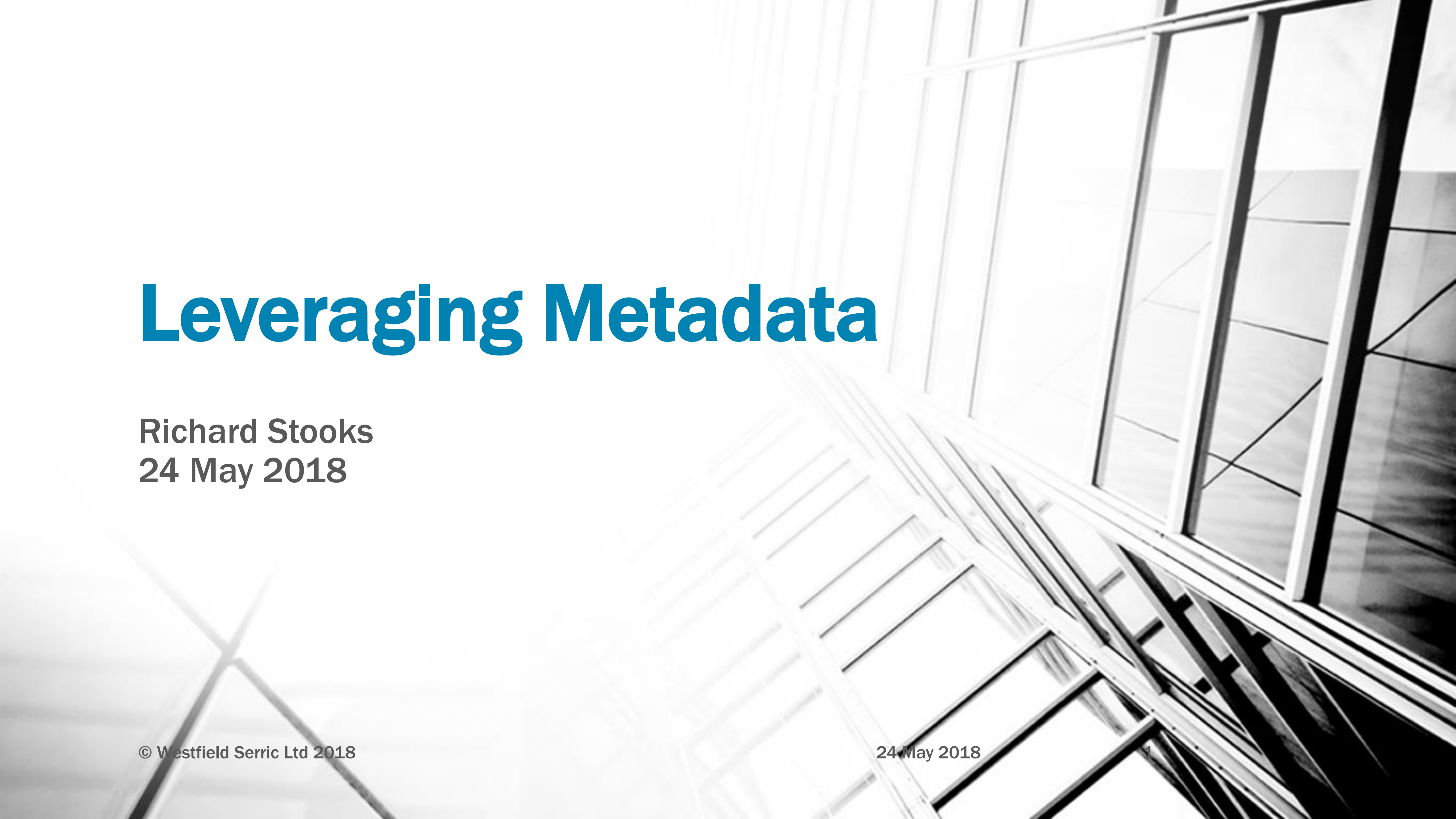


Leveraging Metadata

Richard Stooks
24 May 2018

Covering…

Who Am I?

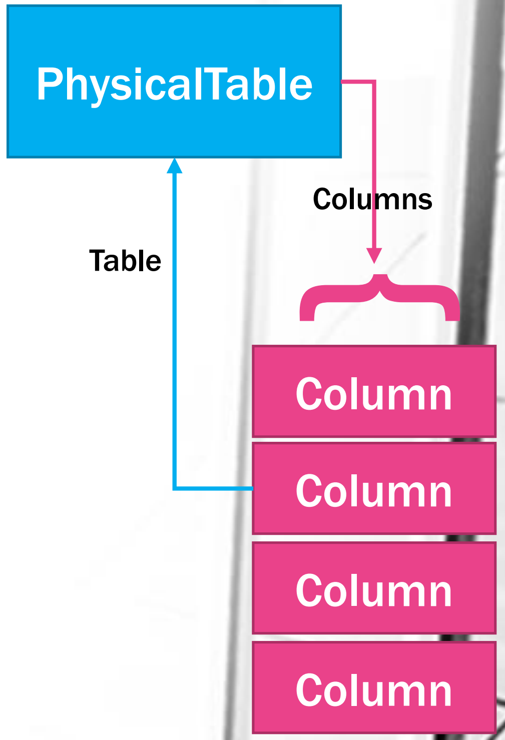
- Richard Stooks
- Work in Financial Services now
- Used Most SAS versions 82.4 to 9.4
  - First project was a Timesheeting system in SAS/AF
  - Now mainly BI/Web – HTML5
  - Backend Data Gathering, Assimilation and Reporting
- Gained interest in Metadata
  - Fixing broken metadata in DI Studio Exported jobs.
  - Latterly, to take away the drudgery of documenting production data flows.

What is SAS Metadata?

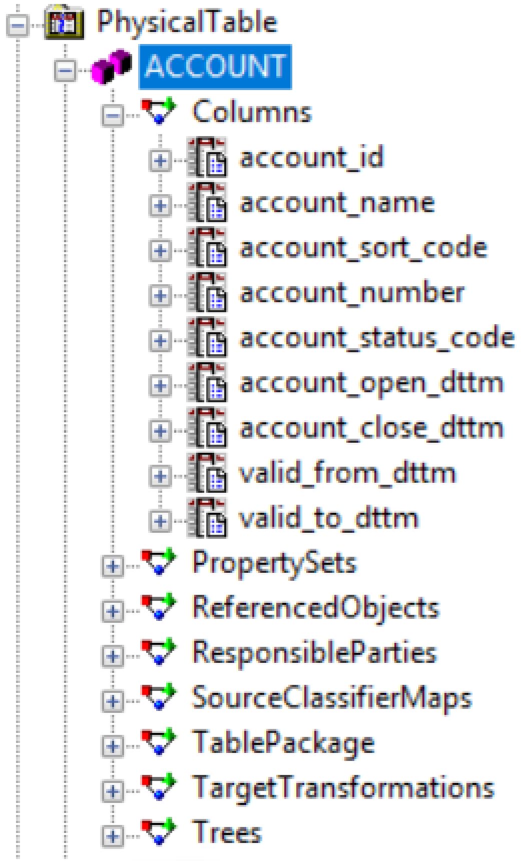
- SAS Metadata holds a complete description of the physical and logical SAS environment
  - the servers,
  - the users,
  - the “jobs”,
  - other things besides.
- Each "thing" is described in one or more metadata **objects**.
- Each metadata object has a **type** and metadata objects are related to each other by **associations**.
- Each object also has **attributes**.
  - The one that uniquely identifies each object is its Id.
  - Objects can have many attributes, but most, if not all, have a Name as well.

