

SAS Data Quality Monitoring workshop exercises

Paul Van Mol

Contents

1. Read Breakfast_Items and Manufacturers with an import	3
2. Add 2 discovery Agents as an SAS Viya Administrator for SASDM Compute and Public Caslib.	4
3. Go to Discover Information Assets as Alex and search for the Breakfast_Items table.	6
4. Create Ruleset to check Validity of UPC (Uniform Product Code) in Breakfast_Items:	8
5. Create Ruleset to check Accuracy of UOM (Unit of Measure).....	11
6. Create a Monitoring Task: Mon_Breakfast_items	14
7. Appendix.....	16
Prepare Demo on Training Virtuallab	16

1. Read Breakfast_Items and Manufacturers with an import

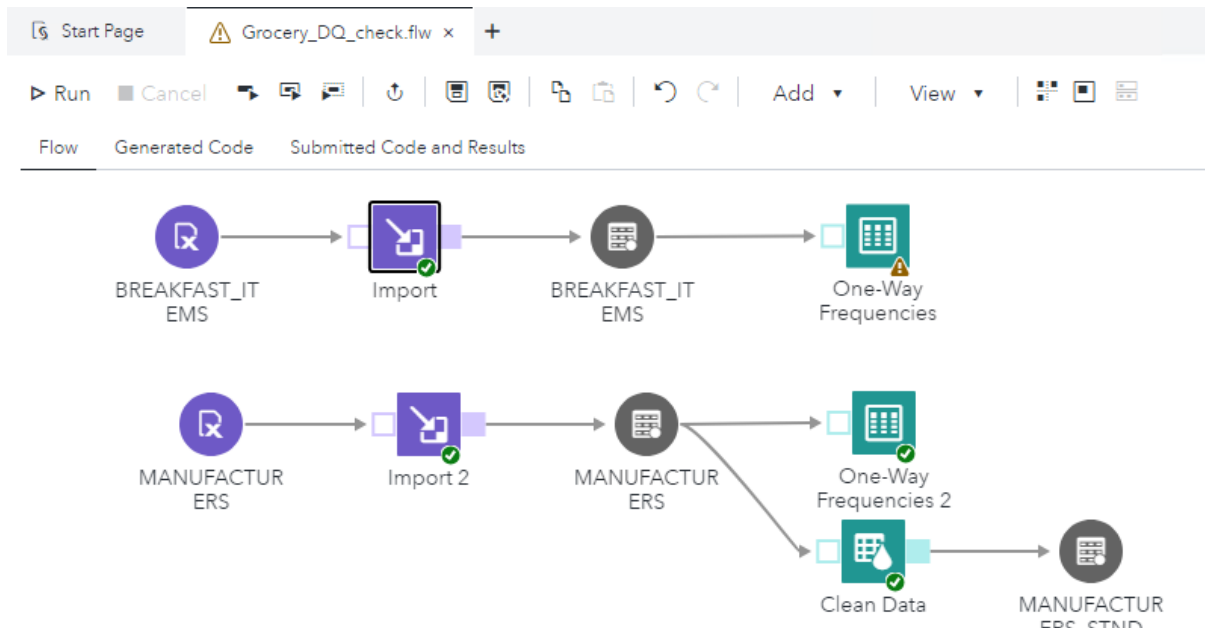
Navigate to the Folder: /gelcontent/dqdashboard/data/

Add the BREAKFAST_ITEMS.xlsx, then add an import step.

Analyze the columns and store the result in a table SASDM.BREAKFAST_ITEMS

Add the MANUFACTURERS.xlsx, then add an import step.

Analyze the columns and store the result in a table SASDM.MANUFACTURERS




In order to check the values in BREAKFAST_ITEMS: UPC, UOM and BRAND, add a One-Way Frequencies Step to the Breakfast_Items table:

One-Way Frequencies

Data	Options	Node	Notes
Filter input data:			
<input type="text"/>			
Analysis variables: *			
<input type="checkbox"/> <input checked="" type="checkbox"/> UPC			
<input type="checkbox"/> <input checked="" type="checkbox"/> UOM			
<input type="checkbox"/> <input checked="" type="checkbox"/> BRAND			

In options unselect include cumulative frequencies and percentages:


In order to check the values in MANUFACTURERS fields: COUNTRY, STATE_PROV, CONTACT_CNTRY, CONTACT_STATE_PROV add a One-Way Frequencies Step to the MANUFACTURERS table:


 One-Way Frequencies 2


Data Options Node Notes


Filter input data:

Analysis variables: *

☐  COUNTRY


☐  STATE_PROV

☐  CONTACT_STATE_PROV

☐  CONTACT_CNTRY

Finally, Standardize the COUNTRY and STATE_PROV fields in the MANUFACTURERS table:


Using a Clean Data step:

 Clean Data


< QKB Locale Standardization Casing Gender Analysis Identification Analysis

You can perform up to 10 standardization operations.

