Paper SAS1857-2015

Hands-Off SAS® Administration—Using Batch Tools to Make Your Life Easier

Eric Bourn, Amy Peters, and Bryan Wolfe, SAS Institute Inc., Cary, NC

ABSTRACT

As a SAS® Intelligence Platform Administrator, have your eyes ever glazed over as you performed repetitive tasks in SAS® Management Console or some other administrative user interface? Perhaps you're setting up metadata for a new department, managing a set of backups, or promoting content between dev, test, and prod environments. Did you know there is a large library of batch utilities to help you automate many of these common administration tasks? This paper explores content reporting and management utilities, such as viewing authorizations or relationships between content, as well as administrative tasks such as analyzing, creating, or deleting metadata repositories or performing a backup of the system. The batch utilities can be incorporated into scripts so that you can run them repeatedly on either an ad hoc or scheduled basis. Give your mouse a rest and save yourself some time.

INTRODUCTION

The SAS Intelligence Platform consists of a collection of command-line batch utilities that are designed to assist administrators perform a variety of functions in order to manage a SAS environment. Tasks such as content management and overall system administration of the SAS® Metadata Server can be implemented without any user interface interaction.

The utilities delivered within this framework are capable of executing on all hosts that are supported by the SAS Intelligence Platform, including Windows, UNIX, and z/OS environments. This framework is installed along with many other SAS applications, including SAS Management Console, SAS® Data Integration Studio, and SAS® OLAP Cube Studio.

Within a given SAS installation directory, these utilities can be found under:

```
!SASHOME\SASPlatformObjectFramework\9.4
```

A few of the original utilities created in previous SAS releases exist directly under this location.

Beginning in the SAS 9.4 release, however, a new naming convention, using a "sas-" prefix for each tool name, was created to help provide better organization. Some, although not all, were converted to this new syntax. Utilities that follow this naming convention can be found in these two subdirectories:

- `\tools`
  Contains utilities that are designed for general usage and typically do not require special access or privileges.

- `\tools\admin`
  Contains utilities that are intended more for administrative type functions.

While there are several utilities contained within this framework, each utility can be grouped into one of the following categories:

- Reporting – utilities that enable you to perform various reporting and searching functions against content that is managed within your SAS Folders hierarchy. These reporting utilities make it easier not only for you to find your data, but also in determining how objects are related to one another.
- Content management - utilities that manage your content in a generic fashion. This includes performing common actions on content objects, such as managing access controls and promoting content from one environment to another.

- System management - utilities that are used to perform administrative tasks against a SAS Metadata Server environment, including analyzing an existing SAS Metadata Server, creating and removing repositories, as well as performing backups of your SAS content across multiple machines. Utilities in this category are typically located within `\tools\admin`.

Each utility is designed such that it uses a common syntax for specifying input parameters, including environment details (server host and port information), user credentials, and logging instructions. For the sake of brevity, the examples in this paper do not display these common options. Instead, additional information about these options can be found under the section titled "Common Tool Options."

This paper does not describe every utility that is included within the SAS Intelligence Platform in detail. Rather, the following sections focus on how a handful of utilities can be used in concert with one another to perform a specific task - promoting content from a source environment to a target environment. Highlighted in this paper are various batch utilities that can be used to search for the proper content, set up your development environment, and promote the data all by using the command line. No mouse required.

STEP 1 - ANALYZING THE SOURCE ENVIRONMENT

Now that you have learned about the overall infrastructure and the various tools at your disposal, it's time to jump in and get started with the first step in our task; searching for the right set of content to migrate.

Depending on the complexity of an environment, it is conceivable to have hundreds, if not thousands, of diverse content objects spread out across a given SAS Folders hierarchy. Even if these folders are well organized, it still might be a challenge to find exactly what you are looking for. A new batch utility introduced in SAS 9.4, sas-list-objects, can be of an assistance.

The sas-list-objects utility enables you to search for any content object stored within the SAS Folders hierarchy using a wide array of filter criteria. This includes the ability to search for objects by their type, name, folder location, date information, and even associated responsibilities. This utility mimics the same features that are found within SAS Management Console's Search tab. The output from this utility can be returned in an easy-to-read text, comma-separated values (CSV), or standard XML format. By default, the text format is used.

