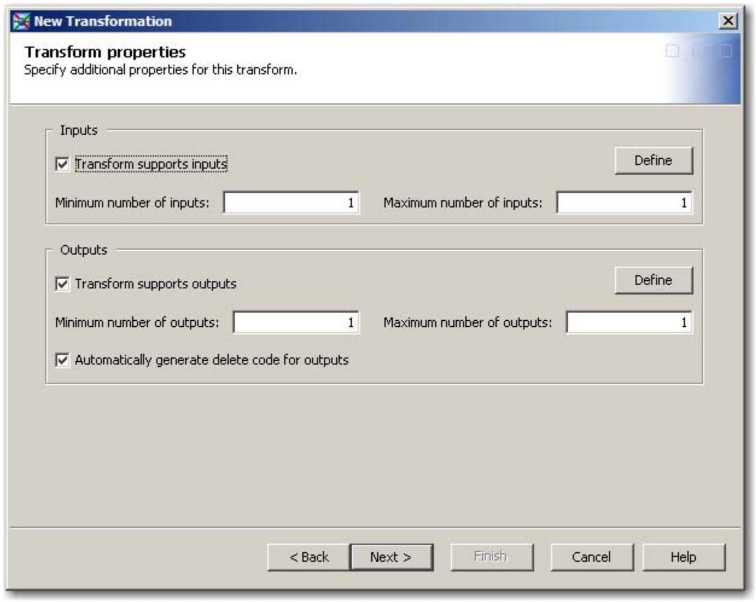
Figure 7. Transformation Generator: Name the Transformation


New Transformation

SAS Code
Enter the SAS code that will be generated when this transformation is used.

```
data &_OUTPUT0 (keep=&columnswanted);
 set &syslast;
 if _n_ < 101;
 run;
```

Figure 8. Transformation Generator: Paste In Your SAS Code



☑ Transform supports inputs

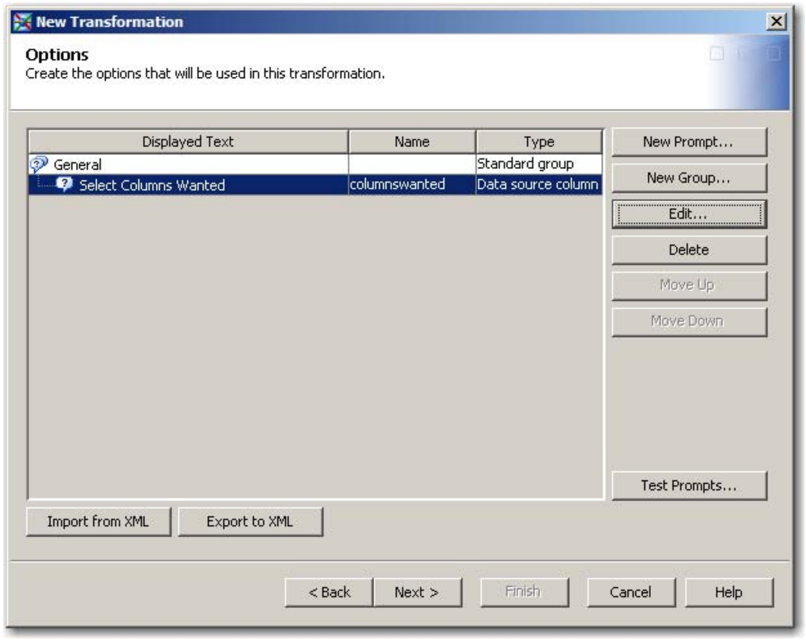
Minimum number of inputs: 1 Maximum number of inputs: 1