Select column: *

 COUNTRY

Definition: *

Country (ISO 3 Char) 

Column options:

☐ Replace existing column

☒ Create new column

New column:

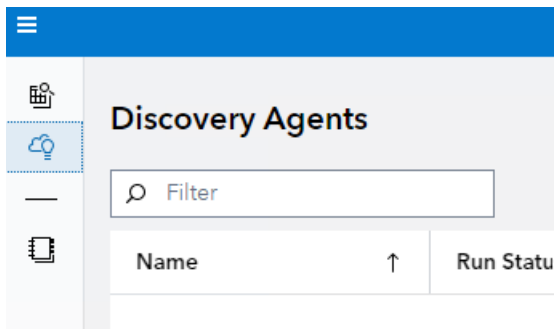
COUNTRY_STND

2. Add 2 discovery Agents as an SAS Viya Administrator for SASDM Compute and Public Caslib.

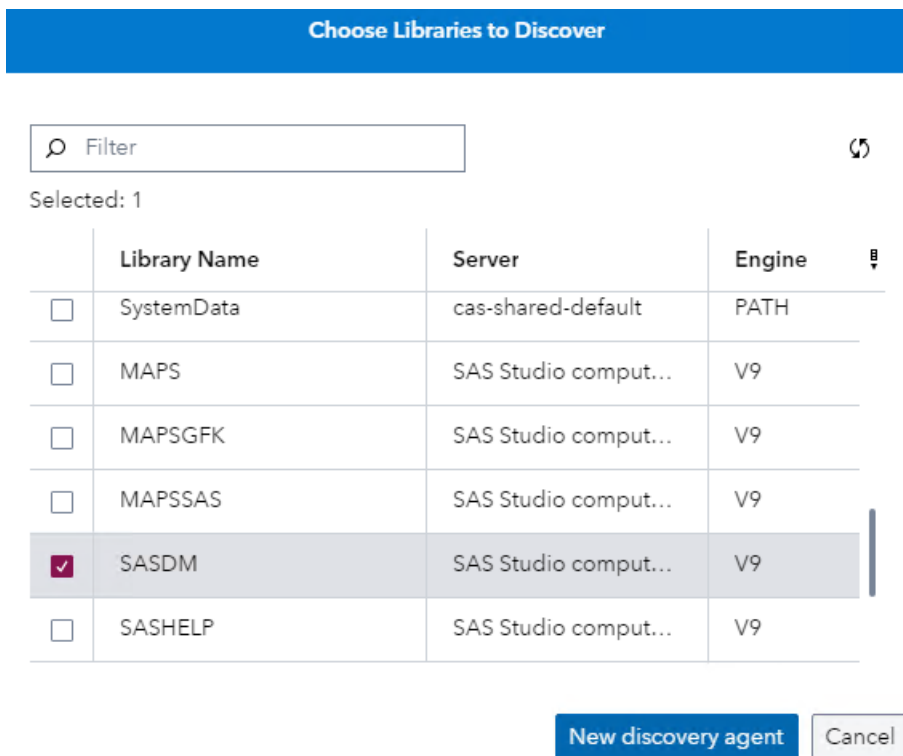
Login to SAS Viya as sasadm & Inxsas.

Go to Discover Information Assets

Select Discovery Agents



Select new Discovery Agent and choose the compute library SASDM.



No changes are required in the configuration:

Select Run Now. This will save the discovery agent and run the Discovery Agent.

Same can be done for the PUBLIC Caslib.

3. Go to Discover Information Assets as Alex and search for the Breakfast_Items table.

Search for the Breakfast_Items table:

Catalog Home > Search Results								breakfast
Search indexes: (no filter)								
Top 1 Results								Open details
	Name	★	Status	Date Analyzed	Asset Type	Date Modified	Modified By	Date Created
<input type="checkbox"/>	BREAKFAST_ITEMS	☆		Dec 10, 2024 4:38 AM	SAS table	Dec 10, 2024 4:01 AM	--	Dec 10, 2024 4:01 AM

Select the Table and look at the Overview page:

BREAKFAST_ITEMS
SASDM

Completeness: 89%

Columns: 15 Rows: 2.6 K Size: 512 KB

Status: None

Date analyzed: Dec 10, 2024 4:38 AM

Overview Column Analysis Sample Data

Private Information Privacy

Jan 1, 1960 - Dec 30, 2009

Time Period Covered

(none found)

Top Areas Covered

Summary

This dataset describes information about the following entities: **organization, individual**. The most important columns are **SIZE, HEIGHT, and WIDTH**. The storage format is **SAS**. The frequency of observations is **monthly** from date column **DATE**. The data was collected between **January 1, 1960 and December 30, 2009**. The data contains outliers and has values that could be considered **private**.

Description

The description is not available.

