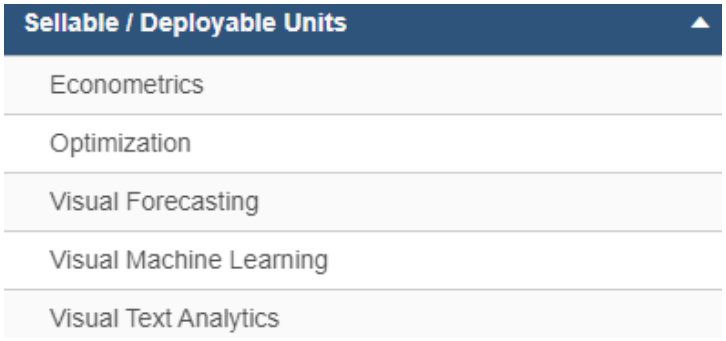
TECHNICAL PAPER

SAS® Viya® 3.4 to SAS® Viya® 4 Migration Best Practices

Last update: October 26, 2023

Relevant Products and Releases

- SAS Viya 4 – LTS 2022.1 on Microsoft Azure
    - SAS Viya 4 – LTS 2022.1



Content Migration

For migration from SAS Viya 3.4 to SAS Viya 4, both full system migration and content migration are supported. Based on the customer specific requirements and concerns in this scenario, the decision was made to go with the content migration route so that customers can have a well-organized and clean new system to manage critical business operations to meet business needs.

To migrate user content from SAS Viya 3.4 to SAS Viya 4, it must be exported from the SAS Viya 3.4 system and imported into SAS Viya 4. There are three approaches to performing the export and import, although they all rely on the same underlying base functionality. For this customer's migration, we used a combination of all the methods that are described here for portions of the migration. The content on this customer's system included SAS® Visual Analytics (VA) reports, folders, SAS programs, data files, pipelines, and data plans numbering in the tens of thousands for some users. Because of the quantity of material, we ran into memory limitations during the underlying export process and had to try various approaches to continue with the migration.

In this document, we outline some of the issues that we encountered and how we were able to work around them to complete the migration.

What Content Can Be Migrated/Promoted?

Content and objects can be migrated, but there are several exceptions. Our migration project was from SAS Viya 3.4 to SAS Viya 4, but you should check the specific supported content based on your source system, because content migration of additional types of content might be added regularly.

For detailed up-to-date information about support for SAS 9.4 content migration, see this link, which contains content that is updated regularly: SAS® Viya® Platform: Content Migration from SAS® 9.4.

Participating Resources: Promotable with the Transfer CLI

The following types of resources were promoted for this project: files, folders, SAS Model Studio projects, reports, and SAS Data Studio data plans. It is possible to promote additional types of content, but this should be verified by checking the most current SAS® Viya® Platform: Content Migration from SAS® 9.4 documentation.

Please note that when importing content, the creator of the resource might be changed to the ID of the person performing the import into the target environment. It is also important that all associated user IDs exist in the target environment before the content migration process is started.

Useful Settings

The biggest pitfall that we encountered during the migration of users' folders and their content was running out of memory. This is due to the underlying export functionality. So regardless of whether you export content from SAS® Environment Manager, use the CLI, or use the transfer.sh tool, you can still encounter this issue. This issue occurs because the system tries to hold all the content in memory until process completion, and it then writes everything out to a single file. The two ways to mitigate this limitation are by exporting smaller chunks of content or by increasing the memory limits. You can export smaller chunks of content more easily by using either the CLI or the transfer.sh tool, which will be covered in later sections. For the java_option_xss and java_option_xmx options, you can increase the memory even more than indicated below if the resources are available on your environment. We

required both mitigations for this project.

