SAS5562-2016

New for SAS® 9.4: Integrating Third-Party Metadata for Use with Lineage and Impact Analysis

Liz McIntosh, SAS Institute Inc.

ABSTRACT

Customers are under increasing pressure to determine how data, reports, and other assets are interconnected, and to determine the implications of making changes to these objects. Of special interest is performing lineage and impact analysis on data from disparate systems (for example, SAS®, IBM DataStage, Informatica PowerCenter, SAP BusinessObjects, and Cognos). SAS provides utilities to identify relationships between assets that exist within the SAS system. Beginning with the third maintenance release of SAS® 9.4, these utilities support third-party products. This paper provides step-by-step instructions, tips, and techniques for using these utilities to collect metadata from these disparate systems, and then performing lineage and impact analysis.

INTRODUCTION

Data lineage is the management and analysis of object and metadata relationships, including dependencies and lifecycle. This management and analysis process reveals where data comes from, how it is transformed, and where it is going. Impact analysis reveals which objects are affected when another object is changed or deleted.

Enterprises large and small use a variety of products and solutions to produce data, processes, reports, and other content. The challenge is to understand how the content produced relates to each other. With a good understanding of the relationships, individuals throughout the enterprise can answer questions pertaining to data lineage and impact analysis.

The data steward, for example, can assess the impact of a change to the data model. Before increasing the length of a column, the data steward asks questions like "Where is a column used?" and "Will the data be truncated by an ETL process using the column?" Likewise, before a job is deleted, the ETL developer can determine which reports are impacted and contact the report owners. Or the business analyst, upon seeing a surprising result in a report, can trace the origins of the data to verify that the result is indeed accurate before taking action.

Lineage capabilities, especially our ability to tie business and technical information together in a single and cohesive information store, are strategic toward helping you understand all the data assets in your environment.

Managing data as a key corporate asset is the primary premise behind the concept and discipline of data governance. Data governance has grown in importance as organizations are forced to comply with industry or government regulations, cut costs to improve margins, or use data-driven initiatives to increase revenue. A successful data governance initiative helps by improving the visibility of a corporation's data assets, which can be used to drive better and quicker business decisions. Data lineage and impact analysis are important components of any data governance initiative.

SAS METADATA BRIDGE RELATIONSHIP LOADER ARCHITECTURE

The SAS® Metadata Bridge Relationship Loader, new with the third maintenance release for SAS 9.4, enables you to use the SAS metadata bridges to load third-party lineage information to the SAS Relationship Service.

The metadata bridges provide support for accessing metadata from data management, business intelligence, and data integration third-party tools. The accessed metadata is profiled for physical and logical object relationships. Discovered relationships are then imported to the relationship service database. For a full list of supported tools, see http://support.sas.com/software/bridges/.

PREPARING TO IMPORT THIRD-PARTY METADATA

Lineage, as a component of a data governance plan, includes data assets that touch every aspect of an organization. The metadata for these assets is collected and integrated to support lineage and impact analysis. An initial challenge to getting started is the amount of metadata. The volume of metadata can be overwhelming, and the loading might take a very long time. In addition, when you are dealing with a large volume of metadata, the process of making sense of it all can be difficult.

For this reason, it is recommended to start with a phased approach. It is best to start with data flows for key decision processes. A bank, for example, might choose to start with a data flow that creates risk calculations. Start small and add additional flows in multiple phases.

Once a decision process is chosen for the initial phase of lineage, identify subject matter experts for the process. The subject matter experts will help determine which assets need to be captured, where the assets reside, and which tools are used create and manage the assets. The subject matter experts will also help identify the integration points between different tools and validate the analysis results.

Responsibility for data assets associated with a business process might be distributed between different business units. Subject matter experts need to be identified for all data assets. The subject matter experts need to be available to help gain the appropriate access and understand the data and business uses of the data.

ADDING THIRD-PARTY CONTENT USING SAS METADATA BRIDGE RELATIONSHIP LOADER

The SAS Metadata Bridge Relationship Loader batch utility is only supported on Microsoft Windows operating systems. The utility is an executable file launched from a command line. On a typical Windows installation of SAS 9.4, it is located in the directory `<SASHOME>\SASMetadataBridges\4.1\tools\`.

The metadata bridge relationship loader is launched using the following:

```
sas-metabridge-relationship-loader [options…]
```

The following command-line connection options are required to execute the batch utility:

- -host : the host name of the SAS middle-tier
- -port : the port of the SAS middle-tier
- -user : the user name; must specify an unrestricted user
- -password : the user's password

Option names are case-sensitive.

The remainder of this section provides a guide through the steps to obtain a list of available bridges, review the bridge requirements, load relationships, and create equivalent relationships.

For brevity, the examples that follow do not display the connection options.