Name/Label	Length	Semantic Type	Information Privacy	Terms
ID ID	8	Generic ID	None	(none)
BRAND BRAND	16	Organization	None	(none)
MANUFACTURER_ID MANUFACTURER_ID	8	ORGANIZATION_ID	None	(none)
UPC UPC	15	(none)	None	(none)
LEGACYUPC LEGACYUPC	14	(none)	None	(none)
NAME NAME	32	Individual	Private	(none)
SIZE SIZE	8	(none)	None	(none)
UOM UOM	2	(none)	None	(none)
ITEM_ID ITEM_ID	13	Generic ID	None	(none)

Contacts (0)
No contacts are assigned.

Tags (0)
No tags are assigned.

Properties

Asset type: SAS table
Date modified: Dec 10, 2024 4:01 AM
Modified by: --
Date created: Dec 10, 2024 4:01 AM
Created by: UNKNOWN
Library: SASDM
Encoding: utf-8 Unicode (UTF-8)
Label: --

Select column analysis:

Investigate the columns UPC, UOM and BRAND.

- What is the completeness of each field?
- For UPC and UOM what is the most common Pattern?

Close the Breakfast_items table

Now search for the Manufacturers table:

Catalog Home > Search Results > MANUFACTURERS

MANUFACTURERS
SASDM

Completeness: 85%

Columns: 17 Rows: 199 Size: 128 KB

Status: None

Date analyzed: Dec 10, 2024 4:38 AM

Overview Column Analysis Sample Data

Private Information Privacy

Jan 1, 1960 - Dec 27, 2009

Time Period Covered

USA, US, U.S., United St...

Top Areas Covered

Summary

This dataset describes information about the following entities: **organization, phone**. This dataset contains information from multiple **country, delivery address, and city** locations. The storage format is **SAS**. The frequency of observations is **daily** from date column **POSTDATE**. The data was collected between **January 1, 1960 and December 27, 2009**. The data has values that could be considered **private**.

Description

The description is not available.

Name/Label	Length	Semantic Type	Information Privacy	Terms
ID ID	8	Generic ID	None	(none)
MANUFACTURER MANUFACTURER	50	Organization	None	(none)
STREET_ADDR STREET_ADDR	46	Delivery address	Candidate	(none)
CITY CITY	17	City	Candidate	(none)
STATE_PROV STATE_PROV	10	State/province	Candidate	(none)
POSTAL_CD POSTAL_CD	10	Postal code	Candidate	(none)
COUNTRY COUNTRY	14	Country	Candidate	(none)
PHONE PHONE	20	Phone	Private	(none)
CONTACT CONTACT	22	(none)	None	(none)

Contacts (0)
No contacts are assigned.

Tags (0)
No tags are assigned.

Properties

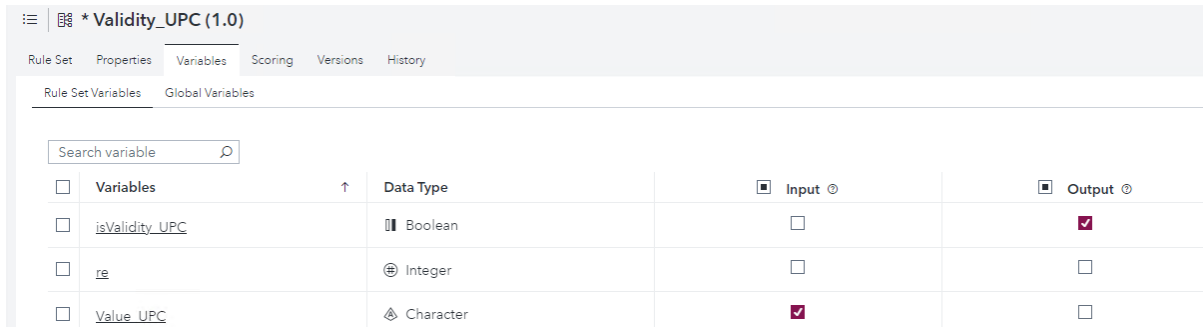
Asset type: SAS table
Date modified: Dec 10, 2024 4:01 AM
Modified by: --
Date created: Dec 10, 2024 4:01 AM
Created by: UNKNOWN
Library: SASDM
Encoding: utf-8 Unicode (UTF-8)
Label: --

In the column analysis, investigate the completeness and Frequency Distribution of different fields:

What is the completeness of Street_addr, STATE_PROV, COUNTRY

4. Create Ruleset to check Validity of UPC (Uniform Product Code) in Breakfast_Items:

1. create Validity_UOM Ruleset:
 - Duplicate the Ruleset Validity_Phone and save as Validity_UOM
 - Rename Variables isAccuracy_UOM (Boolean), Value_UOM (string)



Variables	Data Type	Input	Output
<input type="checkbox"/> isValidity_UPC	Boolean	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/> re	Integer	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> Value_UPC	Character	<input checked="" type="checkbox"/>	<input type="checkbox"/>