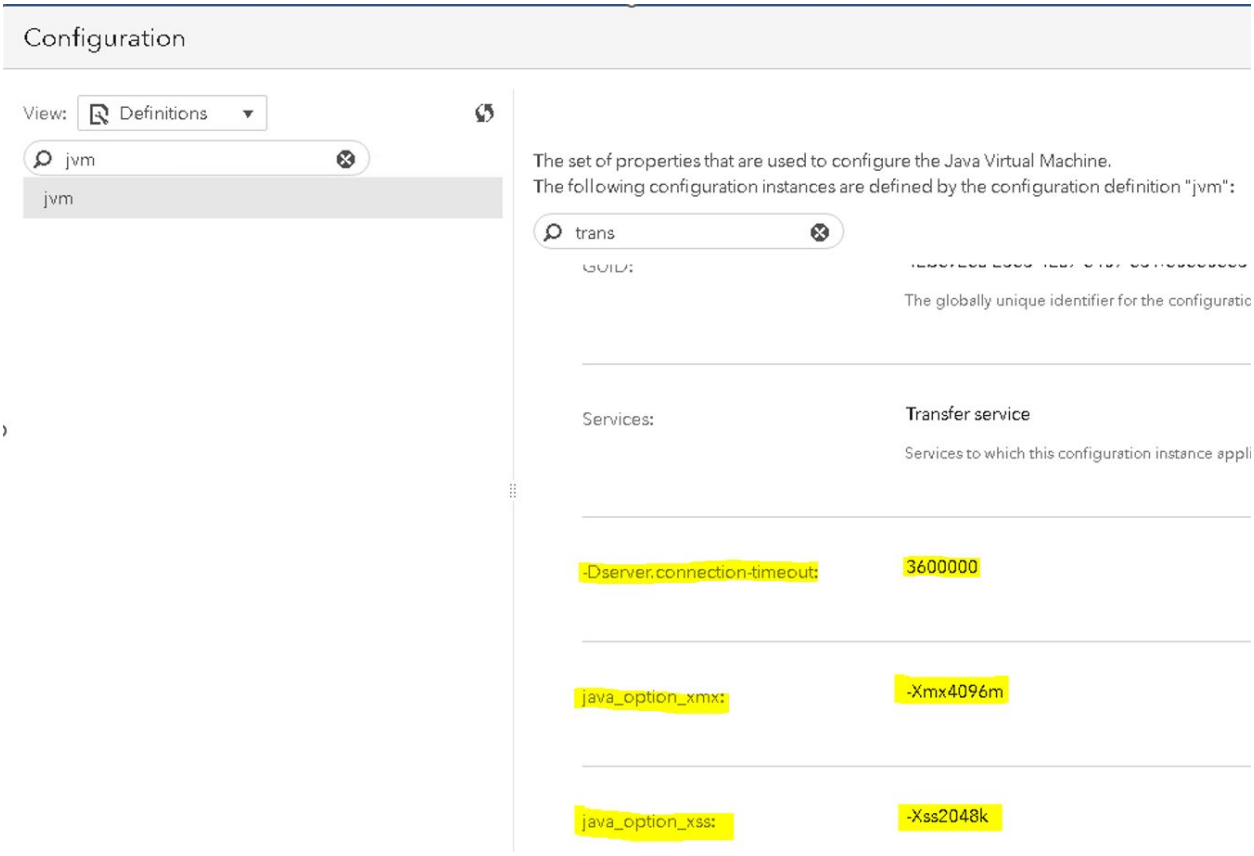
First, from the Configuration option in SAS Environment Manager (EVM), go to Definitions and select jvm. Search for the transfer service. Set the transfer options as shown in Figure 1:

`java_option_xss to -Xss2048k` (maximum memory per thread)

`java_option_xms to -Xms4096m` (maximum memory allocation for the jvm)

`-Dserver-connection-timeout to 3600000` (how long Java will wait for a response in milliseconds)

Figure 1. Configuration option



Next, from a command line in the SAS Viya 3.4 source environment, restart the transfer and file services as follows:

```
sudo systemctl restart sas-viya-transfer-default
sudo systemctl restart sas-file-service-default
```

Three Methods of Promoting Content

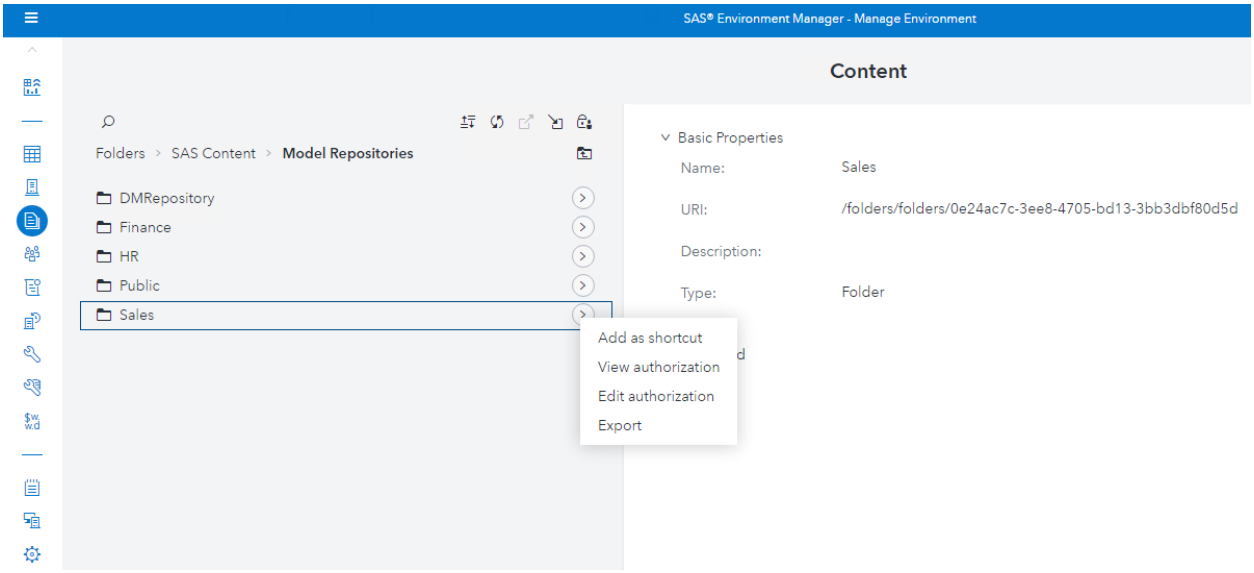
Because all methods utilize the underlying Transfer service, if you experience issues with any of these approaches, you might need to restart the Transfer service and File service (use the commands above) before attempting a different approach.