☑ Transform supports outputs

Minimum number of outputs: 1 Maximum number of outputs: 1

☑ Automatically generate delete code for outputs

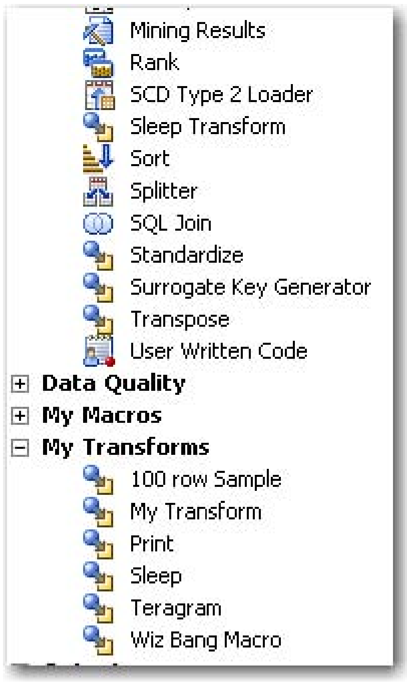
Figure 9. Transformation Generator: Specify If There Are Inputs and Outputs



Displayed Text	Name	Type
General		Standard group
Select Columns Wanted	columnswanted	Data source column

Figure 10. Transformation Generator: Create the Options That Control Your Code (these will be surfaced as metadata attributes or prompts)

After completing the wizard, we now have an additional transformation in the My Transforms folder. (See Figure 11.) This new transformation can be used an unlimited number of times across multiple jobs. It is reusable in many scenarios because we provide it options that can be set at the job level in metadata. To get an impact analysis report showing all the SAS Data Integration Studio jobs that are using our new transformation, we simply click on the transformation and select **Analyze**.



6767

Figure 11. SAS Data Integration Studio Transformation Library (with our own 100 row Sample transformation)

After a period of time, you might want to change the way the transformation works. You can edit the code once and changes will be available to all jobs that use the transformation. The next time you open a job, it will refresh the code with the changes you made to it.

CREATE A SAS DATA INTEGRATION STUDIO JOB USING IMPORT SAS CODE (SAS CODE ANALYZER)

Available in SAS Data Integration Studio 4.21 is an enhanced capability called **Import SAS Code**. This works with the SAS® Code Analyzer (PROC SCAPROC) in SAS 9.2 to analyze your code and to automatically create SAS Data Integration Studio jobs. It provides additional analysis and documentation of your SAS code, which enables SAS Data Integration Studio to do more with your code.

Using this capability, you select a SAS program, and SAS Data Integration Studio creates a job automatically. Behind the scenes, it calls the SAS Code Analyzer procedure to analyze your SAS program. The SAS Code Analyzer captures information about input, output, and the use of macro symbols from a SAS job while it is running. The output generated is a file with your SAS program and additional comments.

When importing a SAS program, you can specify whether you have already run your program with the SAS Code Analyzer or not. This is useful for programs that might not be on the local machine, or might be currently unavailable to execute due to your environment. Figures 12 and 13 show the use of **Import SAS Code**.

Here is example code that is submitted to call the SAS Code Analyzer:

```
proc scaproc;
  Record 'program1.sca' attr;
run;

/* insert some sas code here */
data work.females;
  set sashelp.class;
  where sex='F';
 run;

proc scaproc;
  write;
run;
```

Here is the output of the procedure—a file called program1.sca:

```
/* JOBSPLIT: DATASET INPUT SEQ #C00001.CLASS.DATA */
/* JOBSPLIT: LIBNAME #C00001 V9 C:\Program Files\SAS 9.2\SASFoundation\9.2\core\sashelp */
/* JOBSPLIT: DATASET OUTPUT SEQ WORK.FEMALES.DATA */
/* JOBSPLIT: LIBNAME WORK V9 C:\DOCUME~1\sasjst\LOCALS~1\Temp\SAS Temporary Files\_TD6896 */
/* JOBSPLIT: FILE OUTPUT c:\temp\program1.sca */
/* JOBSPLIT: ATTR #C00001.CLASS.DATA INPUT VARIABLE:Name TYPE:CHARACTER LENGTH:8 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: ATTR #C00001.CLASS.DATA INPUT VARIABLE:Sex TYPE:CHARACTER LENGTH:1 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: ATTR #C00001.CLASS.DATA INPUT VARIABLE:Age TYPE:NUMERIC LENGTH:8 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: ATTR #C00001.CLASS.DATA INPUT VARIABLE:Height TYPE:NUMERIC LENGTH:8 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: ATTR #C00001.CLASS.DATA INPUT VARIABLE:Weight TYPE:NUMERIC LENGTH:8 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: ATTR WORK.FEMALES.DATA OUTPUT VARIABLE:Name TYPE:CHARACTER LENGTH:8 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: ATTR WORK.FEMALES.DATA OUTPUT VARIABLE:Sex TYPE:CHARACTER LENGTH:1 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: ATTR WORK.FEMALES.DATA OUTPUT VARIABLE:Age TYPE:NUMERIC LENGTH:8 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: ATTR WORK.FEMALES.DATA OUTPUT VARIABLE:Height TYPE:NUMERIC LENGTH:8 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: ATTR WORK.FEMALES.DATA OUTPUT VARIABLE:Weight TYPE:NUMERIC LENGTH:8 LABEL: FORMAT: INFORMAT: */
/* JOBSPLIT: SYMBOL GET SYS_IOUSEEE */
/* JOBSPLIT: ELAPSED 15  */
/* JOBSPLIT: PROCNAME DATASTEP */
/* JOBSPLIT: STEP SOURCE FOLLOWS */

data work.females;
  set sashelp.class;
  where sex='F';
 run;

/* JOBSPLIT: END */
```