Concept: Metadata Objects and Associations

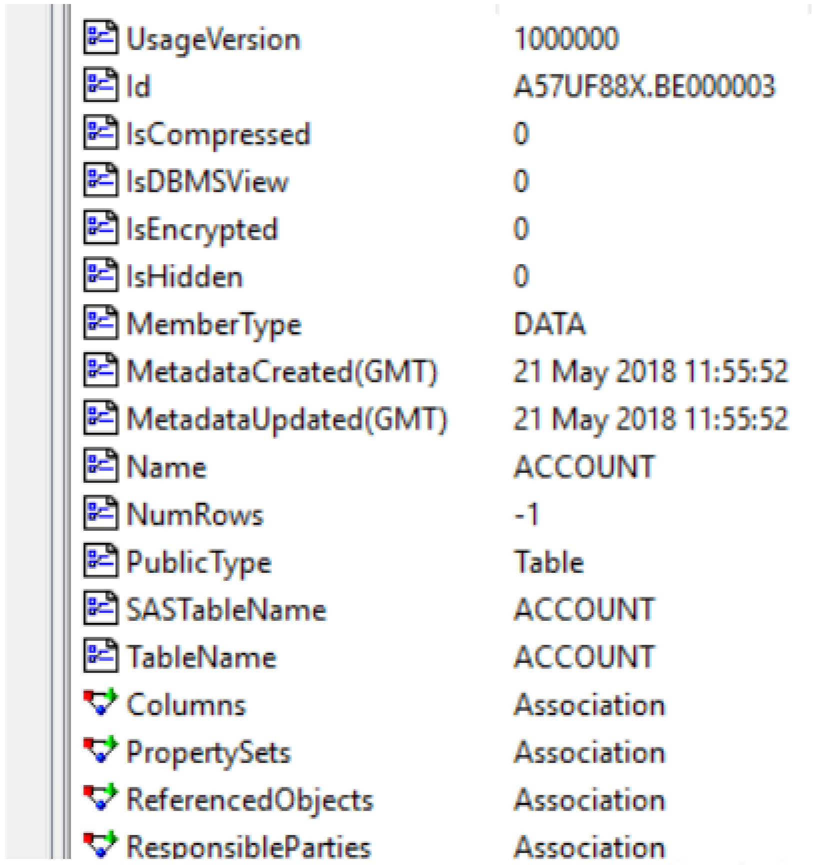
- A PhysicalTable object represents (describes, not is) a table.
  - As well as a Name, a PhysicalTable object has a SASTableName attribute that is the name of the table as known to SAS programs
- The PhysicalTable will, of course, have columns:
  - these are represented by Column objects and the PhysicalTable is linked to them by the Columns *association*. An individual Column object, on the other hand is linked to the table in which it is defined by the Table *association*.
- A PhysicalTable will be stored in a library.
  - To represent this, the PhysicalTable is linked to the SASLibrary object by the TablePackage *association*. Inversely, the SASLibrary is linked to all the tables in it by the Tables *association*.
  - Different objects come into play for third party databases, such as Oracle



A Physical Table Metadata Object and Associations



```
PhysicalTable
  ACCOUNT
    Columns
      account_id
      account_name
      account_sort_code
      account_number
      account_status_code
      account_open_dttm
      account_close_dttm
      valid_from_dttm
      valid_to_dttm
    PropertySets
    ReferencedObjects
    ResponsibleParties
    SourceClassifierMaps
    TablePackage
    TargetTransformations
    Trees
```



UsageVersion	1000000
Id	A57UF88X.BE000003
IsCompressed	0
IsDBMSView	0
IsEncrypted	0
IsHidden	0
MemberType	DATA
MetadataCreated(GMT)	21 May 2018 11:55:52
MetadataUpdated(GMT)	21 May 2018 11:55:52
Name	ACCOUNT
NumRows	-1
PublicType	Table
SASTableName	ACCOUNT
TableName	ACCOUNT
Columns	Association
PropertySets	Association
ReferencedObjects	Association
ResponsibleParties	Association

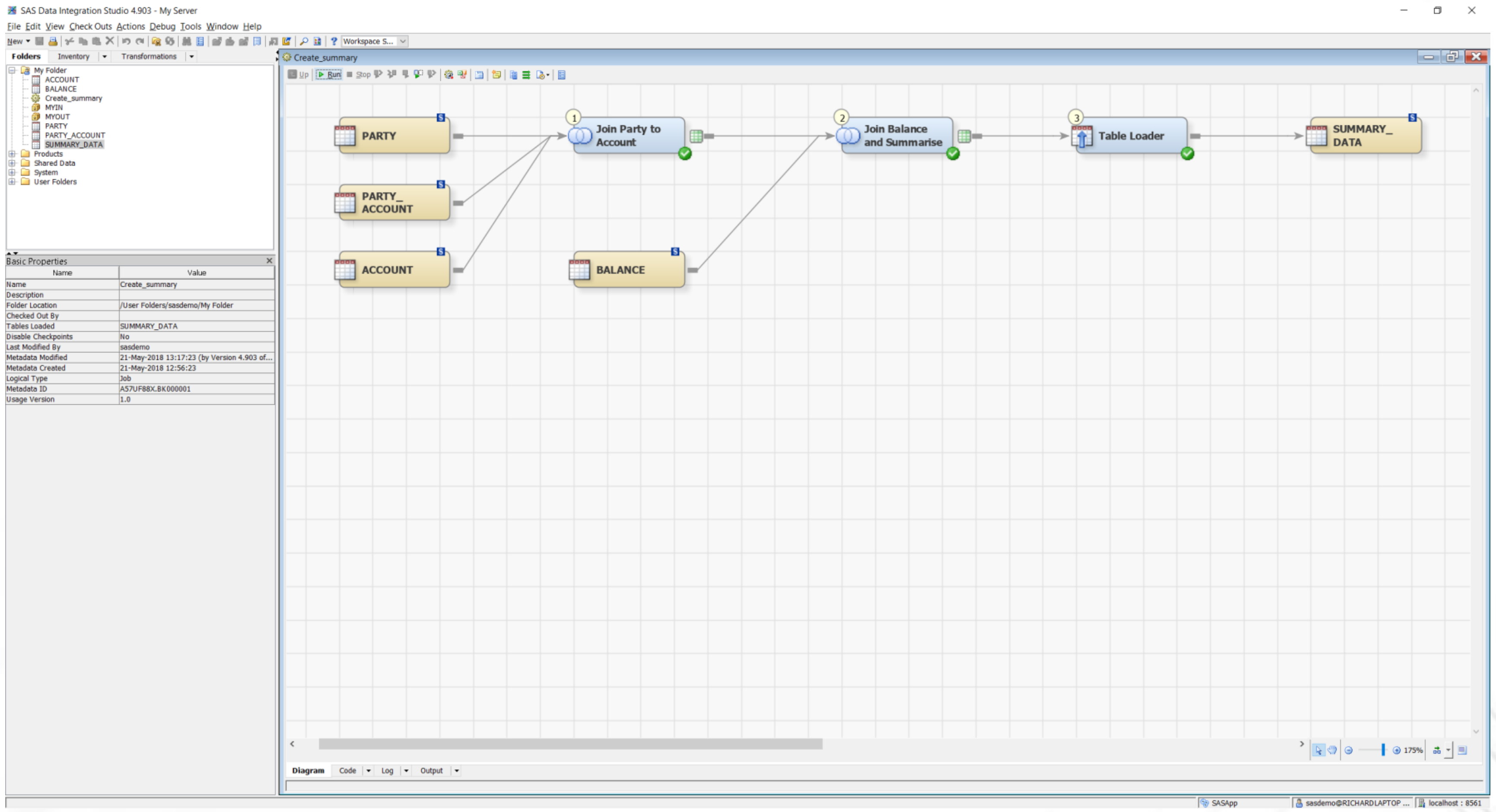
Concept: The URI

- Uniform Resource Identifier – used to identify objects
- Metadata ID – eg A57UF88X.BQ000001 or simply BQ000001
- Type/ID – eg PhysicalTable/A57UF88X.BE000003
- Search String – eg PhysicalTable?@Name='ACCOUNT'
- Can be prefixed with omsobj: - eg omsobj:PhysicalTable?@Name='ACCOUNT'
- Search strings can contain the *association path* – more later