SAS® Environment Manager

Exports and imports can be performed from SAS Environment Manager on the front end. This works well for a small number of items.

You can pick specific content elements to export in SAS Viya 3.4 using the SAS Environment Manager Export function as shown in Figure 2. An important tip is that the ID used to perform the export tasks should be in the SAS Administrators group, so that you have access to pull all the content with the correct permissions.

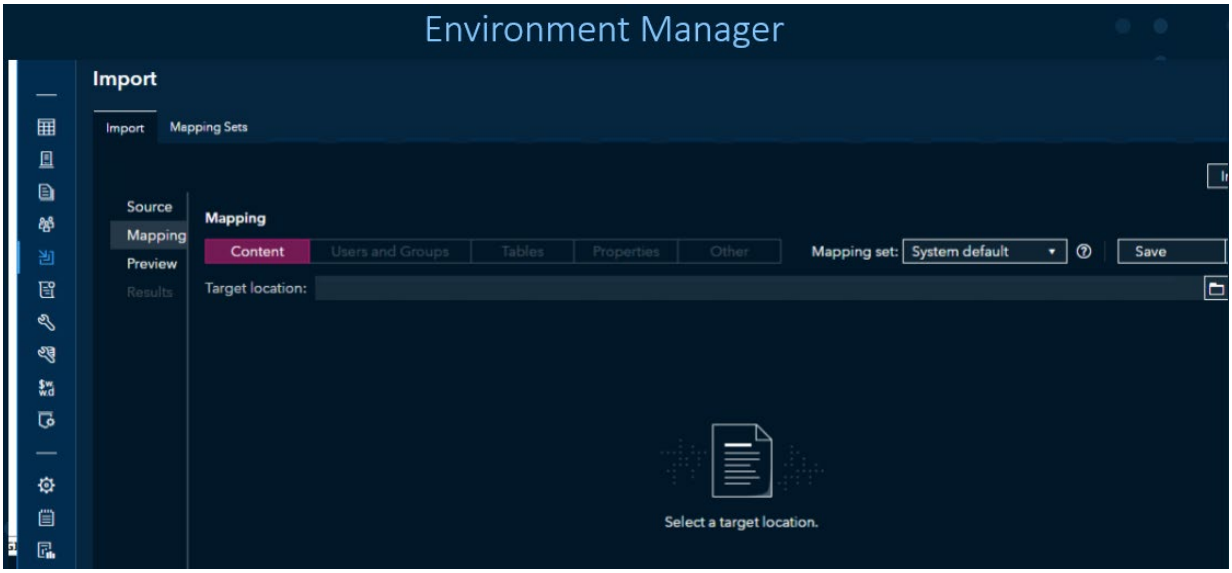
Figure 2. SAS Environment Manager Export function



Once the export packages from the SAS Viya 3.4 source environment are created successfully, then you can copy them to a jump server where you can access a SAS Viya 4 target environment and import those packages, as shown below. The same tip applies here—the ID should be in the SAS Administrators group. Also, the ID that is used to import the packages on the target system should match and have the same permissions as the ID that is used to export the packages on the source system. As shown in Figure 3 and 4, you can select the import file from a local disk and then set up mappings from the source environment to the target environment to achieve a successful import.