Next, behind the scenes, a SAS Data Integration Studio job is automatically created using the program1.sca file from PROC SCAPROC. The job has a series of user-written code nodes, one for each DATA step or procedure in your SAS code.

At this point, you can do several things.

- Run the SAS Data Integration Studio job.
- View source and target table metadata.
- View impact analysis reports.
- View the SAS code .
- Replace tables in the job.
- Reorder transformations in the job.
- Add or insert new transformations.
- Modify existing jobs.

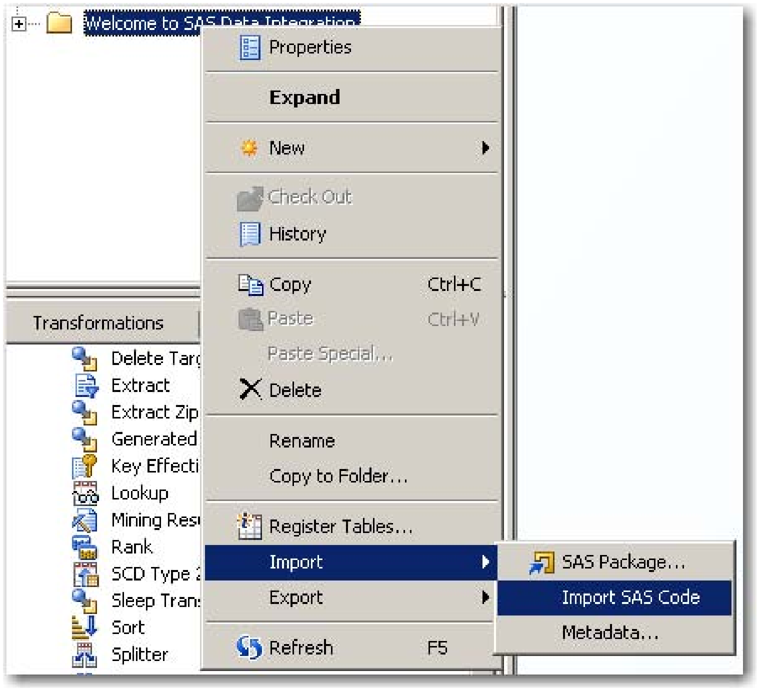
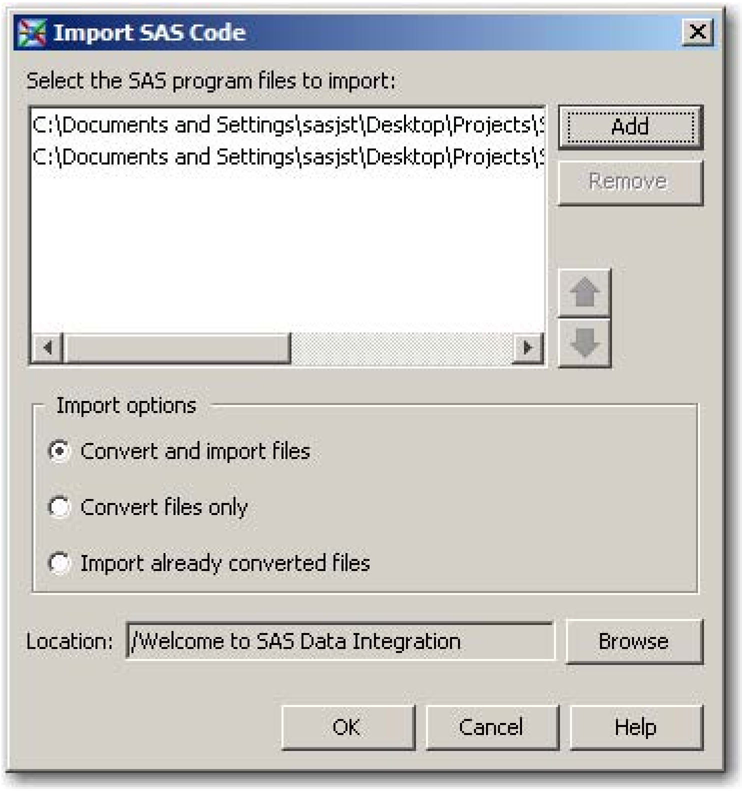