In this particular use case, we need to find a collection of recently modified stored processes. The problem, however, is that it is unknown as to where these stored processes exist or what their exact names are. By submitting the following command, a search can be executed that will help determine which objects are needed.

```
$ sas-list-objects -types StoredProcess -folderTree "/Dev/Source" -modified
-since "Week to Date" -sort "path"

/Dev/Source/Stored Processes/cascadingCarData-package(StoredProcess)
/Dev/Source/Stored Processes/cascadingCarData-stream(StoredProcess)
/Dev/Source/Stored Processes/procFreq(StoredProcess)
/Dev/Source/Stored Processes/procMeans(StoredProcess)
/Dev/Source/Stored Processes/procReport(StoredProcess)
/Dev/Source/Stored Processes/procSummary(StoredProcess)
/Dev/Source/Stored Processes/procTabulate(StoredProcess)
7 objects found.
```

Output 1. sas-list-objects

A total of seven stored processes were found which match the filter criteria specified in the sas-list-objects command. Notice the use of the `-types` option specified on the command. This option is used to determine which object type(s) should be searched for. If necessary, multiple types can be specified within a space-delimited list. The full set of available types might vary depending on which SAS solutions are installed, which you can find by navigating to the "`/System/Types`" folder in the SAS Folders tree. Otherwise, additional information can also be found in the *SAS® 9.4 Intelligence Platform: System Administration Guide* in the "General Instructions for Using the SAS Intelligence Platform Batch Tools" section.

The next few options used within the command are `-folderTree`, `-created`, and `-since`. These options help you scope the filter even further by searching for all stored processes contained within a specific folder that have been modified since the beginning of the current week.

Finally, the `-sort` option is included to control the specific order of the returned objects. By default the utility uses a sort order based on the objects' names. In this case, an alphabetical sort by the objects' paths is used instead.

Combined together, the filter and sorting options enable us to easily determine which objects need to be promoted to the target environment. In this case, we focus on the first two; cascadingCarData-package and cascadingCarData-stream.

A few of the available command-line options for this utility are highlighted in this example. Many of the utilities in the SAS Intelligence Platform, including the sas-list-objects tool, provide a rich set of filter options that can be specified separately or in unison, including the following:

- Date filtering: When a combination of the `-since` and `-before` options, along with `-created` and `-modified`, are used this utility performs a search for all content objects that have either been created or modified within a given time period. An extremely helpful feature here is that not only can the dates be specified as absolute dates, but relative date expressions can also be specified. This can be critical in situations where the utility is used within a scheduled process.
- Name filtering: The `-name` and `-nameMatchType` options enable you to perform searches against the name of a content object. By default the utility uses a "Contains" search, but other possibilities include "Equals" and "StartsWith." You can also specify the `-includeDesc` option in situations where the objects' descriptions need to be searched too.
- Folder path filtering: Use the `-folder` option when searching for objects that are contained directly within a folder, or `-folderTree` when searching within a folder and any of its subfolders.
- Notes and extended attributes filtering: Through the `-notes`, `-extName`, and `-extValue` options, you can perform advanced filtering by searching for objects that have any associated Notes or Extended Attributes.

STEP 2 - VERIFYING DEPENDENCIES

When dealing with content, it is important to understand how objects might be related to others, as well as understanding where data is actually coming from. By visualizing the relationships between objects, you can start to appreciate how changes to one object might affect others. Before promoting content across environments, it is useful to understand an object's set of dependencies. With this information, you can determine whether any of these other dependencies need to be included in the migration process. Two utilities added in the SAS 9.4 release can help with this process; sas-relationship-loader and sas-relationship-reporter.

LOADING RELATIONSHIPS

The purpose of the sas-relationship-loader utility is to collect relationship data for various SAS content objects into a central database. Once the data is loaded, reporting tools such as the sas-relationship-reporter can be used to see the flow of how objects are connected to one another. As a side note, the

SAS Management Console does provide a mechanism for automating the loading process. Within its Configuration Manager plug-in, under the Application Management node, simply navigate to:

```
SAS Application Infrastructure --> Web Infra Platform Services 9.4 -->
RelationshipContentService
```

From there, you can set your own custom schedule for when and how often this relationship information should be loaded.

When using the sas-relationship-loader, an administrative user is required to execute the utility since they have the necessary permissions to retrieve all of an object's relationships. The utility accepts many of the standard filtering options that have been discussed previously in order to determine which objects should be processed.