Figure 3. Import file from a local disk

Figure 4. Set mappings



SAS Command-Line Interface

To use the SAS CLI, you need to set up the SAS CLI and create a user profile on both the source and the target system. The sas-admin CLI should be downloaded and set up for SAS Viya 3.4, and the sas-viya CLI should be downloaded and set up for SAS Viya 4. Download the latest sas-admin tools here: SAS Downloads: SAS Administration CLI

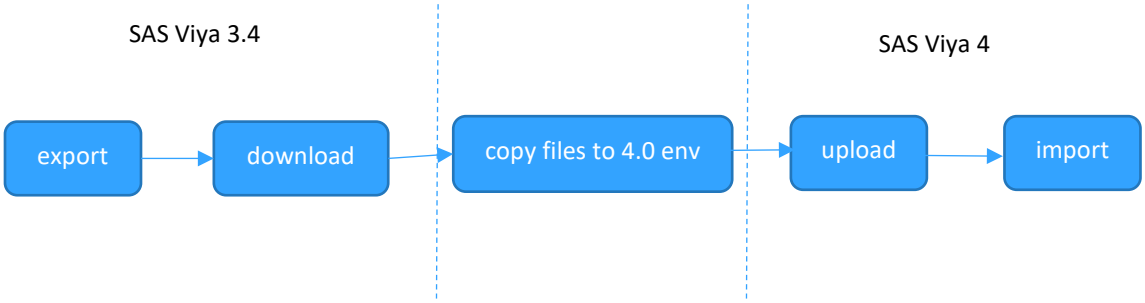
Here is an example of creating a user profile:

```
[user ~]$ sas-admin --profile Source profile init
Enter configuration options:
Service Endpoint> http://source.viya.example.com
Output type (text|json|fulljson)> text
Enable ANSI colored output (y/n)?> y
Saved 'Source' profile to /your_source_path/.sas/config.json.
```

```
[user ~]$ sas-viya --profile Target profile init
Enter configuration options:
Service Endpoint> http://target.viya.example.com
Output type (text|json|fulljson)> text
Enable ANSI colored output (y/n)?> y
Saved 'Target' profile to /your_target_path/.sas/config.json.
```

Figure 5 shows the workflow diagram and the commands that you need to run in sequence to export, download from SAS Viya 3.4, and upload and import into SAS Viya 4:

Figure 5. Workflow diagram



When the export completes, you must obtain the package ID that it prints out for the download step. Likewise, when you perform the upload, it prints another ID number, which you need to use for the import. The example here is for user folders, but the same commands should work for reports and other content, too.

Export and download folders 3.4

Using the default profile that you have created in the previous step, you can do the following:

```
export
SSL_CERT_FILE=/opt/sas/viya/config/etc/SASSecurityCertificateFramework/cacert
s/trustedcerts.pem
./sas-admin --profile viya34 auth login
./sas-admin --profile viya34 transfer export --timeout 60m -u <folder uid -
can get from environment manager>

#Use the .json file generated from the previous step.
./sas-admin --profile viya34 transfer download  -f ./<nameofoutputfile>.json
--id <package uid>
```

Upload and import folders 4.0

Using the default profile that you have created in the previous step, you can do the following:

```
./sas-viya --profile viya4 auth login
./sas-viya -profile viya4 --output json transfer upload -f
./<nameofoutputfil>.json -id <id from download>
./sas-viya -profile viya4 --output text transfer import  --id <uid from
upload step>
```

In the SAS Viya 3.4 environment (the source environment), you perform the export. An example of each command along with its console output is below:

Export and download folders 3.4

```
-sh-4.2$ ./sas-admin --profile viya34 transfer export -u
/folders/folders/a95d4857-9397-4700-9e23-07eaca2083be
running
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
..................................................... completed
{
    "items": [
        {
            "contentType": "",
            "message": "Transfer object processing complete.",
            "name": "PROD_CAT_OTHER-79d5cb45-9b78-4215-bf89-4386055e6682",
            "state": "completed"
        }
    ]
}

The package with the ID 118894d2-6dd5-421f-bf1c-85e6c1b383d4 was created.

./sas-admin --profile viya34 transfer download  -f ./prod_cat_other.json  --
id 118894d2-6dd5-421f-bf1c-85e6c1b383d4
```

The downloaded file might need to be copied to a location that is accessible from the SAS Viya 4 system for the upload and import steps.