Figure 12. Importing SAS Code



Select the SAS program files to import:

C:\Documents and Settings\sasjst\Desktop\Projects\
C:\Documents and Settings\sasjst\Desktop\Projects\

Import options

[✓ Convert and import files]

Convert files only

Import already converted files

Location: /Welcome to SAS Data Integration

Figure 13. Selecting the Files to Import

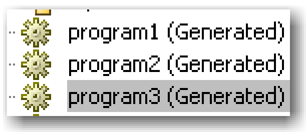


Figure 14. The Resulting Generated SAS Data Integration Jobs

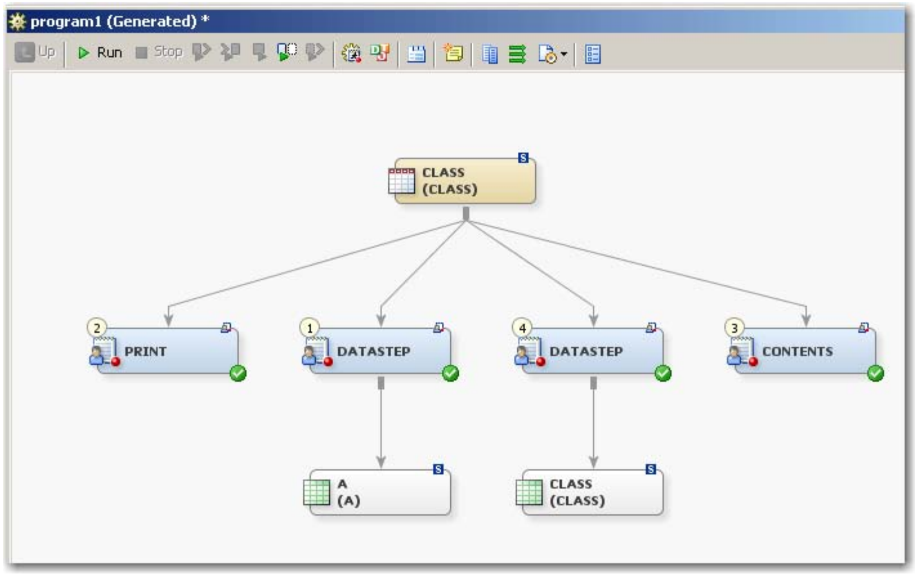


Figure 15. An Imported Job (the user-written code is broken into DATA steps and procedures, note the inputs and outputs)