In the following example, the loader utility is executed against all objects that are contained within the "/Dev/Source" folder to ensure that their relationships have been processed. The `-folderTree` option is specified to ensure that only objects contained within the specified folder are retrieved.

```
$ sas-relationship-loader -folderTree "/Dev/Source"

11:31:24 INFO  Processed 58 total relationships for 12 objects.
sas-relationship-loader has completed.
```

Output 2. sas-relationship-loader

RELATIONSHIP REPORTING

Now that this information has been loaded, the sas-relationship-reporter utility can be used to create a report that indicates which dependencies an object might have. For example, an impact analysis report can be executed in order to determine what type of impact there would be if changes were made to an object. You can select from a set of pre-configured report types, or customize the report options to meet your needs using a variety of filter criteria. The pre-configured report types include the following:

- Lineage - returns the downstream dependencies a subject might have
- Impact - returns objects that would be impacted by any change made to the subject
- Direct dependencies - returns the set of direct dependencies a subject has
- Indirect dependencies - returns the set of objects the subject is directly or indirectly dependent on, regardless of the number of intermediary objects

This last report type, "Indirect dependencies" is extremely helpful in cases where you know that a subject is connected to a particular object, but you do not know where this connection is. For this task, the "Indirect dependencies" type is used in order to determine all of the dependencies for the stored processes, regardless of the number of levels involved.

```
$ sas-relationship-reporter -folderTree "/Dev/Source" -name
"cascadingCarData" -report indirectDependencies

"/Dev/Source/Stored Processes/cascadingCarData-package" (Stored process)
   Contains: "Parameters" (Prompt group)
      Is dependent on: "/Dev/Source/Data/CARS" (Table)
         Is dependent on: "/Dev/Source/Data/DevData" (Library)

"/Dev/Source/Stored Processes/cascadingCarData-stream" (Stored process)
   Contains: "Parameters" (Prompt group)
```

```
      Is dependent on: "/Dev/Source/Data/CARS" (Table)
         Is dependent on: "/Dev/Source/Data/DevData" (Library)
```

Output 3. sas-relationship-reporter

Output 3 gives shows the exact details as to what objects these two stored process are dependent on. Notice the labels “Contains” and “Is dependent on”. Each relationship modeled within the SAS relationship database consists of a relationship type, as well as additional metadata such as when a relationship was created. In this example, “Contains” describes the relationship between the stored processes and their prompts, whereas the “Is dependent on” label describes the relationship between the prompts and the “CARS” table, and the same for the underlying “CarData” library.

Armed with this information, you now have a good understanding of which dependencies are used by the stored processes, and what else would need to be promoted to the target environment.

STEP 3 - CONFIGURING THE TARGET ENVIRONMENT

This step focuses on setting up the target environment to ensure that the content is promoted to the proper location and that the necessary set of access controls are applied such that only a select group of users are allowed to view the content. Note that it is possible for this target location to exist on the same system as the source system, or on an entirely different system altogether.

The first priority is to make sure that the proper folder structure is created within the SAS Folders hierarchy. In order to create the specific folders, the sas-make-folder utility can be used. When the `-makeFullPath` option is specified as follows, the utility is instructed to automatically create the intermediate folders, thus preventing the need to invoke the utility multiple times.

```
$ sas-make-folder -makeFullPath "/Dev/Target/Playpen"

INFO  Folder /Dev/ created.
INFO  Folder /Dev/Target/ created.
INFO  Folder /Dev/Target/Playpen created.
```

Output 4. sas-make-folder

Based on Output 4, all three folders were created successfully. The next step is to make sure that the proper set of access controls are established for this location. In this particular case, only a select number of users and groups should be allowed to view and edit the content that is contained within this folder. First, the sas-show-metadata-access utility can be used to view which access controls are currently configured.

```
$ sas-show-metadata-access -effective "/Dev/Target/Playpen"

-grant "SASAdministrators(UserGroup)":Administer, CheckInMetadata, Read,
ReadMetadata, WriteMetadata, WriteMemberMetadata
-deny  "SASAdministrators(UserGroup)":Delete, Write, Create
-deny  "SASUSERS(UserGroup)":Administer, CheckInMetadata, Delete, Read,
ReadMetadata, Write, WriteMetadata, WriteMemberMetadata, Create
-deny  "PUBLIC(UserGroup)":Administer, CheckInMetadata, Delete, Read,
ReadMetadata, Write, WriteMetadata, WriteMemberMetadata, Create
```

Output 5. sas-show-metadata-access

The information that is returned from this command confirms that the SASUSERS and PUBLIC groups are denied access to this location, and only the SASAdministrators group has the necessary ReadMetadata and WriteMetadata permissions to manage the content. By using the `-effective` option, you can view all of the inherited permissions, not just the ones that were explicitly set on that folder.