Upload and import folders 4.0

```
./sas-viya --profile viya4 transfer upload --file prod_cat_other.json --id
118894d2-6dd5-421f-bf1c-85e6c1b383d4

$ ./sas-viya -profile viya4 --output json transfer upload -f
./prod_cat_other.json
{
    "id": "5a1db5f4-8d69-49f3-a4e6-720aa67e0f77",
    "name": "Export"
}
$
```

Notice that the upload step produces yet another ID that will be needed for the final import step:

```
$ ./sas-viya -profile viya4 --output text transfer import  --id 5a1db5f4-
8d69-49f3-a4e6-720aa67e0f77
```

Because using multiple IDs is not easy to work with and we had a large number of items to export and import, we scripted parts of the process. We piped the output to text files, which were cleaned and piped into the subsequent step. Obviously, any output from the export steps in the SAS Viya 3.4 source environment that was needed to perform the import had to be copied to the SAS Viya 4 target environment before continuing.

Even if the export and import process is only somewhat scripted, it is still time saving. This is true even though it is still advisable to review the output of each step before continuing in case there are unexpected errors in the output. In this case, some output might need to be cleaned up manually before continuing.

```
# export and download folders 3.4
export
SSL_CERT_FILE=/opt/sas/viya/config/etc/SASSecurityCertificateFramework/cacert
s/trustedcerts.pem
./sass-admin --profile viya34 auth login
./export_folders.sh >export_folders_output.txt  &
grep name  export_folders_output.txt |   sed 's/"name": "//g' | sed 's/",//g'
>folder_name.txt
grep ID export_folders_output.txt    | sed 's/The package with the ID //'  |
sed 's/ was created.//' > folder_package_ids.txt
./combine_folders.sh  > folders_download_input.txt
# remove lines with missing info
./download_folders.sh > download_folders_output.txt &


# upload and import folders 4.0
./sassily --profile viya4 auth login
./sas-viya -profile viya4 --output json transfer upload -f
./model_export_test.json
./sas-viya -profile viya4 --output json transfer upload -f  user_folders/
$ ./sas-viya -profile viya4 --output text transfer import  --id 5a1db5f4-
8d69-49f3-a4e6-720aa67e0f77
```

Here is a breakdown of the commands and scripts. These steps could be developed into a single script that performs all these tasks.

```
# export and download folders 3.4
export
SSL_CERT_FILE=/opt/.../SASSecurityCertificateFramework/cacerts/trustedcerts.p
em
./sas-admin --profile viya34 auth login


# export_folders.sh
./export_folders.sh >export_folders_output.txt  &
#!/bin/bash
while read line; do
        echo $line
        ./sas-admin --profile viya34 transfer export --timeout 60m -u $line
done < folder_names.txt
```

```
# prepare input
# clean and then combine names and ids
grep name  export_folders_output.txt |   sed 's/"name": "//g' | sed 's/",//g'
>folder_name.txt

grep ID export_folders_output.txt    | sed 's/The package with the ID //'  |
sed 's/ was created.//' > folder_package_ids.txt

# combine_folders.sh
./combine_folders.sh  > folders_download_input.txt
#!/bin/bash
while IFS=$'\t' read -r f1 f2
do
echo $f1  $f2
done < <(paste folder_name.txt folder_package_ids.txt)


# download_folders.sh
# remove lines with missing info
./download_folders.sh > download_folders_output.txt &
#!/bin/bash
while read -r name id; do
        echo "$name" "$id"
        ./sas-admin --profile viya34 transfer download  -f ./$name.json  --id
$id
done <  folders_download_input.txt



# Another useful tool, jq, can grab ids from *.spk files to simplify some of
the scripting. Example:

import_id=$(./sas-viya transfer upload --file "user_path\\identities.spk" |
jq -r .id)
echo "Importing identities The ID of the package to import is" $import_id
./sas-viya -q  transfer import --id "$import_id" --mapping
"user_path\\mapping.json"
```

SAS Technical Support's Transfer.sh

Another option is to use the transfer.sh script that is available here: Shell Scripts/transfer.sh GitLab (sas.com). The transfer.sh script creates a separate JSON file for each folder in a given path. For example, if you instruct it to export **/Users**, it creates an export for **/Users/user1**, **/Users/user2**, and so on. While navigating and exporting folders, the script ignores anything that is not a folder, and any such "loose" items will not be included in the export. For this to work well, we had to ask users to reorganize their user folders so that each folder contained either all folders or all files. This prevented the script from ignoring content and not including it in the JSON file. In addition, we asked users to clean up their workspaces by removing any data files from their folders; these were causing issues because of their size and could be easily copied over to the new environment separately.

Although there is an option for the transfer.sh script that exports files instead of folders, we did not use it for this project. We recommend reviewing the documentation for transfer.sh when planning your migration to gain awareness of features that are not included in this paper and any new enhancements to the script. For example,

transfer.sh can now support exporting all objects of a given type, regardless of the path of a given chunk size. The transer.sh tool enhances the existing CLI functionality by automatically navigating through SAS content from a given starting point and creating a separate JSON file for each subfolder. This allows it to export more content with a single command without running out of memory.