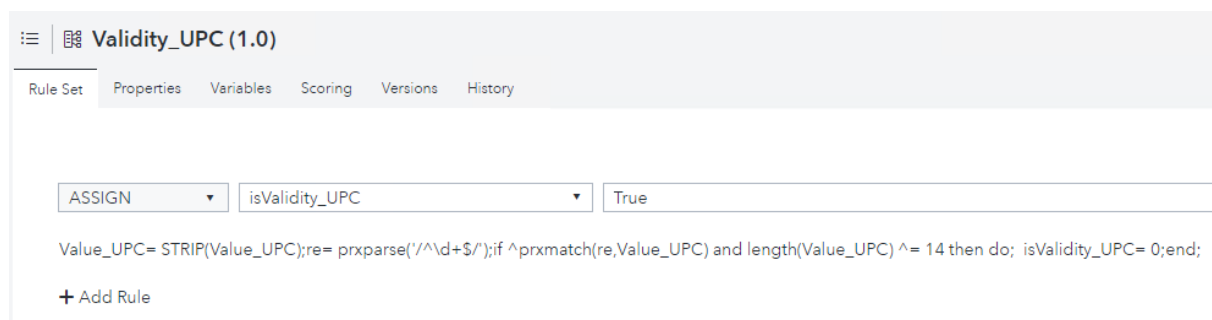
On the Rule Set tab:

- Rule: Validity_UPC
- Dimension: Validity
- Rule_Description: Checking type and length of UPC
- Expression:

Assign: isValidity_UPC True

Change the custom expression to the expression below. It uses a Regex to check if not numbers are found in the Value_UPC and if the length is not 14 long.

```
Value_UPC= STRIP(Value_UPC)
re= prxparse('/^\d+$/');
if ^prxmatch(re,Value_UPC) and length(Value_UPC) ^= 14 then do;
    isValidity_UPC= 0;
end;
```



Validity_UPC (1.0)

Rule Set Properties Variables Scoring Versions History

ASSIGN isValidity_UPC True

Value_UPC= STRIP(Value_UPC);re= prxparse('/^\d+\$/');if ^prxmatch(re,Value_UPC) and length(Value_UPC) ^= 14 then do; isValidity_UPC= 0;end;

+ Add Rule

2. Create a Decision Field_UPC
 - Select Decisions: select Field_Phone, check the checkbox, then top left menu: duplicate. Rename the Decision to Field_UPC

Duplicate Decision ✕

Name: *

Description:

Version:

Location: *
 📁

Duplicate included objects ☒ [View included objects](#)

Name suffix: ⓘ

Duplicate
Cancel

Select the variables tab, rename all variables and initial values from Phone to UPC

Remark: sometimes a variable rn also contains an initial value that needs to be renamed:

Edit Variable

Name: *

Description:

Data type: *

Variable type:
☐ Input
☒ Output

Initial value:
 ✎

Length:

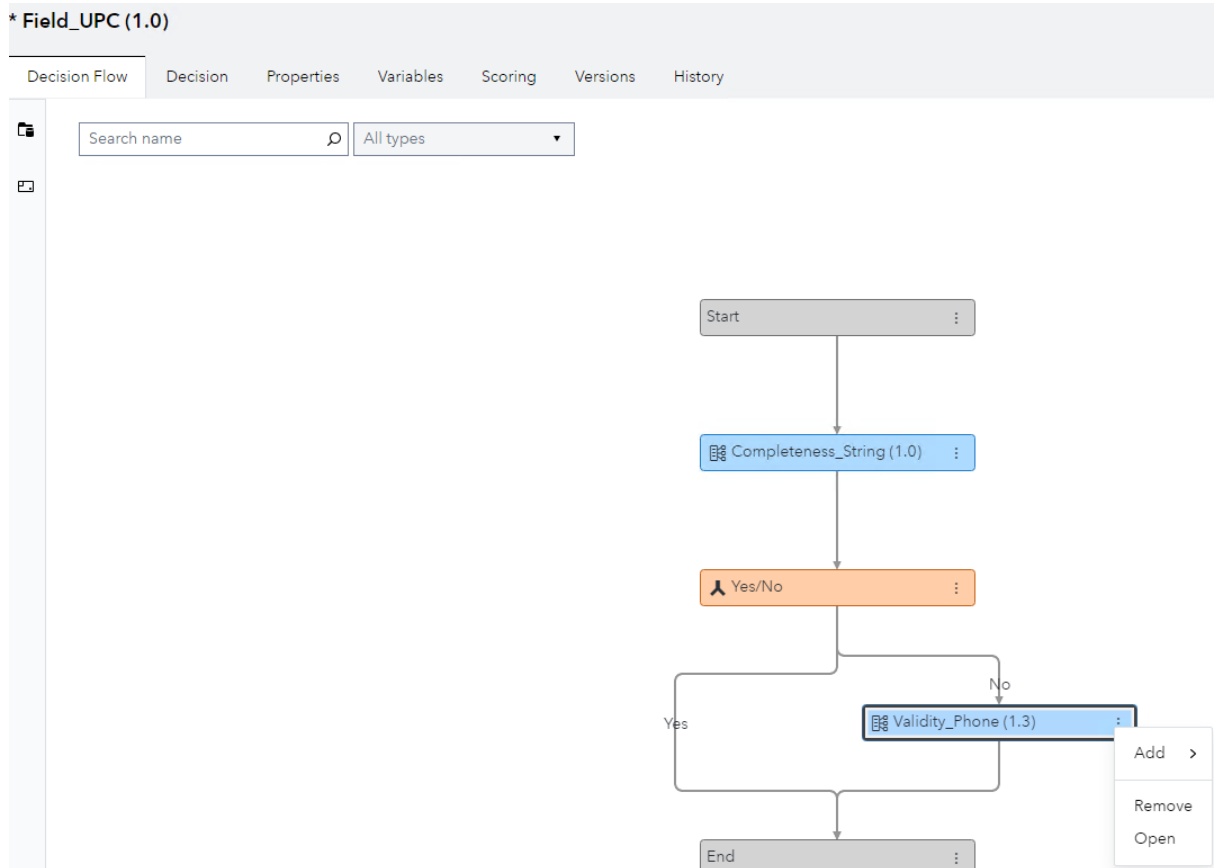
Maximum row count: ⓘ

OK
Cancel

After the renaming, the variables tab should like like following screen:

* Field_UPC (1.0)					
Decision Flow Decision Properties Variables Scoring Versions History					
Decision Variables Global Variables					
Filter		Object type: All objects		Object: Select an item	
				Import Export	
<input type="checkbox"/>	Name	Data Type	<input type="checkbox"/> Input	<input checked="" type="checkbox"/> Output	Initial Value
<input type="checkbox"/>	ID	Character	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	isCompleteness_UPC	Boolean	<input type="checkbox"/>	<input checked="" type="checkbox"/>	True
<input type="checkbox"/>	isValidity_UPC	Boolean	<input type="checkbox"/>	<input checked="" type="checkbox"/>	True
<input type="checkbox"/>	rmCompleteness_UPC	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'Completeness_UPC'
<input type="checkbox"/>	rmValidity_UPC	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'Validity_UPC'
<input type="checkbox"/>	thCompleteness_UPC	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'20:40'
<input type="checkbox"/>	thValidity_UPC	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'20:40'
<input type="checkbox"/>	Value_UPC	Character	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

On the Decision Flow, remove the Validity_Phone Ruleset and Add to the No Branch after the Completeness check, the Validity_UPC Ruleset



Then add to the No branch the Validity_UPC Ruleset:

Choose an Item

🔍

SAS Content > Public > Data Management > DQ Dashboard > Monitor Data Quality > Rules
🔍 📄

- Recent
- My Favorites
- My Folder
- SAS Content
 - Decision Repository
 - ESP Projects
 - Model Repositories
 - Products
 - Public
 - _GitHubCustomSteps
 - Data Management
 - DQ Dashboard
 - Custom Steps
 - Dashboard
 - Dashboard Plans

Name	Date Modified	Type
Validity_Forename	12/09/24 5:48 AM	Rule set
Validity_IBAN	12/09/24 9:09 AM	Rule set
Validity_Phone	12/09/24 5:48 AM	Rule set
Validity_Postcode	12/09/24 5:48 AM	Rule set
Validity_Salary	12/09/24 9:09 AM	Rule set
Validity_Surname	12/09/24 5:48 AM	Rule set
Validity_UPC	12/10/24 2:21 PM	Rule set

Name:

Validity_UPC

Type:

Rule sets

Validity_UPC

Details

Comme

▼ Thumbnails

> Properties

> More

Check the mappings of the Input and Output variables. The Value_UPC in the decision should be mapped to the Value_UPC in the Ruleset. Same for the isValidty_UPC.

5. Create Ruleset to check Accuracy of UOM (Unit of Measure)

3. Create Lookup Table UOMLookup:
CT=Carat, LB=Pound, OZ=Ounce, PK=Peck
Store the lookup table and Activate it
4. create Accuracy_UOM Ruleset:
Duplicate the Ruleset Accuracy_County and save as Accuracy_UOM
 - Rename Variables isAccuracy_UOM (Boolean), Value_UOM (string)
 - Rule: UOM Check
 - Rule_Description: Checking the values for UOM
 - Assign isAccuracy_UOM False
 - Add Rule: if Value_UOM Lookup UOMLookup THEN then isAccuracy_UOM=true

Accuracy_UOM (1.0)

Rule Set Properties Variables Scoring Versions History

ASSIGN isAccuracy_UOM False

▼ Check if valid county

IF Value_UOM LOOKUP UOMLookup

THEN ASSIGN isAccuracy_UOM True

+ Add Rule

5. Create a Decision Field_UOM

Select Decisions: select Field_County, check the checkbox, then top left menu: duplicate

Rename the Decision to Field_UOM.