Example Data Tables

- PARTY – information about an individual
- ACCOUNT – Information about a bank account
- PARTY_ACCOUNT – Links each party to each of their accounts
- BALANCE – the daily account balance table
- SUMMARY_DATA – a desired monthly summary at party-account level of basic financial information (balances)

The DI Studio SAS Script – The “Job”

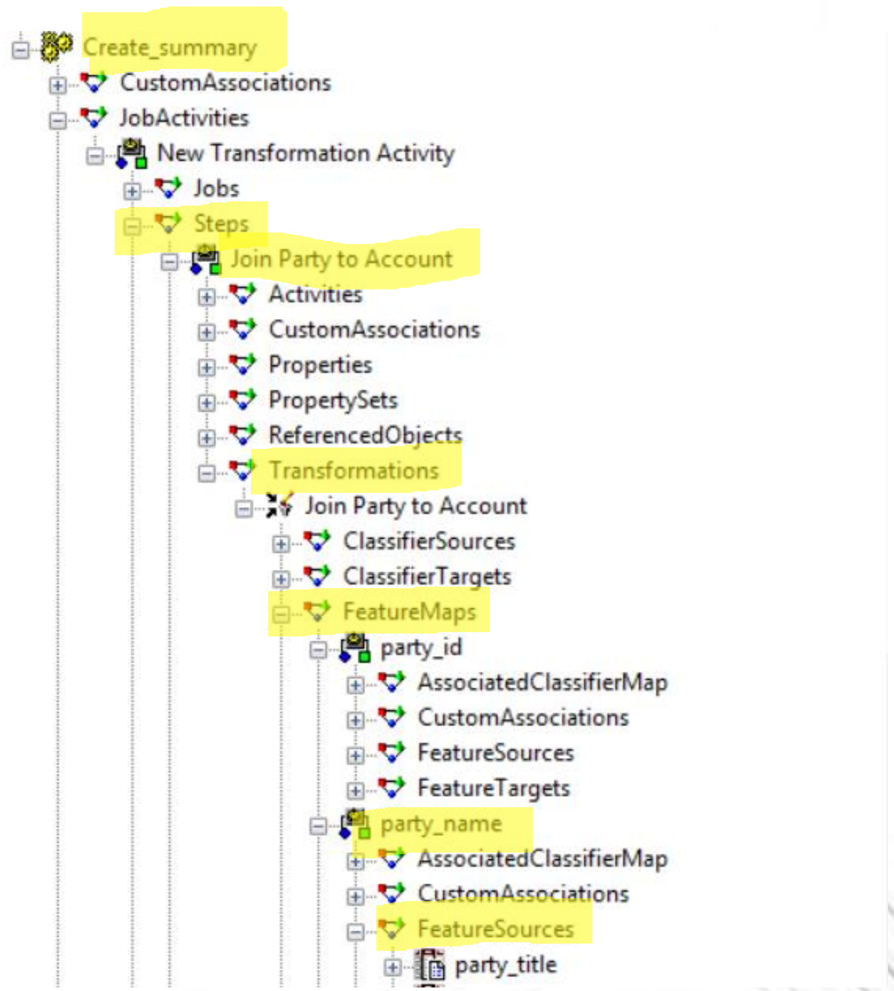


Basic Properties

Name	Value
Name	Create_summary
Description	
Folder Location	/User Folders/sasdemo/My Folder
Checked Out By	
Tables Loaded	SUMMARY_DATA
Disable Checkpoints	No
Last Modified By	sasdemo
Metadata Modified	21-May-2018 13:17:23 (by Version 4.903 of...
Metadata Created	21-May-2018 12:56:23
Logical Type	Job
Metadata ID	A57UF88X.BK000001
Usage Version	1.0

The Job Metadata

- DI Studio creates metadata objects and associations to store the script definition.
- Interrogating metadata allows us to ask questions about the job



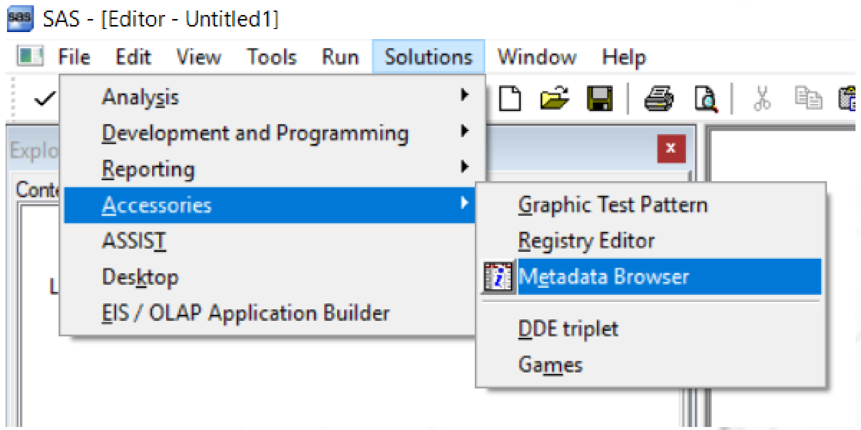
```
Create_summary
  CustomAssociations
  JobActivities
    New Transformation Activity
      Jobs
      Steps
        Join Party to Account
          Activities
          CustomAssociations
          Properties
          PropertySets
          ReferencedObjects
          Transformations
            Join Party to Account
              ClassifierSources
              ClassifierTargets
              FeatureMaps
                party_id
                  AssociatedClassifierMap
                  CustomAssociations
                  FeatureSources
                  FeatureTargets
                party_name
                  AssociatedClassifierMap
                  CustomAssociations
                  FeatureSources
                    party_title
```

What Questions?

- Where has my data come from and *how has it been manipulated*?
- What processes are affected if I make this change to that table?
- Are there any examples of jobs that do this?

Accessing Metadata

- SAS Metadata browser
  - Complete and easy to use, but hard to find everything easily
- SAS DI Studio
  - Visually good, one job at a time
  - Harder to see column mappings
  - No searching
  - “Analyse” limited to whole tables or one column at a time
- SAS XML Queries
  - Complete solution.
  - Quite hard to understand – need (user defined) XMLMAPS to help out
- SAS Datastep Queries
  - Enable step by step expansion (working out!) of queries
  - Quite hard to start with



Useful Data Step Metadata Functions

- rc=metadata_getnobj(i_uri,n,o_uri)
  - Gets the *n*th object that matches the *i_uri* specification and returns the *o_uri* for the associated object
- rc=metadata_getnasn(i_uri,asn_name,n,o_uri)
  - Gets the *n*th object associated by *asn_name* with the *i_uri* specification and returns the *o_uri* for it
- rc=metadata_getattr(i_uri,attr_name,value)
  - Gets the *value* of the attribute *attr_name* for *i_uri*
- rc=metadata_getnasl(i_uri,n,name,)
  - Gets the *n*th association *name* for *i_uri*
- rc=metadata_getnatr(i_uri,n,name,value)
  - Gets the *n*th attribute for *i_uri* and returns the *name* and *value*