To use this tool, first download the latest sas-admin tools: SAS Downloads: SAS Administration CLI

If you get an error because the jq package is not installed, execute this command from Linux (might require sudo access):

```
yum install epel-release -y; yum install jq –y
```

The transfer.sh script is capable of both exporting and importing. For this project, we did not use the import capabilities.

For example, the command below creates a separate export package for each subfolder in the **`/Users`** directory; download it into a path in **`/tmp`**. It also performs a check against the source and, on failure, attempts to export/download again for the specified number of retries, which was five in this case.

Because the import flag is also set, it then uploads and imports the package and finally compares the source and destination for errors. This usage assumes that the source and target are on the same system or at least both are accessible from the location where the transfer script is running. In our environment, the 3.4 and 4 installations were completely separate, and packages had to be transferred before importing:

```
./transfer.sh --export --expcheck --srccheck --import --impcheck --tgt-
profile Target --src-profile Source --output-path /tmp --content-path /Users
--retries 5
```

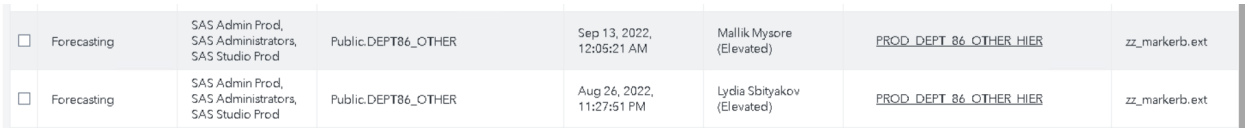
To skip the import steps, simply remove the import related flags:

```
./transfer.sh --export --expcheck --srccheck --src-profile Source --output-
path /tmp --content-path /Users --retries 5
```

SAS® Visual Forecasting Projects Migration

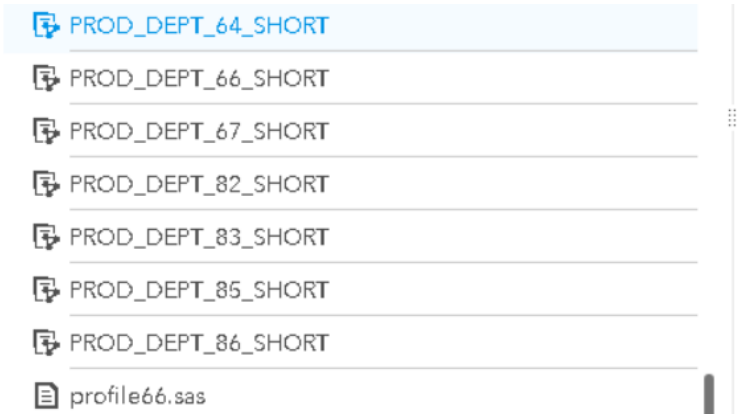
For SAS Visual Forecasting projects migration, SAS Help Center: Migration within the SAS Viya Platform: Tasks suggests two methods, SAS Environment Manager or the transfer plug-in to the CLI. Using SAS Environment Manager, the first step of the migration is to locate the projects from the source system, SAS Viya 3.4 in this case. However, not all projects can be found in the users' folders in SAS Environment Manager. The projects in Figure 6 indicate that the SAS Visual Forecasting project **PROD_DEPT86_OTHER_HIER** is created by the user zz_markerb.ext, but in the "My folder" of user zz_markerb.ext (Figure 7), the project does not exist.