STEP 1: OBTAIN A LIST OF AVAILABLE BRIDGES

The first step is to determine if the bridge for the data of interest is licensed. The list of licensed bridges is obtained by using the `-bridgeList` option:

```
sas-metabridge-relationship-loader -bridgeList
  -outputFile "C:\temp\bridgeList.xml"
```

A list of the available bridges is written to the file specified in the `-outputFile` option (in this example, "bridgeList.xml"). The format of the bridge list is XML. The file can be viewed using a web browser. Alternatively, you can view the file with an XML editor or text editor that supports both Windows and UNIX line-terminating characters.

```
<?xml version="1.0" encoding="UTF-8"?>
  <ImportBridgeList>
    <Bridge bridgeDisplayName="CA ERwin 7.x Data Modeler (File)"
      bridgeIdentifier="CaErwin7Xml"/>
    <Bridge bridgeDisplayName="Oracle Database (via JDBC)"
      bridgeIdentifier="JdbcOracle"/>
  </ImportBridgeList>
```

Output 1. The bridgeList.xml File

The bridgeList.xml file in Output 1 includes a bridge element for each bridge. The element has an attribute for the display name of the bridge and another for the bridge identifier. Make note of the bridge identifier for the bridge of interest, because this value must be used with the `-bridgeIdentifier` option for the rest of the batch utility functions. For the examples in this paper we are referencing the Oracle Database (via JDBC) bridge which is part of the SAS Metadata Bridges for General Industry Standards. The bridge identifier is JdbcOracle.

The metadata bridges check the license contained in the metadata server that is associated with the mid-tier specified by the web connection options. If no bridges are listed, verify that the metadata server has been updated with the license information. For instructions on how to update the SID file in metadata, see your software license renewal instructions (SAS Institute Inc. 2015a).

STEP 2: REVIEW THE BRIDGE REQUIREMENTS

The next step is to review the requirements for the bridge. Detailed information for a bridge is obtained by using the `-bridgeInfo` option:

```
sas-metabridge-relationship-loader -bridgeIdentifier JdbcOracle -bridgeInfo
  -outputFile "C:\temp\jdbcInfo.xml"
```

In this command, `-bridgeInfo` is used to request information about the Oracle Database (via JDBC) bridge using the JdbcOracle as the bridge identifier. JdbcOracle is the bridge identifier discovered by examining Output 1. The bridge information is written to the file specified by the `-outputFile` option, jdbcInfo.xml. This file contains details about the bridge, including requirements and troubleshooting tips. It is important to review all the information in this file before loading relationships.

Output 2 is a partial view of the file that was created by using the `-bridgeInfo` option in the previous command. The description contains important information to review before using the bridge to import metadata. In this example, the description explains that the bridge establishes a JDBC connection with a physical database. It is of *critical* importance to fill in the required parameters correctly to satisfy the connection requirements. The parameters are described later in the jdbcInfo.xml file. Also listed here is an *important* note about the permissions that are required to import metadata from the database. Specific required permissions are listed. It is common to encounter incorrect connection information and for the user to not have the required permissions. The subject matter expert for this data asset should verify the login information and permissions before using the bridge.

```
<ns1:GetBridgeInformationResponse bridgeDisplayName="Oracle Database
(via JDBC)" bridgeIdentifier="JdbcOracle" description="Import tool:
Oracle Oracle Database 6i to 12c (http://www.oracle.com/database)
Import interface: [Database] Data Store (Physical Data Model,
Expression Parsing) via JDBC API from Import bridge: 'JdbcOracle' 8.0.1
- 2014-11-25 10:27:59

    IMPORTING FROM AN ORACLE DATABASE USING JDBC.

This bridge establishes a JDBC connection with a physical database in
order to extract the physical metadata.  It is critical that the
parameters are filled correctly to satisfy the local connection
requirements on the client workstation that is running the bridge.
Please refer to the individual parameter's tool tips for more detailed
examples.

Important note about permissions:

The username specified must have the SELECT_CATALOG_ROLE role and
CONNECT permissions.
```

Output 2. Partial View of the jdbcInfo.xml File Showing Critical and Important Information

Output 3 notes another common issue and the workaround if you encounter this issue. Experience has shown that if a note or tip is listed in the description, then the information is important and worth reviewing.

```
Important note about Cursors:

Importing the metadata structure of a database is a highly recursive
activity that is an unusual type of access for a database. Hence, most
Oracle databases are not tuned for this type of access. The result is
often a message like:

    ORA-01000: maximum open cursors exceeded

The recommended solution is to contact the Oracle DBA and request an
increase in the setting for open_cursors in init.ora.
```

Output 3. Continuation of Bridge Description Showing Another Common Issue

Output 4 provides details about the JDBC driver and the expected location. The driver is not delivered as part of the SAS metadata bridges. You must locate the specified driver and then copy it to the default location or specify the location in the bridge options file. Obtaining and modifying the bridge options file is discussed in the "Step 3: Obtain and Modify the Bridge Options File" section. The default location is specified by "file location." [Meta Integration Home] refers to the install location of the SAS metadata bridges.

```
Driver details:

  - driver name: Oracle Database 11g Release 1 JDBC Driver
      (oracle.jdbc.driver.OracleDriver)
  - driver version / build: 11g / 11.1.0.7.0
  - file name: ojdbc6.jar (1,993,248 bytes)
  - file location: [Meta Integration Home]\java\Jdbc\oracle
```

Output 4. Continuation of Bridge Description Showing the JDBC Driver Details

The methodology attribute of the specification element is highlighted in Output 5. The information in this attribute gives insight into the type of metadata imported by this bridge. In this case, the Oracle Database (via JDBC) bridge obtains information about the physical data store and some expression parsing. Expression parsing for the database might include database triggers or view information.