Simple Query – "Names of the Jobs that have a parallel loop" – Step 1

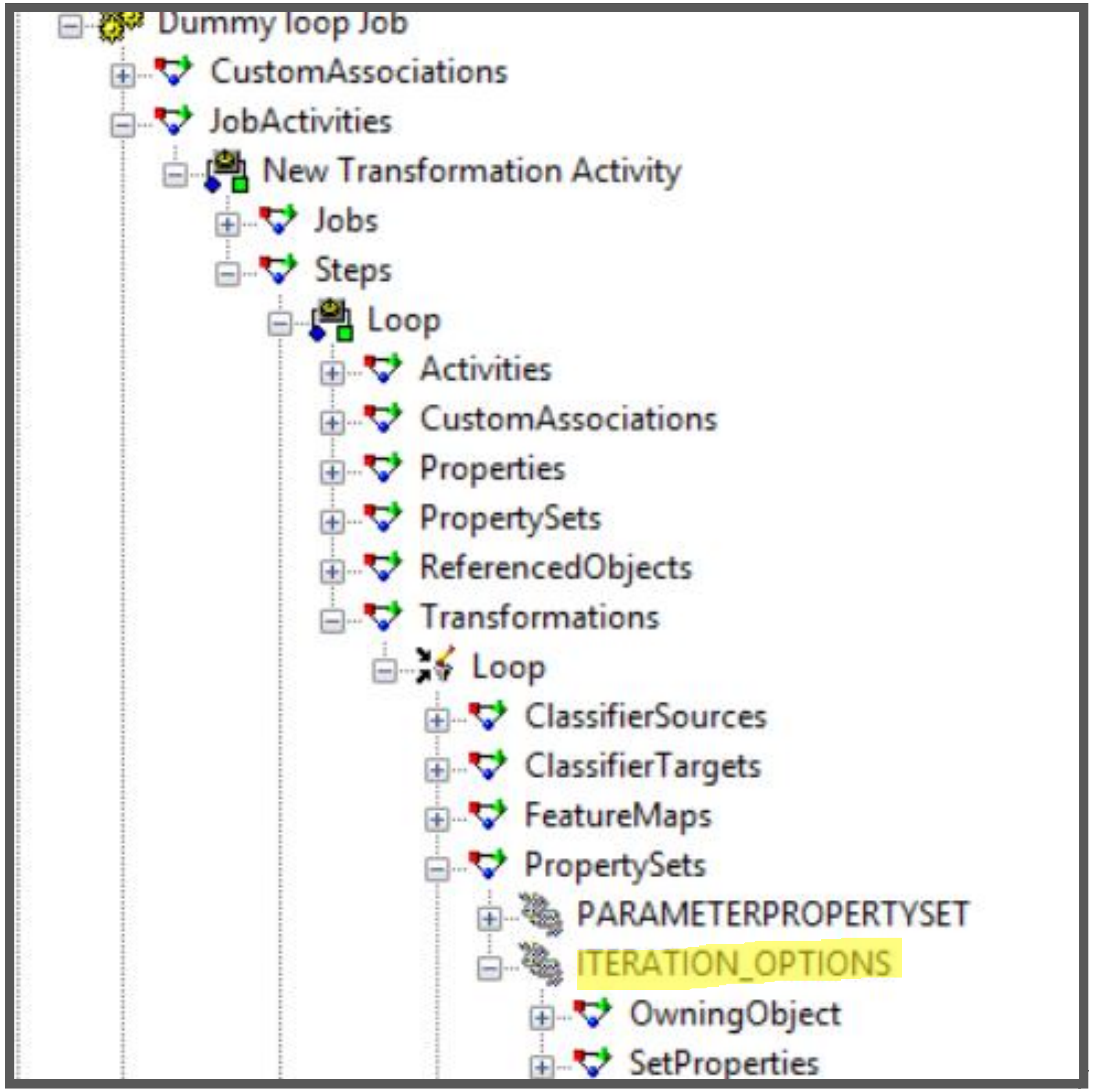
```
Data _null_;

length uri $256;

call missing(of _all_);

numProps=metadata_getnobj("omsobj:Property?*[@Name='EXECUTEPARALLEL']",1,uri);

Put numProps=;

run;
```

Log:

```
numProps=1
```

How ? – The Metadata Browser

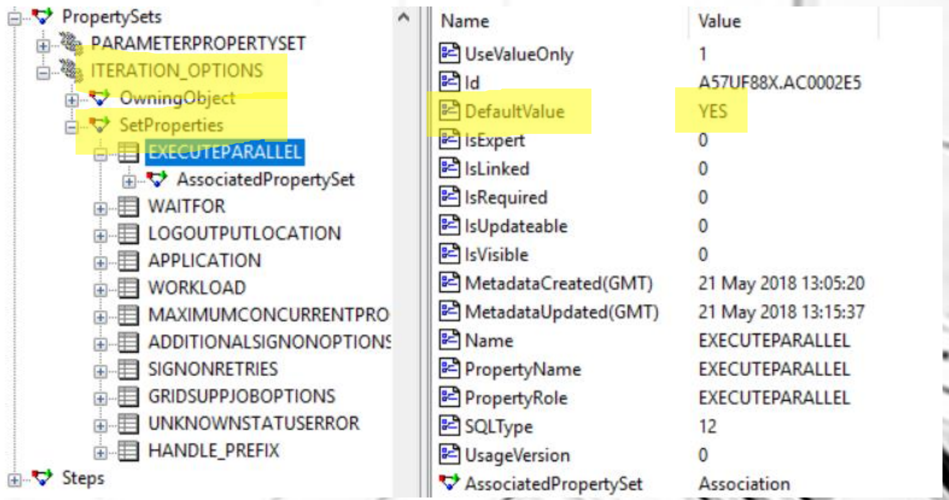
- Create a dummy job
- The required option MUST be stored somewhere
- Look for it
- Understand it
- Use it



```
Dummy loop Job
  CustomAssociations
  JobActivities
    New Transformation Activity
      Jobs
      Steps
        Loop
          Activities
          CustomAssociations
          Properties
          PropertySets
          ReferencedObjects
          Transformations
            Loop
              ClassifierSources
              ClassifierTargets
              FeatureMaps
              PropertySets
                PARAMETERPROPERTYSET
                ITERATION_OPTIONS
                  OwningObject
                  SetProperties
```

Simple Query – “Names of the Jobs that have a parallel loop” – Step 2 – Expand It

```
Data _null_;
length uri ps_uri cm_uri ts_uri a_uri j_uri value $256;
call missing(of _all_);
numProps=metadata_getnobj("omsobj:Property?*[@Name='EXECUTEPARALLEL']",1,uri);
do propNum=1 to numProps;
     numProps=metadata_getnobj("omsobj:Property?*[@Name='EXECUTEPARALLEL']",propNum,uri);
     rc=metadata_getattr(uri,"DefaultValue",value);
     if value="YES" then link get_yes;
     end;
stop;
```



```
PropertySets
  PARAMETERPROPERTYSET
  ITERATION_OPTIONS
    OwningObject
    SetProperties
      EXECUTEPARALLEL
        AssociatedPropertySet
      WAITFOR
      LOGOUTPUTLOCATION
      APPLICATION
      WORKLOAD
      MAXIMUMCONCURRENTPRO
      ADDITIONALSIGNONOPTIONS
      SIGNONRETRIES
      GRIDSUPPJOBOPTIONS
      UNKNOWNSTATUSERROR
      HANDLE_PREFIX
Steps
```

Name	Value
UseValueOnly	1
Id	A57UF88X.AC0002E5
DefaultValue	YES
IsExpert	0
IsLinked	0
IsRequired	0
IsUpdateable	0
IsVisible	0
MetadataCreated(GMT)	21 May 2018 13:05:20
MetadataUpdated(GMT)	21 May 2018 13:15:37
Name	EXECUTEPARALLEL
PropertyName	EXECUTEPARALLEL
PropertyRole	EXECUTEPARALLEL
SQLType	12
UsageVersion	0
AssociatedPropertySet	Association

Simple Query – “Names of the Jobs that have a parallel loop” – Step 2 – Expand It