Figure 6. Projects in Model Studio on SAS Viya 3.4



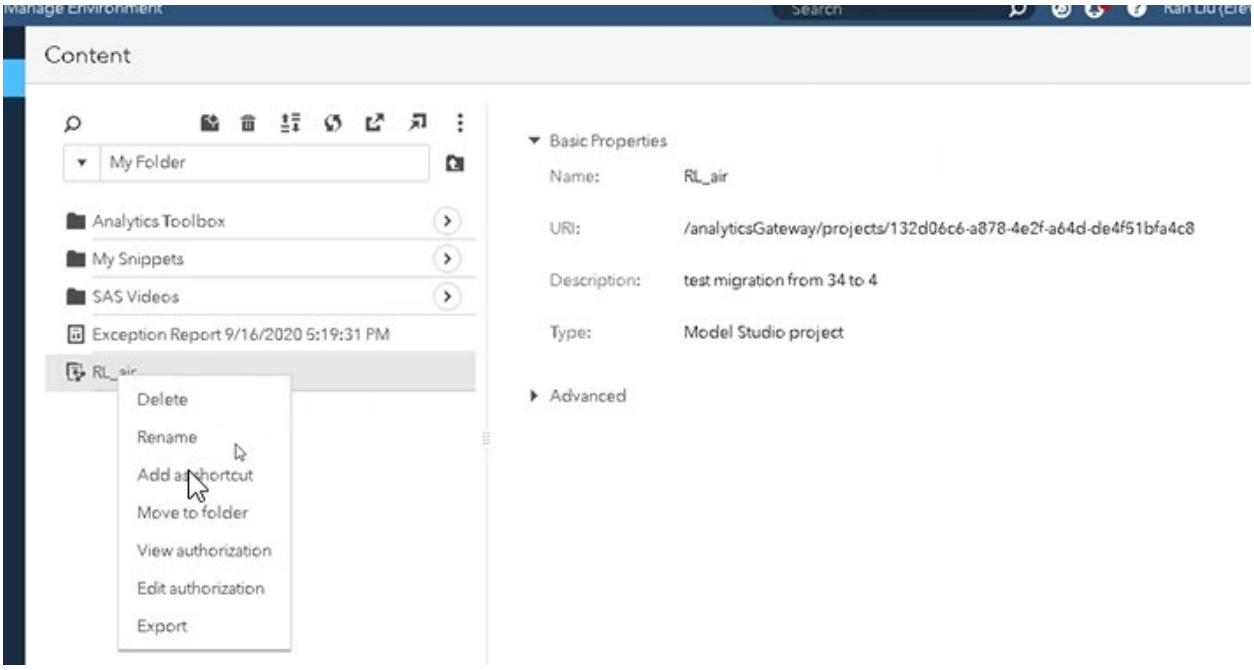
☐	Forecasting	SAS Admin Prod, SAS Administrators, SAS Studio Prod	Public.DEPT86_OTHER	Sep 13, 2022, 12:05:21 AM	Mallik Mysore (Elevated)	PROD_DEPT_86_OTHER_HIER	zz_markerb.ext
☐	Forecasting	SAS Admin Prod, SAS Administrators, SAS Studio Prod	Public.DEPT86_OTHER	Aug 26, 2022, 11:27:51 PM	Lydia Sbityakov (Elevated)	PROD_DEPT_86_OTHER_HIER	zz_markerb.ext

Figure 7. SAS Visual Forecasting projects under "My folder" of zz_markerb.ext



So, exporting from the UI is not an option. In order to export using the CLI, the URI is needed in the format of: **`/analyticsGateway/projects/<project-id>`**. See Figure 8 for a SAS Visual Forecasting project under the user's folder with a valid URI. The workaround that we used to find the URI is by going to the CAS folder on disk that hosts all the project data, and then reformatting the paths so that they match the URI format. You can also build some logic to exclude the projects that are older than a certain range.

Figure 8. SAS Visual Forecasting project with URI



The next step is to pass the URIs to the CLI that is documented on Page **Error! Bookmark not defined.** of this paper, export and download the projects, and then import them to the target SAS Viya 4 environment.

Note in the SAS Help Center: Promotions and Upgrades within SAS Viya for Visual Forecasting, there is a list of information to be considered before performing a promotion:

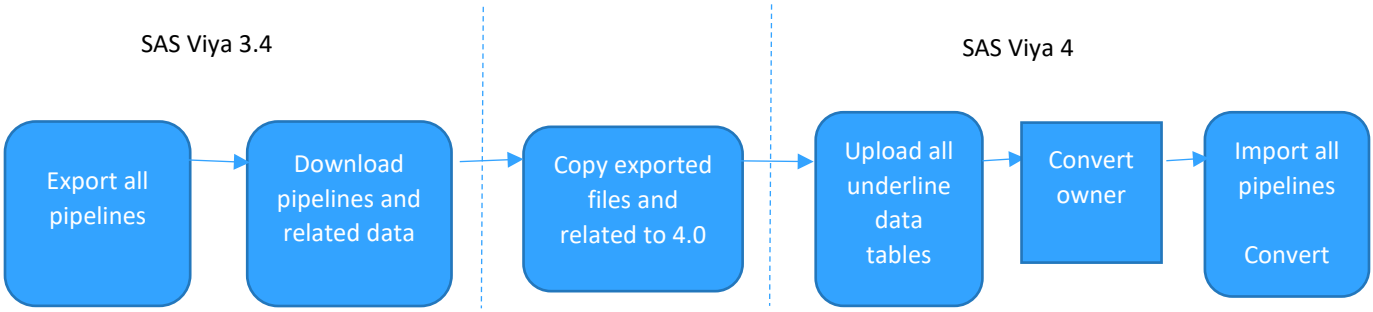
- The owner of a project that is being promoted must sign into the target environment before any projects can be imported.
- Before you promote a project, you must promote the input data for that project to the target environment.
- A user-created pipeline or node template must be promoted separately before you can use it in a new project. If a custom template is derived from another template in the exchange, both templates are required to be on the source system in order to successfully import projects that use the template.
- You must rerun all nodes and pipelines in a promoted project before the results are available in the target environment.