The productName and productVersion attributes of the Specification element provide information about the specific product and version(s) supported by the bridge. If support is not listed for the version required, review the list of available bridges to determine if another bridge supports the correct version. If there is not a bridge that matches the exact version, check the list for another bridge that might apply. For the Oracle database in this example, the generic JDBC bridge, Database (via JDBC), might be a viable option.

```
<Specification isMultiModel="false" methodology="[Database] Data Store
(Physical Data Model, Expression Parsing) via JDBC API"
productCompany="Oracle" productName="Oracle Database" productVersion="6i
to 12c" productWeb="http://www.oracle.com/database" version="8.0.1 - 2014-
11-25 10:27:59" xsi:type="ns2:BridgeSpecificationType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns2="http://metaintegration.com/MIMB/Integration/5"/>
```

Output 5. Continuation of Bridge Description Showing the Type of Metadata Imported, Product Name, and Product Version

Details about the parameters are listed in the bridge description, and information about the "Host" parameter is shown in Output 6. The BridgeParameterIdentifier attribute (❶) specifies that "Host" is the name of this parameter. Additional details needed to provide a valid value of the parameter are in the "description" attribute (❷). The value of "Host" can be a host name, an IP address, or a fully qualified JDBC connection string. A value must be specified for this parameter because the "mandatory" attribute is set to "true" (❸). The default value, if any, is specified in the "DefaultValue" element (❹). The default value for this parameter is "localhost".

```
   <BridgeParameter
❶    bridgeParameterIdentifier="Host"
❷    description="Host name or IP address where Oracle database
       server is running. Please, check examples 1 and 2.
       OR
       Full qualified JDBC connection string for Oracle JDBC driver.
       Please, check example 3.

         Example #1:
         someservername.com

         Example #2:
         192.169.10.20

         Example #3:
         jdbc:oracle:thin:@(DESCRIPTION=(LOAD_BALANCE=on)
         (ADDRESS=(PROTOCOL=TCP)
         (HOST=oracleserver1)(PORT=1521))(ADDRESS=(PROTOCOL=TCP)
         (HOST=oracleserver2) (PORT=1521))(ADDRESS=(PROTOCOL=TCP)
         (HOST=oracleserver3) (PORT=1521))
         CONNECT_DATA=(SERVICE_NAME=orcl)))"
       displayName="Host"
❸      mandatory="true"
       transferable="false"
       type="STRING">

❹    <DefaultValue
       xsi:type="ns5:BridgeParameterValue"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:ns5="http://metaintegration.com/MIMB/Integration/5">
         localhost
     </DefaultValue>
   </BridgeParameter>
```

Output 6. Continuation of Bridge Description Showing Details for the "Host" Parameter

Bridge-specific parameters are specified in a bridge options file. Obtaining and modifying the bridge options file is discussed in the "Step 3: Obtain and Modify the Bridge Options File" section.

STEP 3: OBTAIN AND MODIFY THE BRIDGE OPTIONS FILE

After reviewing the bridge requirements and gathering the required information, it is time to specify the bridge options. Issue this command to retrieve the bridge options template and store it in the jdbcOptions.xml file:

```
sas-metabridge-relationship-loader -bridgeIdentifier JdbcOracle -bridgeInfo
  -outputFile "C:\temp\jdbcInfo.xml"
  -bridgeOptions "C:\temp\jdbcOptions.xml"
```

The command-line to retrieve the bridge options template is the same as for retrieving the bridge information, except that `-bridgeOptions` is specified. The initial file is shown in Output 7.

```
<?xml version="1.0" encoding="UTF-8"?>
<BridgeInformation bridgeDisplayName="Oracle Database (via JDBC)"
  bridgeIdentifier="JdbcOracle">
    <BridgeParameter bridgeParameterIdentifier="Driver location"
      displayName="Driver location"/>
    <BridgeParameter bridgeParameterIdentifier="Host" displayName="Host"
      mandatory="true">localhost</BridgeParameter>
    <BridgeParameter bridgeParameterIdentifier="Port" displayName="Port"
      mandatory="true">1521</BridgeParameter>
    <BridgeParameter bridgeParameterIdentifier="Service"
      displayName="Service" mandatory="true"/>
    <BridgeParameter bridgeParameterIdentifier="User" displayName="User"/>
    <BridgeParameter bridgeParameterIdentifier="Password"
      displayName="Password"/>
    <BridgeParameter bridgeParameterIdentifier="Schema"
      displayName="Schema"/>
    <BridgeParameter bridgeParameterIdentifier="Import stored procedures"
      displayName="Import stored procedures">False</BridgeParameter>
</BridgeInformation>
```