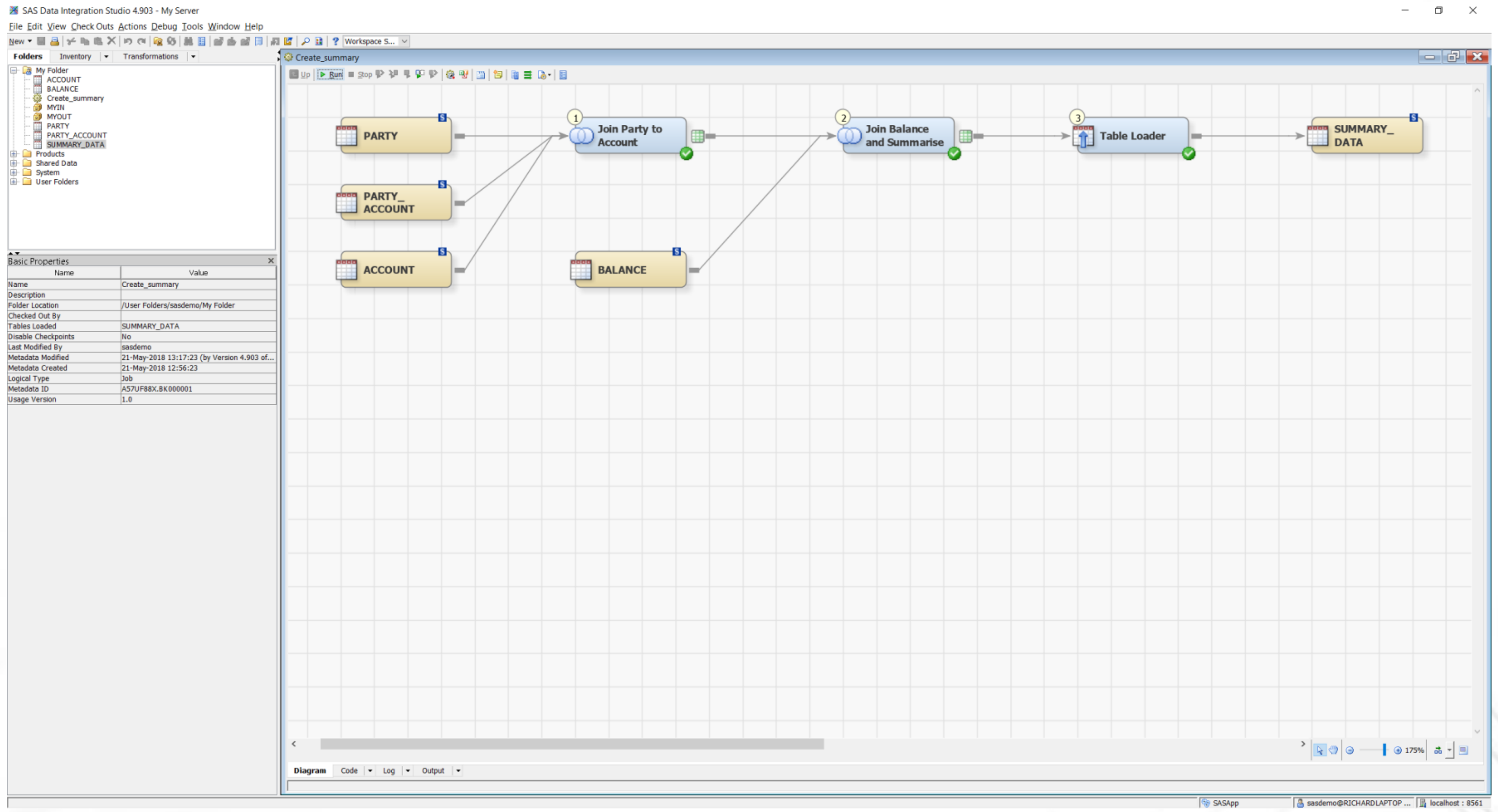
```
get_yes:

numPropertySets=metadata_getnasn(uri,"AssociatedPropertySet",1,ps_uri);

numMaps=metadata_getnasn(ps_uri,"OwningObject",1,cm_uri);

numSteps=metadata_getnasn(cm_uri,"Steps",1,ts_uri);

numActivities=metadata_getnasn(ts_uri,"Activities",1,a_uri);

numJobs=metadata_getnasn(a_uri,"Jobs",1,j_uri);

rc=metadata_getattr(j_uri,"Name",value);

put value;

return;

run;
```

Simple Query – "Names of the Jobs that have a parallel loop" – Step 3 – Simplify

```
data _null_;

length uri value $256;

call missing(of _all_);

i=1;

Do
while(metadata_getnobj("omsobj:Job?*[JobActivities/TransformationActivity/Steps/TransformationSte
p/Transformations/ClassifierMap/PropertySets/PropertySet[@Name='ITERATION_OPTIONS']/SetProperties
/Property[@Name='EXECUTEPARALLEL' AND @DefaultValue='YES']]",i,uri)>0);

     rc=metadata_getattr(uri,"Name",value);

     put value;

     i+1;

     end;

run;
```

More Complex Question – Mapping Data

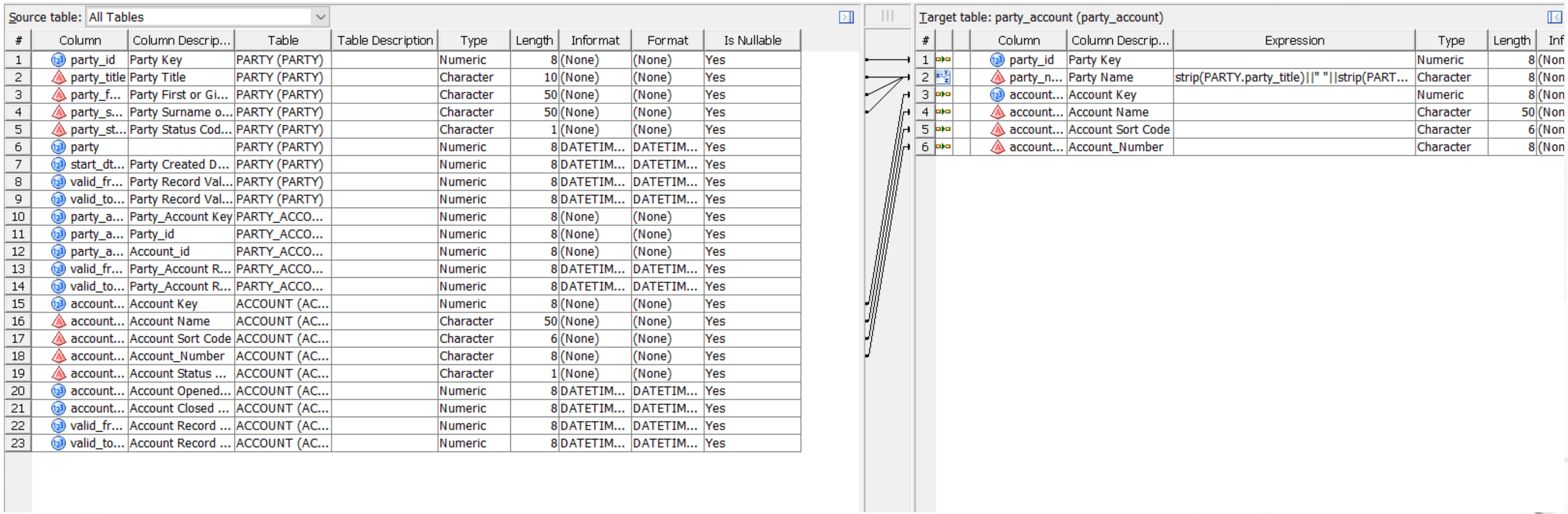
Reminder - The Job



Basic Properties

Name	Value
Name	Create_summary
Description	
Folder Location	/User Folders/sasdemo/My Folder
Checked Out By	
Tables Loaded	SUMMARY_DATA
Disable Checkpoints	No
Last Modified By	sasdemo
Metadata Modified	21-May-2018 13:17:23 (by Version 4.903 of...
Metadata Created	21-May-2018 12:56:23
Logical Type	Job
Metadata ID	A57UF88X.BK000001
Usage Version	1.0