Select the variable tab, rename all variables and initial values from County to UOM

* Field_UOM (1.0)

Decision Flow Decision Properties Variables Scoring Versions History

Decision Variables Global Variables

Filter

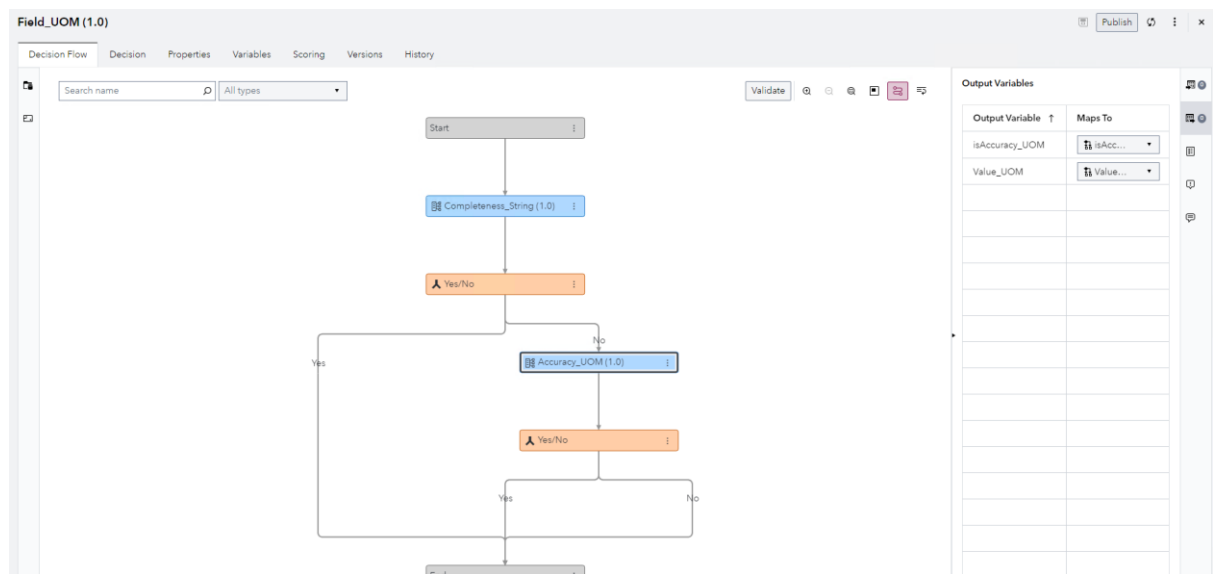
Object type: All objects Object: Select an item

Import Export

<input type="checkbox"/>	Name	Data Type	<input type="checkbox"/> Input	<input checked="" type="checkbox"/> Output	Initial Value
<input type="checkbox"/>	ID	Character	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	isAccuracy_UOM	Boolean	<input type="checkbox"/>	<input checked="" type="checkbox"/>	True
<input type="checkbox"/>	isCompleteness_UOM	Boolean	<input type="checkbox"/>	<input checked="" type="checkbox"/>	True
<input type="checkbox"/>	mAccuracy_UOM	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'Accuracy_UOM'
<input type="checkbox"/>	mCompleteness_UOM	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'Completeness_UOM'
<input type="checkbox"/>	thAccuracy_UOM	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'20:40'
<input type="checkbox"/>	thCompleteness_UOM	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'20:40'
<input type="checkbox"/>	Value_UOM	Character	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

On the Decision Flow, remove the Accuracy_County Ruleset and Add to the No Branch after the Completeness check, the Accuracy_UOM Ruleset

Check the mappings of the Input and Output variables. The Value_UOM in the decision should be mapped to the Value_UOM in the Ruleset. Same for the isAccuracy_UOM.



6. Create a Monitoring Task: Mon_Breakfast_items

Start with a Duplicate of Mon_Person Decision.

Delete all the Field_ decisions from the flow, then delete the variables.

Add Variables from the Field_UOM and Field_UPC decisions from /Public/DataManagement/DQDashboard/Monitor Data Quality

Mon_Breakfast_items (1.0)

Decision Flow Decision Properties Variables Scoring Versions History

Decision Variables Global Variables

Filter

Object type: All objects Object: Select an item

Import Export Add variable

Name	Data Type	Input	Output	Initial Value
ID	Character	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
isAccuracy_UOM	Boolean	<input type="checkbox"/>	<input checked="" type="checkbox"/>	True
isCompleteness_UOM	Boolean	<input type="checkbox"/>	<input checked="" type="checkbox"/>	True
isCompleteness_UPC	Boolean	<input type="checkbox"/>	<input checked="" type="checkbox"/>	True
isValidity_UPC	Boolean	<input type="checkbox"/>	<input checked="" type="checkbox"/>	True
mAccuracy_UOM	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'Accuracy_UOM'
mCompleteness_UOM	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'Completeness_UOM'
mCompleteness_UPC	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'Completeness_UPC'
mValidity_UPC	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'Validity_UPC'
thAccuracy_UOM	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'20:40'
thCompleteness_UOM	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'20:40'
thCompleteness_UPC	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'20:40'
thValidity_UPC	Character	<input type="checkbox"/>	<input checked="" type="checkbox"/>	'20:40'
Value_UOM	Character	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Value_UPC	Character	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Decision
Rule set
Code file
Segmentation tree
Data table
Custom variable