Output 7. Initial Bridge Options File

Some options, such as "User" do not have values specified:

```
<BridgeParameter bridgeParameterIdentifier="User" displayName="User"/>
```

To specify values for these options, first change the format of the XML tag to this:

```
<BridgeParameter bridgeParameterIdentifier="User" displayName="User">
</BridgeParameter>
```

Then specify the value for the option:

```
<BridgeParameter bridgeParameterIdentifier="User" displayName="User">
  dbmsadmin
</BridgeParameter>
```

An XML editor is useful when modifying the bridge options file. The modified bridge options file is shown in Output 8.

```
   <?xml version="1.0" encoding="UTF-8"?>
   <BridgeInformation bridgeDisplayName="Oracle Database (via JDBC)"
     bridgeIdentifier="JdbcOracle">
❶      <BridgeParameter bridgeParameterIdentifier="Driver location"
         displayName="Driver location"/>
❷      <BridgeParameter bridgeParameterIdentifier="Host"
         displayName="Host"mandatory="true">
           host01.example.com
       </BridgeParameter>
❸      <BridgeParameter bridgeParameterIdentifier="Port"
         displayName="Port" mandatory="true">
           1521
       </BridgeParameter>
❹      <BridgeParameter bridgeParameterIdentifier="Service"
         displayName="Service" mandatory="true">
           XE
       </BridgeParameter>
❺      <BridgeParameter bridgeParameterIdentifier="User"
         displayName="User">
           dbmsadmin
       </BridgeParameter>
❻      <BridgeParameter bridgeParameterIdentifier="Password"
         displayName="Password">
           pw123
       </BridgeParameter>
❼      <BridgeParameter bridgeParameterIdentifier="Schema"
         displayName="Schema"/>
❽      <BridgeParameter bridgeParameterIdentifier="Import stored
         procedures" displayName="Import stored procedures">
           False
       </BridgeParameter>
   </BridgeInformation>
```

Output 8. Modified Bridge Options File

The "Driver location" parameter (❶) is left blank because the required driver was copied to the default location indicated during the review of the bridge information. Values specific to the data source are provided for Host (❷), Service (❹), User (❺), and Password (❻). The default value of 1521 for Port (❸) is the correct value for the Oracle system in this example. The schema parameter (❼) can be used as a constraint to import a particular database schema by specifying a list of schemas separated by semicolons. All schemas for this connection are imported when no schema is provided. For the Import stored procedures option (❽), the default value of false remains.

STEP 4: LOAD RELATIONSHIPS

The bridge requirements have been reviewed and a completed bridge options file is ready. An optional step is to define equivalent relationships before the load relationships. Defining equivalent relationships enables the relationship loader to create relationships between imported schemas and tables with equivalent libraries and tables defined in SAS metadata. If you are defining equivalent relationships, review the information provided in this section without executing the load relationships. Then follow the instructions in the "Defining Equivalent Relationships" section.

This command loads the relationships using the bridge options file that you created earlier:

```
sas-metabridge-relationship-loader -bridgeIdentifier JdbcOracle
  -bridgeOptions "C:\temp\jdbcOptions.xml" -loadRelationships
  -log "C:\temp\loadRelationships.log"
```

Review the log for errors and possible remedies if you encounter problems while loading relationships. Connection and memory issues commonly cause problems.

Issues Connecting to the Third-Party Product

- Test the connection to the third-party product with an appropriate client on the same machine. Use the same connection information and drivers that specified in the bridge options.

- Review the required permissions and other information provided in the bridge information file referenced in Step 2: Review the Bridge Requirements. Verify that the correct permissions are set for the specified user.

- Verify that the correct drivers are available.

Memory Issues

If memory errors are indicated in the log, the Java heap size might need to be increased.

1. Make a backup copy of the [SASHome]\SASMetadataBridges\4.1\sas-metabridge-relationship-loader.ini file. Name it [SASHome]\SASMetadataBridges\4.1\sas-metabridge-relationship-loader.ini.**ORI**.

2. Edit the sas-metabridge-relationship-loader.ini file, and add this line to the end of the Java arguments in the file.

   ```
   JavaArgs_xx= -Xmx1024m
   ```

   Change **`xx`** to the next ordinal value in the JavaArgs sequence. For example, if the latest JavaArgs statement in the file is **`JavaArgs_8=...`**, then change **`xx`** to **9** in the above statement.