Step 1 Mapping Within The Job



Source table: All Tables

#	Column	Column Descrip...	Table	Table Description	Type	Length	Informat	Format	Is Nullable
1	party_id	Party Key	PARTY (PARTY)		Numeric	8	(None)	(None)	Yes
2	party_title	Party Title	PARTY (PARTY)		Character	10	(None)	(None)	Yes
3	party_f...	Party First or Gi...	PARTY (PARTY)		Character	50	(None)	(None)	Yes
4	party_s...	Party Surname o...	PARTY (PARTY)		Character	50	(None)	(None)	Yes
5	party_st...	Party Status Cod...	PARTY (PARTY)		Character	1	(None)	(None)	Yes
6	party		PARTY (PARTY)		Numeric	8	DATETIM...	DATETIM...	Yes
7	start_dt...	Party Created D...	PARTY (PARTY)		Numeric	8	DATETIM...	DATETIM...	Yes
8	valid_fr...	Party Record Val...	PARTY (PARTY)		Numeric	8	DATETIM...	DATETIM...	Yes
9	valid_to...	Party Record Val...	PARTY (PARTY)		Numeric	8	DATETIM...	DATETIM...	Yes
10	party_a...	Party_Account Key	PARTY_ACCO...		Numeric	8	(None)	(None)	Yes
11	party_a...	Party_id	PARTY_ACCO...		Numeric	8	(None)	(None)	Yes
12	party_a...	Account_id	PARTY_ACCO...		Numeric	8	(None)	(None)	Yes
13	valid_fr...	Party_Account R...	PARTY_ACCO...		Numeric	8	DATETIM...	DATETIM...	Yes
14	valid_to...	Party_Account R...	PARTY_ACCO...		Numeric	8	DATETIM...	DATETIM...	Yes
15	account...	Account Key	ACCOUNT (AC...		Numeric	8	(None)	(None)	Yes
16	account...	Account Name	ACCOUNT (AC...		Character	50	(None)	(None)	Yes
17	account...	Account Sort Code	ACCOUNT (AC...		Character	6	(None)	(None)	Yes
18	account...	Account_Number	ACCOUNT (AC...		Character	8	(None)	(None)	Yes
19	account...	Account Status ...	ACCOUNT (AC...		Character	1	(None)	(None)	Yes
20	account...	Account Opened...	ACCOUNT (AC...		Numeric	8	DATETIM...	DATETIM...	Yes
21	account...	Account Closed ...	ACCOUNT (AC...		Numeric	8	DATETIM...	DATETIM...	Yes
22	valid_fr...	Account Record ...	ACCOUNT (AC...		Numeric	8	DATETIM...	DATETIM...	Yes
23	valid_to...	Account Record ...	ACCOUNT (AC...		Numeric	8	DATETIM...	DATETIM...	Yes

Target table: party_account (party_account)

#	Column	Column Descrip...	Expression	Type	Length	Inf
1	party_id	Party Key		Numeric	8	(Non
2	party_n...	Party Name	strip(PARTY.party_title)||" "||strip(PART...	Character	8	(Non
3	account...	Account Key		Numeric	8	(Non
4	account...	Account Name		Character	50	(Non
5	account...	Account Sort Code		Character	6	(Non
6	account...	Account_Number		Character	8	(Non

Viewing The Mappings Programmatically

- Mappings are described in a FeatureMap object – one for each OUTPUT column
- Associations From FeatureMap
  - FeatureSources: the source column object(s)
  - FeatureTargets: the target column object

Simple 1:1 Mapping

```
data _null_;
length feature_uri col_uri source_name target_name transformation_rule
$256;
call missing(of _all_);
feature_uri="A57UF88X.BQ000001"; /* PARTY_ID Mapping in Step 1 */
rc=metadata_getattr(feature_uri,"TransformRole",transformation_rule);
if transformation_rule="ONETOONE" then do;
        rc=metadata_getnasn(feature_uri,"FeatureSources",1,col_uri);
        rc=metadata_getattr(col_uri,"Name",source_name);
        rc=metadata_getnasn(feature_uri,"FeatureTargets",1,col_uri);
        rc=metadata_getattr(col_uri,"Name",target_name);
        end;

put source_name= target_name=;
run;
```

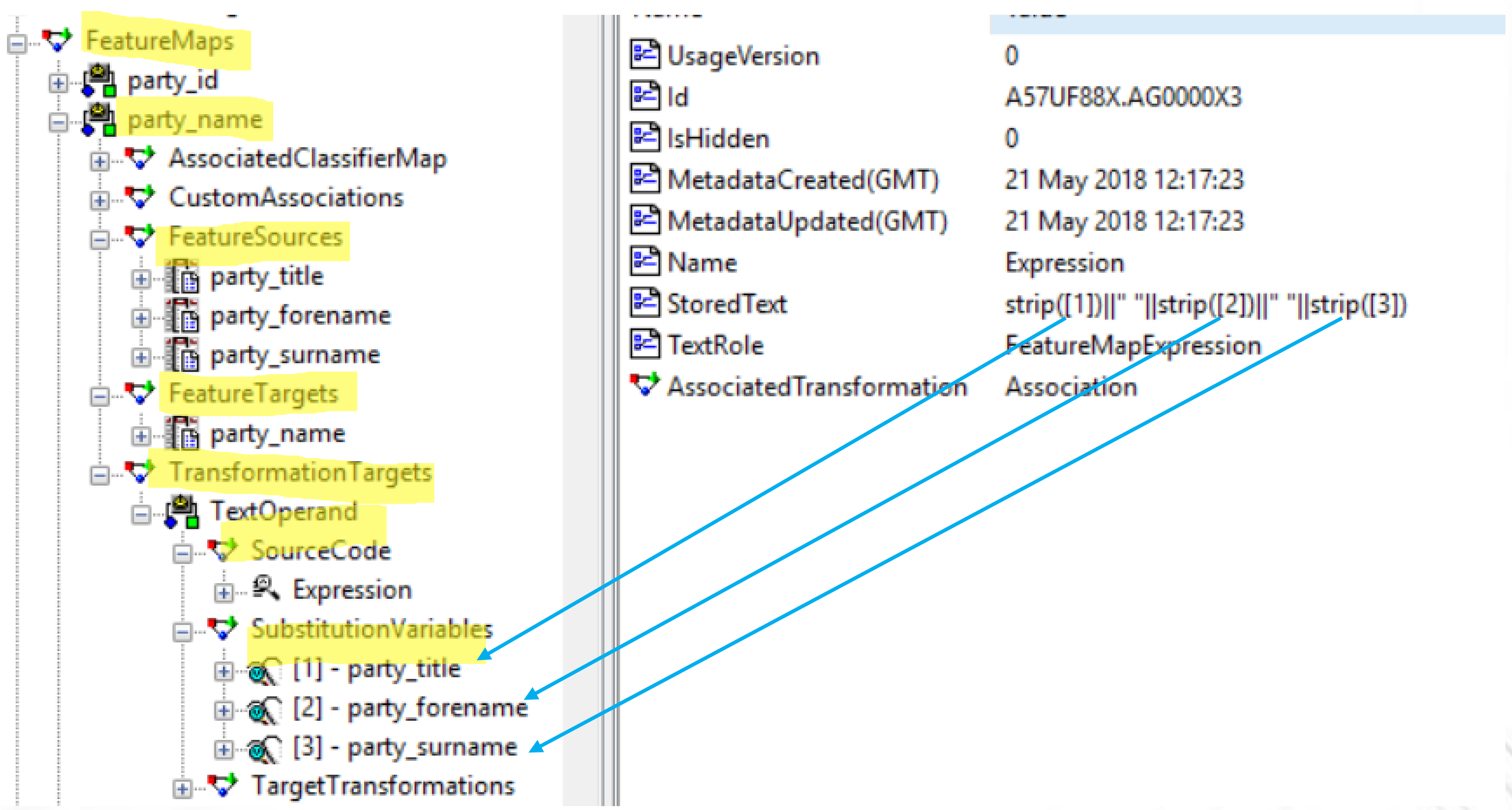
Log:

```
source_name=party_id target_name=party_id
```