This is a good start, but we would like to grant one other group, "PlaypenDevelopers", write access in order for its members to make necessary changes. Whereas the sas-show-metadata-access tool enabled us to view the access control settings, the sas-set-metadata-access tool provides us the ability to set additional access controls. With this tool you can either grant or deny any user or group access to a set of metadata permissions. The `-grant` option is used in this example to grant the ReadMetadata, WriteMetadata, and WriteMemberMetadata permissions to the "PlaypenDevelopers" group.

```
$ sas-set-metadata-access -grant

PlaypenDevelopers:ReadMetadata,WriteMetadata,WriteMemberMetadata
"/Dev/Target/Stored Processes"
```

Output 6. sas-set-metadata-access

The sas-set-metadata-access utility does not return any output by default if the execution was successful. And in this particular case, the desired access controls were properly configured. Alternatively, you could have also used the `-addACT` option to apply a specific Access Control Template to the folder. Not only can this tool add access controls, it enables you to remove previously set access controls as well.

STEP 4 - PROMOTING THE CONTENT

In the previous sections, we determined what content to promote, as well as properly configuring the environment where the content will be created. It is now time to run the promotion.

Promotion is known as the process of copying selected metadata and associated physical content within or between planned deployments of SAS software.

While it is possible to perform the promotion process through user interface applications such as SAS Management Console or SAS Data Integration Studio, it can also be managed via two command-line utilities. First introduced in SAS 9.2, the ExportPackage and ImportPackage tools enable information about content objects to be transported between environments.

The SAS ExportPackage tool enables you to export a collection of content objects (including folders, stored processes, tables, and many other object types) from a source metadata server to a SAS Package file (.spk). This tool provides you the ability to selectively choose the exact set of objects or folders to export. This process exports the metadata for the content objects, as well as any associated physical content files (for example, stored process source code). The exported SAS package file can then be imported to a target location, either within the same system, or within a completely different environment, even across hosts. When promoting content within the same environment, another option could be to use the CopyObjects utility instead because it is capable of storing the pertinent metadata and physical content in memory, rather than interacting with SAS package files as the transport mechanism.

EXPORTING CONTENT

With the help of both the sas-list-objects and the sas-relationship-reporter utility, we determined exactly which objects need to be promoted; two stored processes, one table, and one library. Because the table

and library are considered to be dependencies for the stored processes, they must be included in the promotion in order for everything to function properly on the target system. If, however, the table and library already existed in the target system, we could choose to connect the stored processes to these objects. For the sake of this example though, the dependent objects are included within the package.

The ExportPackage utility as described above can be used to export the data about these four objects and store the output within a SAS package file, named StoredProcesses.spk, on disk. Just as with many of the other batch tools, ExportPackage supports multiple filter options to search for the objects to export. Because it is already known which objects are to be promoted though, their paths can be explicitly specified by using the `-objects` option as follows.

```
$ ExportPackage -package StoredProcesses.spk -objects "/Dev/Source/Stored
Processes/cascadingCarData-package(StoredProcess)" "/Dev/Source/Stored
Processes/cascadingCarData-stream(StoredProcess)"
"/Dev/Source/Data/CARS(Table)" "/Dev/Source/Data/DevData(Library)"

INFO  ******** Starting Export Process ********
INFO  Running batch process.
INFO  Exporting the following objects:
1 Library object:
   /Dev/Source/Data/DevData
2 Stored process objects:
   /Dev/Source/Stored Processes/cascadingCarData-package
   /Dev/Source/Stored Processes/cascadingCarData-stream
1 Table object:
   /Dev/Source/Data/CARS

INFO  ******** Exporting Metadata ********
INFO  Metadata exported successfully.
INFO  ******** Exporting Content ********
INFO  ******** Analyzing Connections ********
INFO  The following connections will need to be mapped to target values
when this package is imported:
1 SAS Application Server mapping:
   - SASApp
1 Source Code Repository mapping:
   - SASApp.C:\stp-source

INFO  ******** Creating Package ********
INFO  Total time to run the export process: 1.10 seconds.
INFO  Created package file "StoredProcesses.spk".
INFO  The export process has finished successfully.
The export process has completed.
For more information, view the export log file:
C:\Users\sasuser\AppData\Roaming\SAS\Logs\Export_150218115506.log
```

Output 7. ExportPackage

The batch promotion tools display a fair amount of information within their respective log files. This information has been trimmed in these examples for easier reading.