Understanding the Load Relationships Log

The log contains information about the results of the load relationships operation.

```
11:19:12 INFO  *** sas-metabridge-relationship-loader started on
                   Feb 22, 2016 ***
11:19:15 INFO  Server host01.example.com:7980, user sasadm@saspw.
11:19:19 INFO  Load relationships working directory is
               C:\Program Files\SASHome\SASMetadataBridges\4.1\tools\
                 relationshipLoadTemp.
```

Output 9. Partial View of the loadRelationships.log File Showing Initial Information

The log (Output 9) begins with the date the relationships were loaded followed by information used to connect to the SAS mid-tier. The working directory is the location where temporary files are written.

The log continues with information about the bridge and options used:

```
11:19:20 INFO   [INFO MIMB_I0028] Import bridge: 'JdbcOracle' 8.0.1 -
  2014-11-25 10:27:59
11:19:20 INFO   [INFO MIMB_I0031]     Driver location =
11:19:20 INFO   [INFO MIMB_I0031]     Host = host01.example.com

11:19:20 INFO   [INFO MIMB_I0031]     Port = 1521
11:19:20 INFO   [INFO MIMB_I0031]     Service = XE
11:19:20 INFO   [INFO MIMB_I0031]     User = dbmsadmin
11:19:20 INFO   [INFO MIMB_I0031]     Password = ******
11:19:20 INFO   [INFO MIMB_I0031]     Schema =
11:19:20 INFO   [INFO MIMB_I0031]     Import stored procedures = False
11:19:20 INFO   [STATUS BLIB_S0139] Loading jar files from
  'C:\Program Files\SASHome\SASMetadataBridges\4.1\java\Jdbc\oracle'
11:19:21 INFO   [INFO MBI_JDBC4_I0235] Loaded JDBC driver 'Oracle JDBC
  driver'. Version: '11.2.0.3.0' (major:11 minor:2)
11:19:21 INFO   [INFO MBI_JDBC4_I0201] Loading schemas
11:19:21 INFO   [STATUS MBI_JDBC4_S0221] Loading metadata from 'Oracle'.
```

Output 10. Partial View of the loadRelationships.log File Showing Bridge Options

Output 10 shows the bridge name and the connection options used to connect to the Oracle database. If the connection had failed, you would see a message indicating the issue. If the connection is successful but the table or view is not found, verify that the database permissions for the specified user match the permission specified in the bridge information from "Step 2: Review the Bridge Requirements". Examine the log for consistency warnings or errors that might impact the metadata.

```
11:20:15 INFO  [main] 2016-02-22 11:20:15 PRFLR_I0017 Loaded root model
                         in memory: xe.
11:20:15 INFO  [main] 2016-02-22 11:20:15 PRFLR_I0032 Using profile [Meta
                         Integration] for model [xe].
```

Output 11. Partial View of the loadRelationships.log File Showing the Profile

Output 11 shows the name of the profile used to determine the relationships from metadata. See "Choosing a Different Profile Definition" for information about using profiles to customize the relationships that are loaded.

Output 12 shows a sampling of the status messages written to the log:

```
12:40:19 INFO  Adding resource elements to all resources : 478
12:40:29 INFO  Adding lineage elements : 28
12:41:04 INFO  Adding delayed lineage relationships.
12:41:04 INFO  Setting owning object for 82260 objects ...4130 xe
```

Output 12. Partial View of the End of the loadRelationships.log File

The final status shows that the loader is setting an owning object for all imported resources and providing a count of the resources. This load of relationships imported 82,260 objects. The default settings import a lot of metadata and relationships. See "Advanced Topics" for strategies to control the metadata imported.

Defining Equivalent Relationships

The data flow for a business process is likely to include multiple sources that access the same physical data. The physical data is represented in the relationships database as a unique object associated with its source. For example, a physical table in an Oracle database is imported, and a resource object is created in the relationships database. Loading relationships from a SAS metadata repository might create another unique object representing the same Oracle table. These two unique objects that represent the same physical table must be identified as equivalent objects for lineage to be complete.

An equivalent relationships XML file specifies relationships between imported schemas and tables and the equivalent libraries and tables existing in the SAS metadata repository. Here is an example of an equivalent relationships file that maps an Oracle schema and three tables to the equivalent SAS library and tables:

```
     <EquivalentRelationships>

       <Equivalent type="Library">
❶        <ImportedResource name="ORSTAR"/>
❷        <Resource metadataPath="/Shared Data/Oracle/OrionStar(Library)"/>
       </Equivalent>

       <Equivalent type="Table">
❸        <ImportedResource name="ORSTAR.Customer_Dim"/>
❹        <Resource metadataPath="/Shared Data/Oracle/Customer_Dim(Table)"/>
       </Equivalent>

      <Equivalent type="Table">
         <ImportedResource name="ORSTAR.Time_Dim"/>
         <Resource metadataPath="/Shared Data/Oracle/Time_Dim(Table)"/>
       </Equivalent>

       <Equivalent type="Table">
         <ImportedResource name="ORSTAR.Product_Dim"/>
         <Resource metadataPath="/Shared Data/Oracle/Product_Dim(Table)"/>
       </Equivalent>

     </EquivalentRelationships>
```

Here are the steps to manually create the equivalent relationships file shown above:

1. Specify the external database schema name for the **name** attribute of the **ImportedResource** element (❶).

2. Specify at ❷ the metadata path, library name, and object type for the SAS metadata library that is equivalent to the third-party schema referenced in step 1. In this example, the path is **/Shared Data/Oracle/**, **OrionStar** is the library name, and the object type is **Library**.