Dealing With Expressions

- Associations From FeatureMap
  - FeatureSources: the source column object(s)
  - FeatureTargets: the target column object
  - TransformationTargets: any Expression used
    - SourceCode: the actual code
    - SubstitutionVariables the list of variable names (not directly column objects) used in the expression

Feature Map for party_name



- FeatureMaps
  - party_id
  - party_name
    - AssociatedClassifierMap
    - CustomAssociations
    - FeatureSources
      - party_title
      - party_forename
      - party_surname
    - FeatureTargets
      - party_name
    - TransformationTargets
      - TextOperand
        - SourceCode
          - Expression
        - SubstitutionVariables
          - [1] - party_title
          - [2] - party_forename
          - [3] - party_surname
        - TargetTransformations

UsageVersion	0
Id	A57UF88X.AG0000X3
IsHidden	0
MetadataCreated(GMT)	21 May 2018 12:17:23
MetadataUpdated(GMT)	21 May 2018 12:17:23
Name	Expression
StoredText	strip([1])||" "||strip([2])||" "||strip([3])
TextRole	FeatureMapExpression
AssociatedTransformation	Association

Chaining through Mappings

- A Column Object that is the FeatureTarget of one FeatureMap
  - Is a FeatureSource of others
  - Until the last time it is used by a DI Job (when you assume it is the ultimate target)
- A Column Object that is a FeatureSource of a FeatureMap
  - Is a FeatureTarget of another FeatureMap
  - Until the first time it is used by a DI Job (when you assume it is the RAW data source)
- Providing All Mappings have been completed accurately
  - Caveat only applies to user written code
- So, from any given start point, you can chain through the FeatureMaps in either direction

Chaining from sources to targets:

```
data _null_;
length feature_uri col_uri table_uri table column $256;
call missing(of _all_);
lev=1;
col_uri="A57UF88X.BG00000B"; /* ACCOUNT_NAME in ACCOUNT table */
rc=1;
do until(rc<=0);
          link get_table_col_info;
          rc=metadata_getnasn(col_uri,"SourceFeatureMaps",1,feature_uri);
          if rc > 0 then rc=metadata_getnasn(feature_uri,"FeatureTargets",1,col_uri);
          end;
stop;

get_table_col_info:
rc2=metadata_getnasn(col_uri,"Table",1,table_uri);
rc2=metadata_getattr(col_uri,"SASColumnName",column);
rc2=metadata_getattr(table_uri,"SASTableName",table);
put lev= table= column=;
lev+1;
return;
run;
```

```
LOG:
lev=1 table=ACCOUNT column=account_name
lev=2 table=party_account column=account_name
lev=3 table=summarised column=account_name
lev=4 table=SUMMARY_DATA column=account_name
```

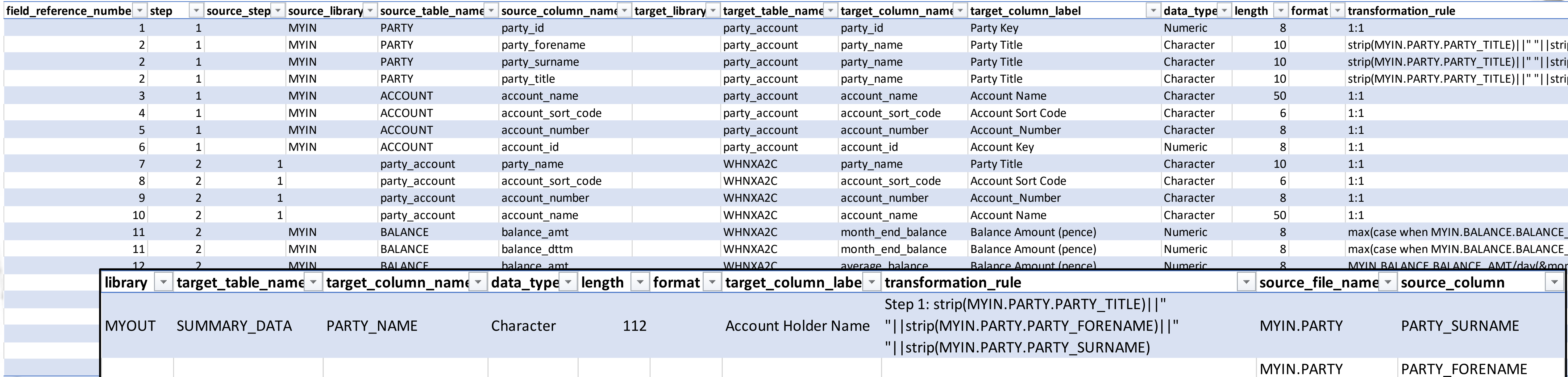
Chaining from targets to sources:

```
data _null_;
length feature_uri col_uri table_uri table column $256;
call missing(of _all_);
lev=1;
col_uri="A57UF88X.BG000006"; /* ACCOUNT_NAME in SUMMARY_DATA table */
rc=1;
do until(rc<=0);
          link get_table_col_info;
          rc=metadata_getnasn(col_uri,"TargetFeatureMaps",1,feature_uri);
          if rc > 0 then numSources=metadata_getnasn(feature_uri,"FeatureSources",1,col_uri);
          /* More complexity needed */
          end;
stop;
get_table_col_info:
rc2=metadata_getnasn(col_uri,"Table",1,table_uri);
rc2=metadata_getattr(col_uri,"SASColumnName",column);
rc2=metadata_getattr(table_uri,"SASTableName",table);
put lev= table= column=;
lev+1;
return;
run;
```

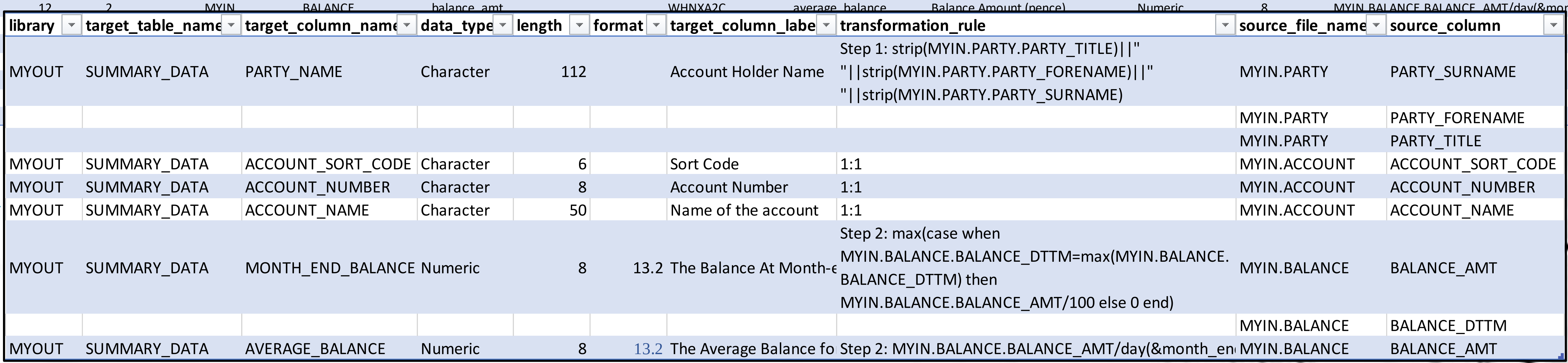
Writing Documentation

- Limited only by imagination and time:
  - EVERYTHING about ALL the code is stored in metadata
- Can present results in XML, HTML, Excel
- Can concatenate text (such as expressions) to form complex descriptions of transformations.
- Can combine information from multiple, consecutive pieces of code
- Can check for “not to standard” code