The first item asks the project owner to sign into the target environment before any projects can be migrated. Errors occur if that condition is not met. For the cases that the owner of the project in the source system cannot sign in to the target environment, please contact SAS support team to convert the owner to a user who has already signed in to the target environment. A script can be easily developed to convert the owner for all projects in a given folder. We developed two SAS scripts to convert the owner and to load input data. Once the owner is converted and after the input data is promoted and loaded into CAS, the projects can be imported.

One specific item that you must pay additional attention to is that this version of the SAS Viya 4 environment requires table/column names to have consistent cases. Target environment input tables and column name casing must match exactly with the source environment. Once all projects are imported successfully, the project access should be changed to include all parties that need access. See the SAS Visual Forecasting project migration flow chart (Figure 9) below.

Figure 9. SAS Visual Forecasting project migration flow chart

The flow is similar for the different methods that were previously shown.



Here is an example script to load data into CAS written by a team member:

```
/*LS tables*/
cas a; caslib _All_ assign;
%let master_list = 41 42 … 85 86;
%let reqd_list   = 41 42 … 85 86;
/*reqd list is a subset of the master list or could be the entire master list
itself*/
/*change reqd_list to promote reqd ls tables*/
options mlogic mprint symbolgen;
%macro promote_tables(promote_list);
    %do i=1 %to %sysfunc(countw(&promote_list));
        %let temp=%scan(&promote_list.,&i,%str( ));
        proc casutil outcaslib="PUBLIC";
        LOAD CASDATA="LS_output_&temp..sashdat" INCASLIB="PUBLIC"
CASOUT="LS_OUTPUT_&temp." PROMOTE;
    QUIT;
        %end;
%mend;
%promote_tables(&reqd_list)
```

Conclusion

Migration can be challenging, especially when dealing with large amounts of content. In this scenario, the underlying export process attempts to export all selected content into a single JSON file, which can use all available memory. This can be mitigated in two ways: by adjusting the memory options as outlined in this paper or by decreasing the amount of content that is selected at a time. By incorporating some of the scripting techniques presented in this paper, the amount of manual work that is necessary for the migration process can be reduced considerably. This is especially true if all or part of the migration process must be repeated due to cutover requirements or the need to set up multiple environments.

References

GitLab. 2023. “Shell Scripts/transfer.sh.” Available at **gitlab.sas.com/tmt/metadata-programs/-/blob/master/Shell%20Scripts/transfer.sh**.

SAS Institute Inc. 2023. “Migration within the SAS Viya Platform: Tasks.” *SAS® Viya® Administration*. Cary, NC: SAS Institute Inc. Available at **go.documentation.sas.com/doc/en/sasadmincdc/v_027/calcontentmig3x/n0ucexhkgs4rfgn12219vojr14nf.htm**.

SAS Institute Inc. 2023. “Participating Resources: Promotable with the Transfer CLI.” *SAS® Viya® 3.4 Administration*. Cary, NC: SAS Institute Inc. Available at **go.documentation.sas.com/?cdcId=calcdc&cdcVersion=3.4&docsetId=calpromotion&docsetTarget=n0y8myshonb0fvn1egg1e4qzt0l5.htm&locale=en#p0u55d1s0bhc5en1bfu7p27fld5w**.

SAS Institute Inc. 2022. “Promotions and Upgrades within SAS Viya.” *Model Studio: SAS® Visual Forecasting*. Cary, NC: SAS Institute Inc. Available at **go.documentation.sas.com/doc/en/vfcdc/v_012/vfug/p0pzm1az06w0nsn17ukbe6jua339.htm**.

SAS Institute Inc. 2023. “SAS Administration CLI Downloads.” Available at **support.sas.com/downloads/package.htm?pid=2133**.

SAS Institute Inc. 2023. “SAS® Viya® Platform: Content Migration from SAS® 9.4.” *SAS® Viya® Administration*. Cary, NC: SAS Institute Inc. Available at **go.documentation.sas.com/doc/en/sasadmincdc/v_027/calcontentmig9x/titlepage.htm**.

Release Information Content Version: 1.0 October 2, 2023.