On the Decision Flow, add the Field_UOM and Field_UPC Decisions from the folder

Choose an Item

Search

SAS Content > Public > Data Management > DQ Dashboard > Monitor Data Quality > Fields

Name	Date Modified	Type
Field_Phone	12/09/24 5:48 AM	Decision
Field_Postcode	12/09/24 5:48 AM	Decision
Field_Surname	12/09/24 5:48 AM	Decision
Field_Title	12/09/24 5:48 AM	Decision
Field_Town	12/09/24 5:48 AM	Decision
Field_UOM	12/10/24 2:01 PM	Decision
Field_UPC	12/10/24 2:35 PM	Decision

Name: Field_UOM Type: Decision

Field_UOM

Details Comments

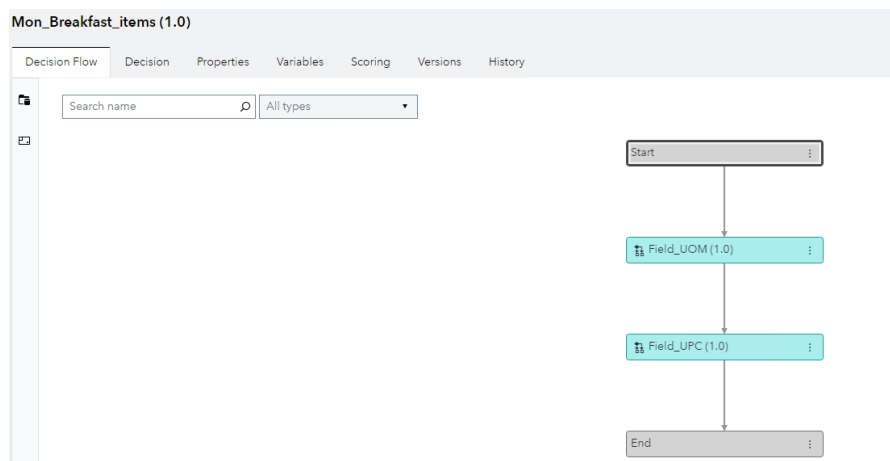
Thumbnails

Properties

More

OK Cancel

The resulting Monitor Task Decision flow should look like:



Select Scoring, add a new Test:

Select Breakfast_items and map the columns UPC and UOM to Value_UPC and Value_UOM.

Run the test, check if the test is successful.

Mon_Breakfast_Items (1.0)

Decision Flow Decision Properties Variables **Scoring** Versions History

Tests Scenarios Publishing Validation

<input checked="" type="checkbox"/>	Name	Results	Status	Date Modified	Decision Version
<input checked="" type="checkbox"/>	Mon_Breakfast_Items_Test2			Dec 10, 2024 4:17:05 PM	1.0

New test Run

Properties

Name: Mon_Breakfast_Items_Test2

Status: Completed successfully

7. Appendix

Prepare Demo on Training Virtuallab

To ensure that DQ Dashboard demo works as desired follow the steps below to load the data into memory and to run the Studio Flows to load the DQ Dashboard tables.


1. Fork the following repository: <https://github.com/paulvanmol/dqdashboard.git> to your personal git account
2. Clone the Repository from SAS Studio
Create a new empty folder dqdashboard:
Create a new git profile:

Clone a Repository

Repository:

<https://github.com/example/repo.git>
<git@bitbucket.org/example/repo.git>

Server location:



Profile:

3. Download the required files from the dqdashboard folder:
 - a. Packages/DQDashboard.json package from SAS Studio to your Downloads folder.
 - b. Data/ADDRESS.xlsx, PERSON.xlsx, DQ_SCORE_HIST_RAW.xlsx
4. Import DQDashboard.json package in SAS Environment Manager
 - a. Login as sasadm + Inxsas to have administrative privileges:
 - b. Select Manage Environment:
 - c. Select Import tab
 - d. Import the DQDashboardPublic.json
 - e. Map the Target Server to cas-shared-default, Map the Target Library PUBLIC, WORK CAS Library to the Public CAS Library:

Import Mapping Sets

Import

Source Mapping Preview Results

Mapping

Content Users and Groups Tables Data Resources Other Mapping set: System default ? Save As