Job Mapping Outputs



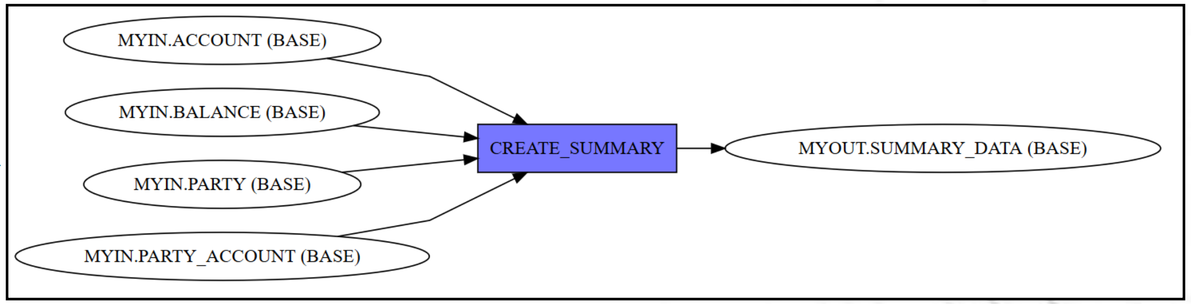
field_reference_numbe	step	source_step	source_library	source_table_name	source_column_name	target_library	target_table_name	target_column_name	target_column_label	data_type	length	format	transformation_rule
1	1		MYIN	PARTY	party_id		party_account	party_id	Party Key	Numeric	8		1:1
2	1		MYIN	PARTY	party_forename		party_account	party_name	Party Title	Character	10		strip(MYIN.PARTY.PARTY_TITLE)||" "||stri
2	1		MYIN	PARTY	party_surname		party_account	party_name	Party Title	Character	10		strip(MYIN.PARTY.PARTY_TITLE)||" "||stri
2	1		MYIN	PARTY	party_title		party_account	party_name	Party Title	Character	10		strip(MYIN.PARTY.PARTY_TITLE)||" "||stri
3	1		MYIN	ACCOUNT	account_name		party_account	account_name	Account Name	Character	50		1:1
4	1		MYIN	ACCOUNT	account_sort_code		party_account	account_sort_code	Account Sort Code	Character	6		1:1
5	1		MYIN	ACCOUNT	account_number		party_account	account_number	Account_Number	Character	8		1:1
6	1		MYIN	ACCOUNT	account_id		party_account	account_id	Account Key	Numeric	8		1:1
7	2	1		party_account	party_name		WHNXA2C	party_name	Party Title	Character	10		1:1
8	2	1		party_account	account_sort_code		WHNXA2C	account_sort_code	Account Sort Code	Character	6		1:1
9	2	1		party_account	account_number		WHNXA2C	account_number	Account_Number	Character	8		1:1
10	2	1		party_account	account_name		WHNXA2C	account_name	Account Name	Character	50		1:1
11	2		MYIN	BALANCE	balance_amt		WHNXA2C	month_end_balance	Balance Amount (pence)	Numeric	8		max(case when MYIN.BALANCE.BALANCE_
11	2		MYIN	BALANCE	balance_dttm		WHNXA2C	month_end_balance	Balance Amount (pence)	Numeric	8		max(case when MYIN.BALANCE.BALANCE_
12	2		MYIN	BALANCE	balance_amt		WHNXA2C	average_balance	Balance Amount (pence)	Numeric	8		MYIN.BALANCE.BALANCE_AMT/day(&mor



library	target_table_name	target_column_name	data_type	length	format	target_column_labe	transformation_rule	source_file_name	source_column
MYOUT	SUMMARY_DATA	PARTY_NAME	Character	112		Account Holder Name	Step 1: strip(MYIN.PARTY.PARTY_TITLE)||" "||strip(MYIN.PARTY.PARTY_FORENAME)||" "||strip(MYIN.PARTY.PARTY_SURNAME)	MYIN.PARTY	PARTY_SURNAME
								MYIN.PARTY	PARTY_FORENAME
								MYIN.PARTY	PARTY_TITLE
MYOUT	SUMMARY_DATA	ACCOUNT_SORT_CODE	Character	6		Sort Code	1:1	MYIN.ACCOUNT	ACCOUNT_SORT_CODE
MYOUT	SUMMARY_DATA	ACCOUNT_NUMBER	Character	8		Account Number	1:1	MYIN.ACCOUNT	ACCOUNT_NUMBER
MYOUT	SUMMARY_DATA	ACCOUNT_NAME	Character	50		Name of the account	1:1	MYIN.ACCOUNT	ACCOUNT_NAME
MYOUT	SUMMARY_DATA	MONTH_END_BALANCE	Numeric	8	13.2	The Balance At Month-e	Step 2: max(case when MYIN.BALANCE.BALANCE_DTTM=max(MYIN.BALANCE.BALANCE_DTTM) then MYIN.BALANCE.BALANCE_AMT/100 else 0 end)	MYIN.BALANCE	BALANCE_AMT
								MYIN.BALANCE	BALANCE_DTTM
MYOUT	SUMMARY_DATA	AVERAGE_BALANCE	Numeric	8	13.2	The Average Balance fo	Step 2: MYIN.BALANCE.BALANCE_AMT/day(&month_en	MYIN.BALANCE	BALANCE_AMT

Create JSON data:

```
digraph finite_state_machine {
rankdir=LR;
outputorder=node_name;
fontsize=6;
splines=ortho
Obj_0 [label="MYIN.ACCOUNT (BASE)" color=black style="filled"
fillcolor="White" shape=ellipse ];
Obj_1 [label="CREATE_SUMMARY" color=black style="filled"
fillcolor="#7777FF" shape=rectangle ];
Obj_2 [label="MYIN.BALANCE (BASE)" color=black style="filled"
fillcolor="White" shape=ellipse ];
Obj_3 [label="MYIN.PARTY (BASE)" color=black style="filled"
fillcolor="White" shape=ellipse ];
Obj_4 [label="MYIN.PARTY_ACCOUNT (BASE)" color=black
style="filled" fillcolor="White" shape=ellipse ];
Obj_5 [label="MYOUT.SUMMARY_DATA (BASE)" color=black
style="filled" fillcolor="White" shape=ellipse ];
Obj_0 -> Obj_1 ;
Obj_2 -> Obj_1 ;
Obj_3 -> Obj_1 ;
Obj_4 -> Obj_1 ;
Obj_1 -> Obj_5 ;
}
```



http://www.webgraphviz.com

Some Pitfalls and Difficulties

- Nested Jobs
- Subqueries
- The Splitter transform
- Calculated field references
- Case Statements
- User Written Code
- People who don’t write code the same way you do…

Questions

SAS 9.4 Metadata Documentation

- Metadata Model:
  https://support.sas.com/documentation/cdl/en/omamodref/67417/HTML/default/viewer.htm#titlepage.htm
- Open Metadata Reference
  http://documentation.sas.com/?docsetId=omaref&docsetTarget=titlepage.htm&docsetVersion=9.4&locale=en
- Language Interfaces to Metadata
  http://documentation.sas.com/?docsetId=lrmeta&docsetTarget=titlepage.htm&docsetVersion=9.4&locale=en