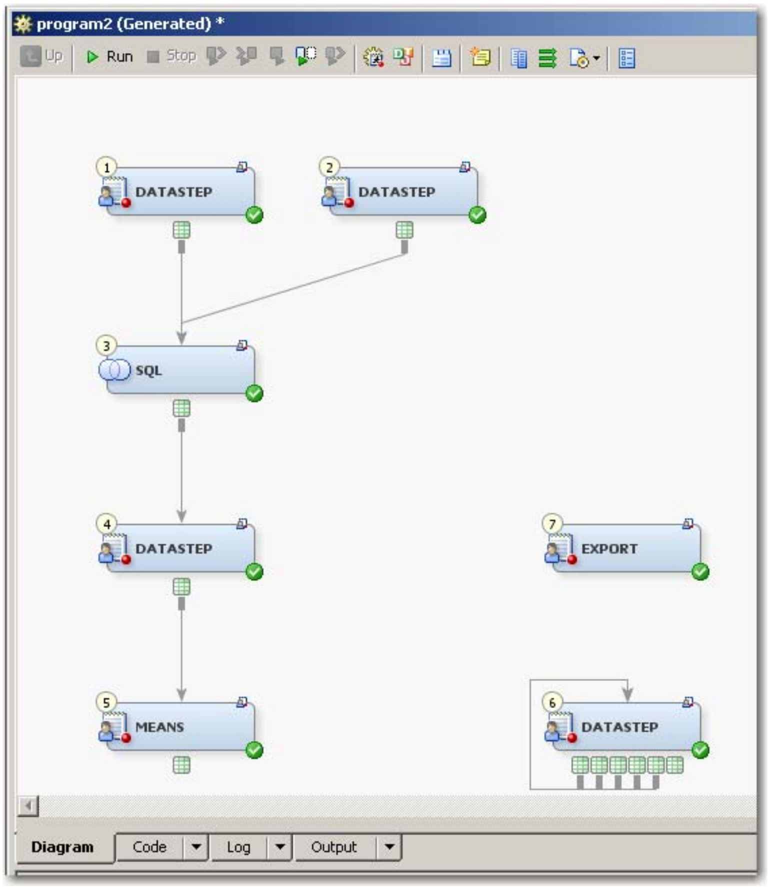


Figure 16. Another Imported Job (note the sixth step, which shows many data sets being created by SAS code generated by a SAS macro)