The output from the export process includes details about which objects were exported, as well as the connections, if any, that these objects need to map to during the import process. In this particular example, the output indicates that the two exported stored processes need to be associated to a SAS

Application Server, as well as to a Source Code Repository (the location where the physical source code exists).

IMPORTING CONTENT

Once the package file has been created, it can now be imported into a target environment. The promotion utilities are capable of moving data across hosts, so the target environment can be of a different topology from the source environment.

Exported objects often contain associations or references to other external objects. As Output 7 from the export process showed, the stored processes are associated to a SAS application server (the server that the stored process is executed on), as well as a source code repository (the physical location where the source is contained). These associations or references are called connection points. When a package file is imported to a target system, these references need to be preserved in order for the objects to be imported correctly. In order for this to happen, these connection points must either be contained within the same package file, or must be present on the target system.

The ImportPackage utility provides the ability to alter these connection point mappings. However, in the next example we are assuming that these connections will remain unchanged.

The next step is to feed the package file, StoredProcesses.spk, into the ImportPackage utility and instruct the tool to create the data under the previously configured target folder location, “/Dev/Target/Playpen”. This tool is capable of promoting new content or overwriting existing content. When the package is loaded, it first determines which objects in the package will be created as new objects and which ones will be imported as an update. Updates occur only if an object with the same name in the same target location already exists. The import process can be executed as follows:

```
$ ImportPackage -package StoredProcesses.spk -target "/Dev/Target/Playpen"

INFO  Importing objects into "/Dev/Target/Playpen".
INFO  ******** Starting Import Process ********
INFO  Running batch process.
INFO  Importing the following objects:
Create 1 Library object:
   /Dev/Target/Playpen/DevData
Create 2 Stored process objects:
   /Dev/Target/Playpen/cascadingCarData-package
   /Dev/Target/Playpen/cascadingCarData-stream
Create 1 Table object:
   /Dev/Target/Playpen/CARS

INFO  The following connection mappings were specified:
1 SAS Application Server mapping:
   - SASApp --> SASApp

1 Source Code Repository mapping:
   - SASApp.C:\stp-source --> SASApp.C:\stp-source

INFO  ******** Importing Metadata ********
INFO  Metadata imported successfully.
INFO  ******** Importing Content ********
INFO  Total time to run the import process: 1.42 seconds.
INFO  The import process has finished successfully.
The import process has completed.
For more information, view the import log file:
C:\Users\sasuser\AppData\Roaming\SAS\Logs\Import_150218120307.log
```

Output 8. ImportPackage

Output 8 reveals a successful completion of the import process. The stored processes, table, and library were successfully promoted to the target location and mapped to the existing SAS Application Server and source code repository as designed.

REVIEW

And there you have it - in only a few steps, the task of migrating content to a new location is complete. All the while, using only a handful of command-line utilities, without any help required from other applications. The following list summarizes the tools involved to achieve this:

- sas-list-objects - used to perform the initial search of content
- sas-relationship-loader - load the relationship information for the searched content
- sas-relationship-reporter - view content objects' relationship data
- sas-make-folder - create folders within the SAS folders tree
- sas-show-metadata-access - report on configured access control settings
- sas-set-metadata-access - add or update existing access control settings
- ExportPackage - export content objects to a SAS package file
- ImportPackage - imports a SAS package file to a target environment

ADDITIONAL UTILITIES

Up to this point, this paper has discussed a few of the batch utilities that are available within the SAS Intelligence Platform. There are countless others that can be invaluable to any administrator. These include utilities for the following:

- Metadata Administration: utilities used for managing metadata content
    - sas-delete-objects - deletes existing objects from the SAS Folders hierarchy
    - sas-make-act - creates new Access Control Templates