3. Specify the third-party schema name (**ORSTAR**) and the table name (**Customer_Dim**) at ❸.

4. Specify at ❹ the metadata path, table name, and object type for the SAS table that is equivalent to the third-party table specified in Step 3. In this example, **/Shared Data/Oracle/** is the path, **Customer_Dim** is the table name, and the object type is **Table**.

Repeat Steps 3 and 4 for each equivalent table.

The equivalent relationships file is specified using the `-equivalentRelationships` option:

```
sas-metabridge-relationship-loader -bridgeIdentifier JdbcOracle
  -bridgeOptions "C:\temp\jdbcOptions.xml" -loadRelationships
  -equivalentRelationships "C:\temp\equivalentRelationships.xml"
```

The equivalent relationships file can be created using SAS code instead of manually entering the information for each table. Refer to the section "Example Code to Create Equivalent Relationships XML" for sample code that was used to create the equivalent relationships file above.

REMOVING THIRD-PARTY CONTENT USING SAS METADATA BRIDGE RELATIONSHIP LOADER

Sometimes it is necessary to remove all objects and relationships created during an import from the metadata bridge relationship loader. The `-clean` option extracts metadata from the source and then profiles the metadata to determine the owning object. All objects that exist in the relationship service associated with the owning object are deleted.

This example command calls `-clean`. For cleaning the same completed bridge options file that was used for load relationships is input with `-bridgeOptions`.

```
sas-metabridge-relationship-loader -bridgeIdentifier JdbcOracle
  -bridgeOptions "C:\temp\jdbcOptions.xml" -clean
```

PERFORMING LINEAGE AND IMPACT ANALYSIS

SAS provides tools to perform lineage and impact analysis, and these types of analyses have been described elsewhere (McIntosh 2014, SAS Institute Inc. 2015c, SAS Institute Inc. 2016). These tools were limited to analyzing SAS metadata prior to the third maintenance release for SAS 9.4 because the SAS Metadata Bridge Relationship Loader did not support third-party metadata. Now with all of the relationships supported loaded, you can use the same tools to perform analyses on the third-party metadata in addition to SAS metadata.

ADVANCED TOPICS

As previously mentioned, a lot of relationship metadata might be imported by the metadata bridge relationship loader. The volume of data can affect performance and make analyzing for lineage cumbersome. Customizations can be performed to control the objects and relationships imported and how the information is mapped to objects and relationship types in the relationship service database.

The metadata bridge relationship loader uses a configuration file, MIR_ObjectType.xml, to determine how objects map from the profiled metadata to resources in the relationships repository. A second configuration file, MIR_Associations.xml, manages which relationships are imported and the relationship type that is assigned to the imported relationship. The contents of the profiled metadata can be altered by choosing a different profile definition.

When changing a configuration file or switching profiles, it is recommended that you use the `-clean` option to remove previously loaded relationships before reloading the metadata.

MAPPING PROFILED METADATA TO SAS OBJECT TYPES

The configuration file that controls the mapping of object types is located in the directory [SASHome]/SASMetadataBridges/4.1/conf/SASMapping.

This file, MIR_ObjectType.xml, contains an element, *objectType*, for each object type mapping. An element exists for all types that exist in the profiled metadata.

```
<objectType>
❶     <name>Connection Data Set</name>
❷     <typeName>ConnectionDataSet.MIR</typeName>
❸     <type>4016</type>
</objectType>
```

Each object type mapping has three sub-elements: **name**, **typeName**, and **type**.

The **name** element (❶) specifies the name of the object. Do not edit this value because it must match the object type name that appears in the profiled metadata.

The **typeName** element (❷) specifies the SAS object type to which this object is mapped. A full listing of SAS object types can be found using the Object Type Service. See the section, "SAS Object Type Service" for an example of calling the Object Type Service REST interface for a list of valid object types. Profiled objects should be mapped to SAS object type names with .MIR appended.

The **type** element (❸) specifies the SAS object type identifier for the object type in the **typeName** element. The object type identifier is available from the Object Type Service.

To change the default mapping, change the values for both **typeName** and **type** elements to specify a different SAS object type. For example, the default setting maps a *Connection Data Set* from the MIR model to the SAS object type *ConnectionDataSet.MIR* with a *type* number of 4016.

To change the mapping to the *Table.MIR* SAS object type, change **typeName** to *Table.MIR* (❷) and **type** *to 4133* (❸).

```
<objectType>
      <name>Connection Data Set</name>
❶     <typeName>Table.MIR</typeName>
❷     <type>4133</type>
</objectType>
```

The type number was obtained by using the Object Type Service. (See the Appendix.)