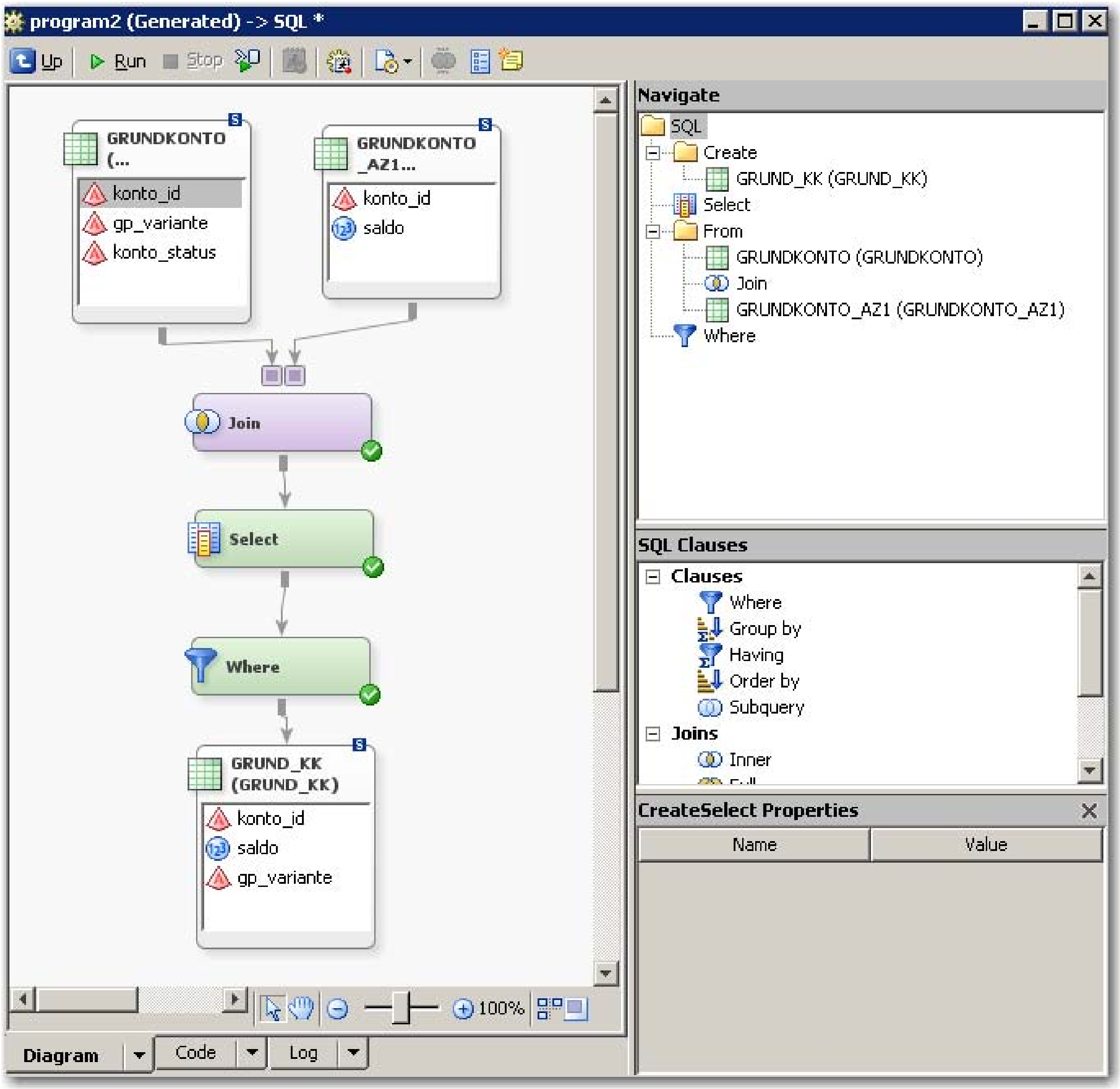
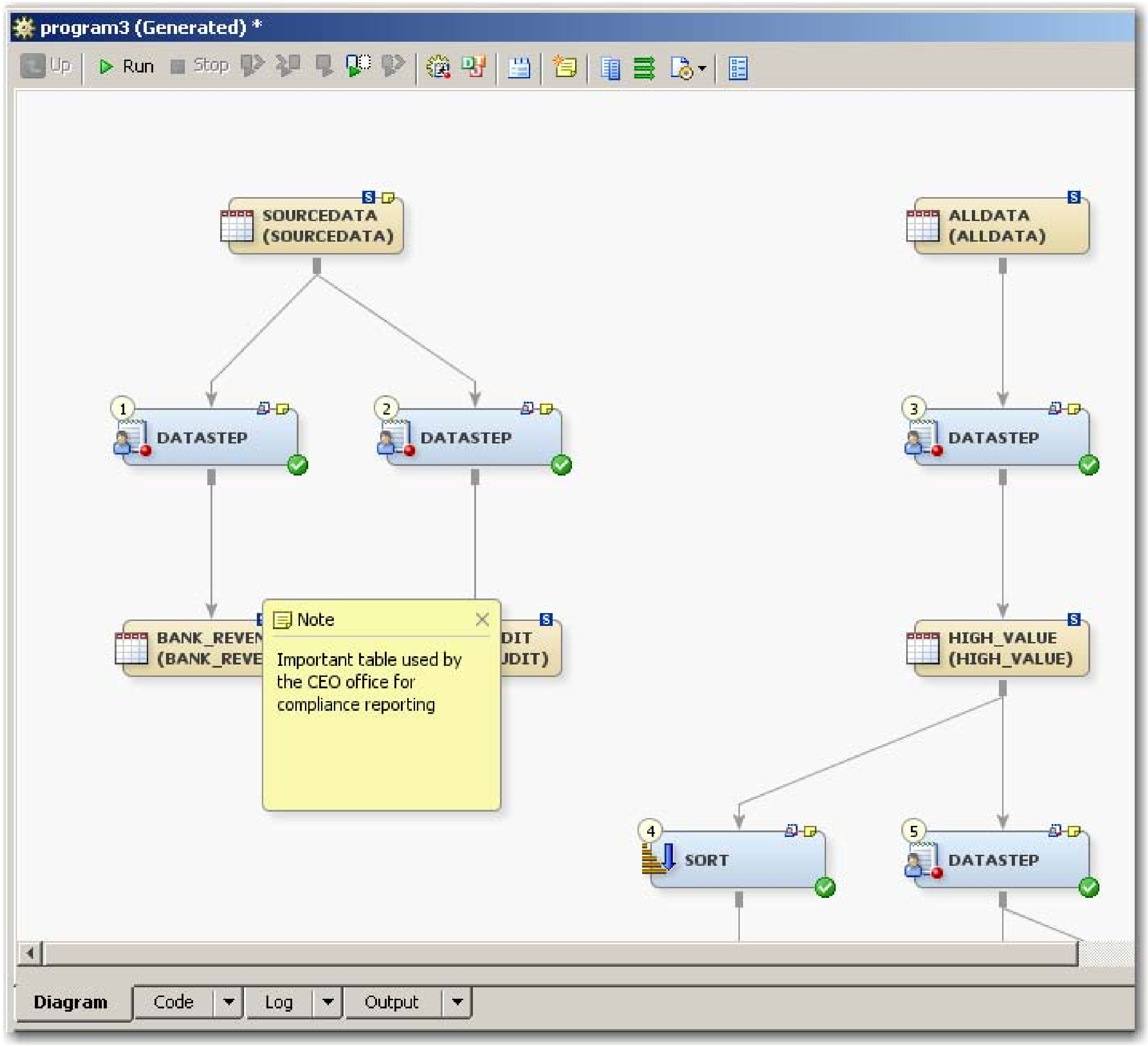