- Metadata Server Administration: a collection of tools used for administering a SAS Metadata Server instance.
    - sas-analyze-metadata - analyzes online metadata repositories, looks for common error conditions, and performs a repair operation if necessary
    - sas-backup-metadata - performs a backup of the SAS Metadata Server
    - sas-create-repository - creates and initializes a new metadata repository
    - sas-delete-repository - deletes an existing metadata repository
    - sas-update-metadata-profile - for clustered metadata servers, updates metadata profiles used by SAS Application Servers with information about the cluster configuration

- Deployment Backup and Recovery: a suite of tools that provides a mechanism for backing up and recovering components of the SAS System across multiple tiers and machines, including the SAS Metadata Server, SAS® Content Server, and the SAS® Web Infrastructure Platform Database.
    - sas-backup - executes an ad hoc backup
    - sas-list-backups - displays history information about previous backup and recovery operations
    - sas-recover - performs a full or partial recovery of the system
    - sas-status-backup - retrieves the status of a given backup instance
    - sas-set-backup-schedule - sets the schedule that is used for determining when a backup occurs

More information about all of the utilities contained within this framework can be found within the *SAS® 9.4 Intelligence Platform: System Administration Guide* in the "SAS Intelligence Platform Batch Tools: Reference" section.

COMMON TOOL OPTIONS

A common set of command-line options is used across the various batch utilities to help provide consistency in behavior and usage. The following sections discuss these standard options. Note that the sas-list-objects tool is used simply as an example.

ACCESSING HELP INFORMATION

Standard help information for each utility can be viewed by specifying the `-?` or `--help` options as follows:

```
$ sas-list-objects -?

usage: sas-list-objects [options...]
Standard options:
  -?,--help                  Print help information.
  [additional options excluded]
```

Output 9. Utility Help Options

The help information that is returned displays both the list of common options, as well as the options that apply to the specific utility.

SUPPLYING CONNECTION DETAILS

Each tool in the SAS Intelligence Platform requires that you supply a set of options used to connect to the appropriate SAS server. Connection information can be specified by including the host, port, user, and password information as follows:

```
$ sas-list-objects -host hostname -port portnumber -user username -
password password [options...] [parameters...]
```

Output 10. Connection Options

Alternatively, when using a tool that connects directly to the SAS Metadata Server, you can specify the name of the appropriate Connection Profile by using the `-profile` option.

```
$ sas-list-objects -profile profile [options...] [parameters...]
```

Output 11. Connection Profile Option

Only utilities that connect to the SAS Metadata Server support the usage of the `-profile` option. When using the Deployment Backup or Relationship utilities, you must provide the connection information for the SAS Web Server or the SAS Web Application Server.

LOGGING DETAILS

Output from most of the utilities can be directed to an external log file. By default, log files are created under the SAS Application Data location within the user's home directory. The name and location of the log file can be altered by specifying an optional *filename* value to the `-log` option.

```
$ sas-list-objects -log filename [options...] [parameters...]
```

Output 12. Logging Options

CONCLUSION

The SAS Intelligence Platform contains a variety of batch utilities that can be invaluable to any SAS administrator. These utilities enable you to perform administrative tasks directly through a command-line interface without the need to interact with any UI application. This paper highlights a few of the utilities that are available within this framework and discusses situations in which they can be used together to complete a task.

REFERENCES

SAS Institute Inc. 2014. *SAS® 9.4 Intelligence Platform: System Administration Guide, Third Edition*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation.

ACKNOWLEDGMENTS

The authors would like to thank Gerry Nelson and Nancy Pipes from SAS Institute Inc. for reviewing this paper.

RECOMMENDED READING

McIntosh, Liz, Nancy Rausch, and Bryan Wolfe. 2014. "Understanding Change in the Enterprise." *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. Available at: http://support.sas.com/resources/papers/proceedings14/SAS396-2014.pdf

CONTACT INFORMATION

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

Eric Bourn
100 Campus Drive
Cary, NC 27513
SAS Institute Inc.
(919) 677-8000

Eric.Bourn@sas.com

Amy Peters
100 Campus Drive
Cary, NC 27513
SAS Institute Inc.
(919) 677-8000
Amy.Peters@sas.com

Bryan Wolfe
100 Campus Drive
Cary, NC 27513
SAS Institute Inc.
(919) 677-8000
Bryan.Wolfe@sas.com