Source	Target Server	Target Library	Target Table	Exists
cas-shared-default	cas-shared-default			✓
Public		Public		✓
DQ_DASHBOARD_DATA	cas-shared-default	Public	DQ_DASHBOARD_DA	✗
DQ_DASHBOARD_OV	cas-shared-default	Public	DQ_DASHBOARD_OV	✗
DQ_SCORE_HIST	cas-shared-default	Public	DQ_SCORE_HIST	✗
60842400-32a5-4e4e-a1c9-391ec02...	Compute			ⓘ
PUBLIC		PUBLIC		ⓘ

f. Other tables should map to Compute, Public and WORK libraries:

Source Mapping Preview Results

Mapping

Content Users and Groups Tables Data Resources Other Mapping set: System default ? Save As

Source	Target Server	Target Library	Target Table	Exists
60842400-32a5-4e4e-a1c9-391ec02...	Compute			ⓘ
PUBLIC		PUBLIC		ⓘ
PERSON	Compute	PUBLIC	PERSON	ⓘ
ADDRESS	Compute	PUBLIC	ADDRESS	ⓘ
DQ_DASHBOARD_OV	Compute	PUBLIC	DQ_DASHBOARD_OV	ⓘ
DQ_SCORE_HIST_RAW	Compute	PUBLIC	DQ_SCORE_HIST_RAW	ⓘ
WORK		WORK		ⓘ

g. Check if Import is successful:

Source Mapping Preview Results

Results

Import job has completed

Total time to import DQDashboardPublic.json: 0:00:19 View log

Show column abbreviations

Object Type	Count	Compl...	Warnin...	Errors	Failed	Skipped	Stopped
Folder	(13)	13	0	0	0	0	0
Flow	(3)	3	0	0	0	0	0

Count: 10

Upload the data to SAS Studio Explorer:

Upload the data of the dqdashboard repository to the home directory of the student:

- Either by doing a Git Clone in SAS Studio to the /gelcontent/dqdashboard folder
- Or by using the upload Button in the Explorer:

ns

View

Open

↑

↓

↑

↓

er Shortcuts

ortcut to My Folder

ome

1 Courses

>

dcon

>

dmvw

>

dqdashboard

>

data

ADDRESS.xlsx

data.zip

DQ_DASHBOARD...

DQ_DASHBOARD...

DQ_SCORE_HIST...

DQ_SCORE_HIST...

Upload Files

Size limit for each selected file is 100 MB.

Upload to: /home/student/Courses/dqdashboard

Attachments (6)

+

PERSON.xlsx

3.1 MB

DQ_SCORE_HIST_RAW.xlsx

6.2 KB

DQ_SCORE_HIST.xlsx

6.2 KB

DQ_DASHBOARD_OV.xlsx

6 KB

DQ_DASHBOARD_DATA.xlsx

1.3 MB

ADDRESS.xlsx

2.2 MB

Files

Home

Courses

dcon

dmvw

dqdashboard

data

ADDRESS.xlsx

data.zip

DQ_DASHBOARD_D...

DQ_DASHBOARD_...

DQ_SCORE_HIST_R...

DQ_SCORE_HIST.xlsx

PERSON.xlsx

18

Go to SAS Studio (Develop Code and Flows)

1.1. Go to location: SAS Content/GTPPub/Data Management/DQ Dashboard/SAS Programs

1.2. Open and run SAS job: Load tables into memory.sas

```
cas casauto;
%let outcaslib=PUBLIC;
%let incaslib=GTPPUB;
%let path=/gelcontent/dqdashboard;

proc cas;
    session casauto;
table.dropcaslib /caslib="&incaslib" quiet=true;
table.addCaslib /
    caslib="&incaslib"
    description="Monitor data"
    dataSource={srctype="path"}
    path="&path/data";

    table.dropTable /
        caslib= "&outcaslib",
        name= "ADDRESS",
        quiet= True;
run;
table.loadTable /
    caslib= "&incaslib",
    path="ADDRESS.xlsx",
    casout={
        caslib="&outcaslib",
        promote= True
    };
run;
table.dropTable /
    caslib= "&outcaslib",
    name= "PERSON",
    quiet= True;
run;
table.loadTable /
    caslib= "&incaslib",
    path="PERSON.xlsx",
    casout={
        caslib="&outcaslib",
        promote= True
    };
run;
table.dropTable /
    caslib= "&outcaslib"
    name= "DQ_SCORE_HIST_RAW",
    quiet= True;
run;
table.loadTable /
```

```
caslib= "&incaslib",  
path="DQ_SCORE_HIST_RAW.xlsx",  
casout={  
    caslib="&outcaslib",  
    promote= True  
};  
run;  
quit;
```

1.3. Go to location: SAS Content/Public/Data Management/DQ Dashboard/Dashboard Flows

1.4. Open and run flows:

- Monitor_Person.flw
- Monitor_Address.flw
- Write_Dashboard_Tables.flw

2. Go to SAS Drive (Share and Collaborate)

2.1. Open Dashboard: DQ Dashboard in location: SAS Content/Public/Data Management/DQ Dashboard/Dashboard to ensure the Dashboard data got generated correctly.