Figure 17. Properties for the SQL Join transform that was automatically generated in the previous job (see Figure 16) when importing Proc SQL code



Note

Important table used by the CEO office for compliance reporting

Figure 18. Another Imported Job (we can view existing metadata such as notes on the source and target data sources to understand more about how data is being used)

ARE THERE ANY LIMITATIONS TO WHAT IMPORT SAS CODE CAN DO?

This capability relies on the descriptive output file that PROC SCAPROC produces. This procedure analyzes the SAS source code as it runs. So, if you are using macros that generate SAS code, keep in mind that it might not capture every path a macro might take.

In the SAS Data Integration Studio mappings window, only one-to-one mappings are captured automatically. Derived mappings you might have (for example, in a SAS DATA step) can be added afterward in metadata.

Resulting nodes that are created in the job are for the most part using the user-written code transformation. If your code contains a Proc Sort or a Proc SQL, the Sort transform and the SQL Join transforms are used. The code linked to the transform will still be user defined, to preserve what you have written. If you want you to, you can adjust the metadata attributes for the transform, to modify its behavior, and to leverage the transform's full capabilities.

CONCLUSION

We have examined two topics in this paper. First, we looked at some of the challenges faced in doing data integration work and the increasing demands placed on those who are still substantially relying on coding by hand.

We looked at a few of the ways in which SAS Enterprise Data Integration Server helps support our data integration activities. There are many ways to get started using this technology. From a SAS programmer's perspective, we looked at several possible ways. Using existing Base SAS programs, we created jobs using SAS Data Integration Studio. Looking forward, we see many Base SAS programmers looking at SAS Enterprise Data Integration Server to better manage their data integration environment, and the data integration work they do for their organizations.

ACKNOWLEDGMENTS

The author expresses his appreciation to the following individuals who reviewed this paper: David Barkaway, Eric Hunley, Nancy Rausch, Ken Hausman, Julie Maddox, and Michael Herrman.

RECOMMENDED READING

SAS Institute Inc. 2008. "The SCAPROC Procedure." *Base SAS 9.2 Procedures Guide*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/proc/59565/HTML/default/a003199742.htm.

SAS Institute Inc. 2009. *SAS Data Integration Studio User's Guide*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/onlinedoc/etls/index.html.

CONTACT INFORMATION

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

Jeff Stander
SAS Institute Inc.
801 International Parkway, 5th Floor
Lake Mary, FL 32746
Work Phone: 1-407-562-1561
E-mail: Jeff.Stander@sas.com