MANAGING WHICH RELATIONSHIPS ARE IMPORTED AND THE RELATIONSHIP TYPE

The configuration file that manages which relationships are imported and the relationship type is located in the directory [SASHome]/SASMetadataBridges/4.1/conf/SASMapping.

The file MIR_Associatons.xml controls the relationships that are imported and the relationship type. There is one association element for each association in the profiled metadata. Here is an example showing the structure of the MIR_Associations.xml file:

```
<association>
❶     <name>RightQueryTable</name>
❷     <import>true</import>
❸     <relType>A</relType>
</association>
```

The association element has three sub-elements; **name**, **import**, and **relType**.

The **name** element (❶) is the name of the association in the MIR model. Do not edit this value. It must exactly match the association name as it appears in the MIR model.

The **import** element (❷) is either true or false. Use True to import this association. Use False to ignore this association.

The **relType** element (❸) defines the relationship type used during import for this association. The value is a one-character code referencing a relationship type. Valid values and the meaning are as follows:

- D - Is dependent on
- I - Contains
- P - Is parent of
- A - Is associated with
- E - Is equal to
- S - Is synonymous with

CHOOSING A DIFFERENT PROFILE DEFINITION

The metadata bridge relationship loader profiles the extracted metadata using a profile definition. These definitions are located in the directory installation-path/conf/MIRProfiles. The default profile used for all load relationships is *Meta Integration*. A default profile is used to ensure that, regardless of data source, the profiled metadata has the same format. Profiles for specific vendors will arrange metadata in a format and use terms specific to the product. A profile is specified using the `–bridgeProfile` option.

The command to load relationships from an ERwin file using the Data Modeling profile:

```
sas-metabridge-relationship-loader –bridgeIdentifier CaErwin7Xml
  –bridgeOptions "C:\temp\erwinOptions.xml" –loadRelationships
  -bridgeProfile "installation-path\conf\MIRProfiles\Data Modeling"
```

WHAT'S NEXT

SAS is continuing research and development in this area. You can expect to see support for more third-party products, tools to aid in stitching lineage metadata, and more customization options.

As we continue to build out the framework and see how the information stored in the relationship database is used, more features will be available. As always, your feedback is welcome and encouraged.

CONCLUSION

The first maintenance release of SAS 9.4 introduced the relationship service and database and reporting features that enable you to understand resources and how they are related. The third maintenance release of SAS 9.4 introduces a batch utility to load relationships from third-party sources to the relationship service. The addition of third-party metadata expands the capability to do data lineage and impact analysis on assets throughout the organization.

APPENDIX

SAS OBJECT TYPE SERVICE

The SAS Object Type Service is used to register object types for use in the SAS mid-tier. Object types are identified by a combination of name and numeric identifier. SAS Metadata Bridge Content Types are identified with a name ending with .MIR and have an identifier in the 4000 range.

To obtain a listing of all object types, log on to the SAS Web Server and send the following REST request.

```
http://host01.example.com/SASWIPClientAccess/rest/content/objectTypes
```

If no metadata bridge content types are loaded, install and configure the SAS Metadata Bridges Content Types on the mid-tier machine. See the installation and configuration guide for more information (SAS Institute Inc. 2015b).

EXAMPLE CODE TO CREATE EQUIVALENT RELATIONSHIPS XML

Here is example code that was used to create the equivalent relationships file shown in the "Defining Equivalent Relationships" section. This example assumes the following:

1. An Oracle database library is defined in the SAS metadata repository.
2. Metadata for the Oracle tables has been imported to the same SAS folder where the library is defined.
3. The table names in the SAS metadata repository match the table names in the Oracle database.

Gather the information listed in Table 1, and then use this information to complete the **User Input** section in the SAS code shown below.

Item	Sample Value	SAS Option or Macro Variable
Metadata server host name	host01.example.com	metaserver
Metadata server port number	8561	metaport
Metadata user name	sasadm@saspw	metauser
Metadata user password	Admin12345	metapass
Database schema name of 3rd-party resource	ORSTAR	IMPORTED_SCHEMA_NAME
SAS metadata library name of 3rd-party resource	OrionStar	IMPORTED_LIBRARY_NAME
SAS metadata library path	/Shared Data/Oracle/	IMPORTED_LIBRARY_PATH
Equivalent relationships file	C:\temp\EquivalentRelationships.xml	OUTPUT_FILE

Table 1. Information Used to Complete the Example SAS Code

Start SAS in UTF-8 mode before executing the code.

```
*  User Input;

options ls=max;

options metaserver = 'metadata-server-host-name'
        metaport   = metadata-server-port-number
        metauser   = 'metadata-user-name'
        metapass   = 'metadata-user-password';

%let IMPORTED_SCHEMA_NAME=third-party-schema-name;

%let IMPORTED_LIBRARY_NAME=third-party-SAS-library-name;

*  Include trailing slash;
%let IMPORTED_LIBRARY_PATH=sas-metadata-library-path/;

*  Do not use quotation marks;
%let OUTPUT_FILE=equivalent-relationships-file;
```

```
**********  Do not edit below this line  **********;

filename equivxml "&OUTPUT_FILE";

data _null_;

length i NTables rc sysrc 8;

length ErrorMessage $1024 LibraryURI $35 SASTableName $32
       sysmsg $1024 TablesURI $38;

*  Initialize all declared variables;

call missing(of _all_);

*;
*  Get the first (and only) occurrence of the imported library and
*  its URI.
*;

rc = metadata_getnobj("omsobj:SASLibrary?@Name eq
       '&IMPORTED_LIBRARY_NAME'",
                      1,
                      LibraryURI);
sysrc=sysrc(); sysmsg=sysmsg();
ErrorMessage='Problem encountered getting the imported library URI.';
link ERRORCHECK;

*;
*  Get the first Tables association of the imported library to
*  determine the number of tables.
*;

NTables = metadata_getnasn(LibraryURI,
                           'Tables',
                           1,
                           TablesURI);
sysrc=sysrc(); sysmsg=sysmsg();
ErrorMessage='Problem encountered getting the number of tables.';
link ERRORCHECK;

*;
*  Get each table and its name, and then create the
*  equivalent relationship.
*;

file equivxml;

do i = 1 to NTables;

  rc = metadata_getnasn(LibraryURI,
                        'Tables',
                        i,
                        TablesURI);
  sysrc=sysrc(); sysmsg=sysmsg();
  ErrorMessage='Problem encountered getting table for i=' ||
               strip(put(i, best.)) || '.';
  link ERRORCHECK;

  rc = metadata_getattr(TablesURI,
                        'SASTableName',
                        SASTableName);
  sysrc=sysrc(); sysmsg=sysmsg();
  ErrorMessage='Problem encountered getting the SAS table name ' ||
               'for i=' || strip(put(i, best.)) || '.';
  link ERRORCHECK;

  if (i eq 1) then do;
    put '<EquivalentRelationships>';
    put;
    put '  <Equivalent type="Library">';
    put '    <ImportedResource name="' "&IMPORTED_SCHEMA_NAME" '"/>';
    put '    <Resource metadataPath="'
        "&IMPORTED_LIBRARY_PATH.&IMPORTED_LIBRARY_NAME"
        '(Library)"/>';
    put '  </Equivalent>';
    put;
  end;

  put '  <Equivalent type="Table">';
  put '    <ImportedResource name="' "&IMPORTED_SCHEMA_NAME" '.'
      SASTableName +(-1) '"/>';
  put '    <Resource metadataPath="' "&IMPORTED_LIBRARY_PATH"
      SASTableName +(-1) '(Table)"/>';
  put '  </Equivalent>';
  put;

  if (i eq NTables) then do;
    put '</EquivalentRelationships>';
  end;

end;

STOP;

ERRORCHECK:

if (sysrc ne 0 or compress(sysmsg) ne '') then do;
  file log;
  put 'ERROR: ' ErrorMessage sysrc= sysmsg=;
  STOP;
end;

run;

;*';*";*/;quit;run;
```

REFERENCES

McIntosh, Liz, et. al. 2014. "Understanding Change in the Enterprise." *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. Available at https://support.sas.com/resources/papers/proceedings14/SAS396-2014.pdf

SAS Institute Inc. 2015a. *SAS 9.4 for Microsoft Windows and Windows for x64 Planned Deployment*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/installcenter/en/ikwinplannedri/66607/PDF/default/setinit_planned.pdf

SAS Institute Inc. 2015b. *SAS 9.4 Intelligence Platform: Installation and Configuration Guide, Second Edition*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/biig/69172/HTML/default/viewer.htm#titlepage.htm

SAS Institute Inc. 2015c. "Using the Relationship Reporter (sas-relationship-reporter)." *SAS 9.4 Intelligence Platform: System Administration Guide, Fourth Edition*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/bisag/68240/HTML/default/viewer.htm#n02xw5d6g8m7v7n1shgdj8pf7vkf.htm

SAS Institute Inc. 2016. "SAS Lineage." Available at http://support.sas.com/software/products/dmlin/

RECOMMENDED READING

Rausch, Nancy. 2016. "What's New in SAS Data Management." *Proceedings of the SAS Global Forum 2016 Conference*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings16/index.html.

SAS Institute Inc. 2015. *SAS 9.4 Intelligence Platform: System Administration Guide, Fourth Edition*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/bisag/68240/HTML/default/viewer.htm#titlepage.htm

Stander, Jeff. 2016. "Enterprise Data Governance across SAS and Beyond." *Proceeding of the SAS Global Forum 2016 Conference.* Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings16/index.html

ACKNOWLEDGMENTS

The author would like to thank Vince DelGobbo of SAS Institute Inc. for his valuable contributions to this paper.

CONTACT INFORMATION

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

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