A Self-Structuring Approach to Acquiring and Traversing JSON Extracts from the CDISC Library

Presented by Carlo Radovsky, Immanant

Background

CDISC Library	Provides content through UI (manual download)
	Also accessible via API in a variety of formats
	Not all formats are the same - Manually downloaded Excel has less content than JSON
JSON	JavaScript Object Notation
	Compact, easy to read, comparatively quick transmit speed
	Simple Hierarchy, content nested in levels that are self-defining
	Full Library Content

Background

Non-SAS Solutions	Python
	Database-Native JSON (PostgeSQL, MongoDB)
	etc.
Benefits	Native Processing
	Acquire and work with JSON objects directly

Background

SAS	+	Integrated with existing code bases and workflows
	~	Requires tabular representation
	−	JSON Extract into SAS is non-intuitive, requiring preprocessing to make usable and accessible

Background

SAS ~ JSON Extract into SAS is non-intuitive, requiring preprocessing to make usable and accessible

✓ It can be optimal to extract only what is needed at a given point in time via the API
- E.g., a single domain

✓ A full extract could be for
- Impact Analysis
- Cross-standard comparisons

Why Non-Intuitive?

SDTMIG 3.4

50 distinct datasets

44 Linking datasets (several empty)	5 Discreet Content Datasets	1 Complete Dataset

Why Non-Intuitive?

Linking	Many datasets are external and supportive e.g., a pointer to the Class in the associated version of SDTM model
	The links (e.g., Class to Dataset) are context free, surrogate keys without intrinsic meaning or hierarchy
Discrete Content Datasets	Requires documentation or pre-existing knowledge of hierarchy to process
Complete Dataset	Nested Hierarchy
	No Path (what contains what) present in the data

What’s More Useful?

Intuitive Representation	Natural Keys with contextual meaning (Domain Class, Dataset Name, Variable Name)
Self-Determined Hierarchy	No foreknowledge of the order of the discrete levels, or even how many levels are present
Flexible Representation	One output per level vs. single output

What's More Useful?

Intuitive Representation	Natural Keys with contextual meaning (Domain Class, Dataset Name, Variable Name)
Self-Determined Hierarchy	No foreknowledge of the order of the discrete levels, or even how many levels are present
Flexible Representation	One output per level vs. single output

What to Use?

Complete Dataset	Has All of the Content
	Has all of the relationships
	Simple structure, with consistent naming conventions across varying hierarchies

Complete Dataset Structure

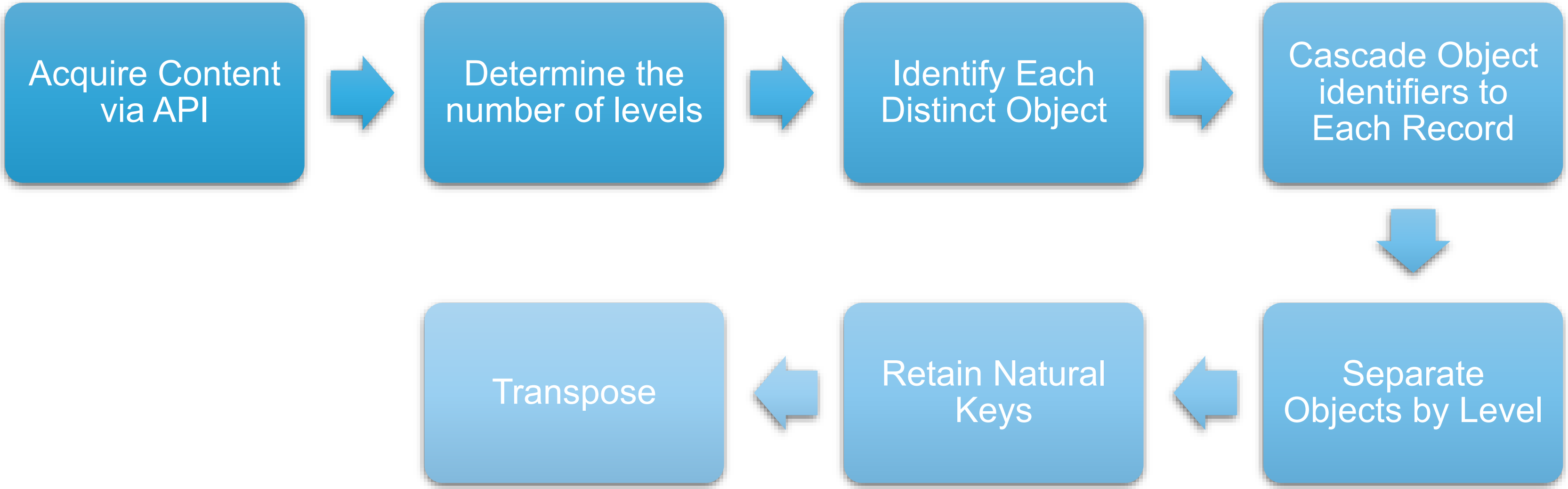
P	P1	P…	P{n}	V	Value
The Hierarchy Level of the Record	Either • Level 1 Hierarchy Name of the record • Attribute Name at Level 1		Either • Level {n} Hierarchy Name of the record • Attribute Name at Level {n}	Value Flag • 0 = Start of a new Item. Value column is Null • 1 = An attribute of the current item	The value of the attribute identified in P{n} where {n} is the number in the P column

Efficient but Insufficient

P	P1	P2	P3	P4	V	Value
4	classes	datasets	datasetVariables	name	1	STUDYID

- 63 records are exact duplicates
- The records before and after determine
  - Dataset
  - Variable order
  - Child attributes
  - etc.

Solution Overview


Acquire Content via API
Determine the number of levels
Identify Each Distinct Object
Cascade Object identifiers to Each Record
Separate Objects by Level
Retain Natural Keys
Transpose

1: Acquire Content via API

Simple call to extract a targeted product

- ➢ A product is a complete version of a standard, terminology, etc.

```
proc http
   url='https://api.library.cdisc.org/api/mdr/sdtmig/3-4'
   out=response;
               headersnote
                            "api-key" = "xxxx"
                            "Accept" = "application/json";
run;

libname json_lib json fileref=response;
```

1: Acquire Content via API

More complex approach

- ➢ Query Library at a higher level

  url="https://api.library.cdisc.org/api/mdr/products"

- ➢ Extract the full product list

- ➢ Subset to the products of interest

- ➢ This returns all versions of all models and IGs (CDASH, SEND, SDTM, ADaM, etc.), and all terminology published in 2025

Query the _LINKS object and retain products with
- ❖ Type = Foundation Model
- ❖ Type = Implementation Guide
- ❖ Type = Terminology and the HREF containing 2025

1: Acquire Content via API

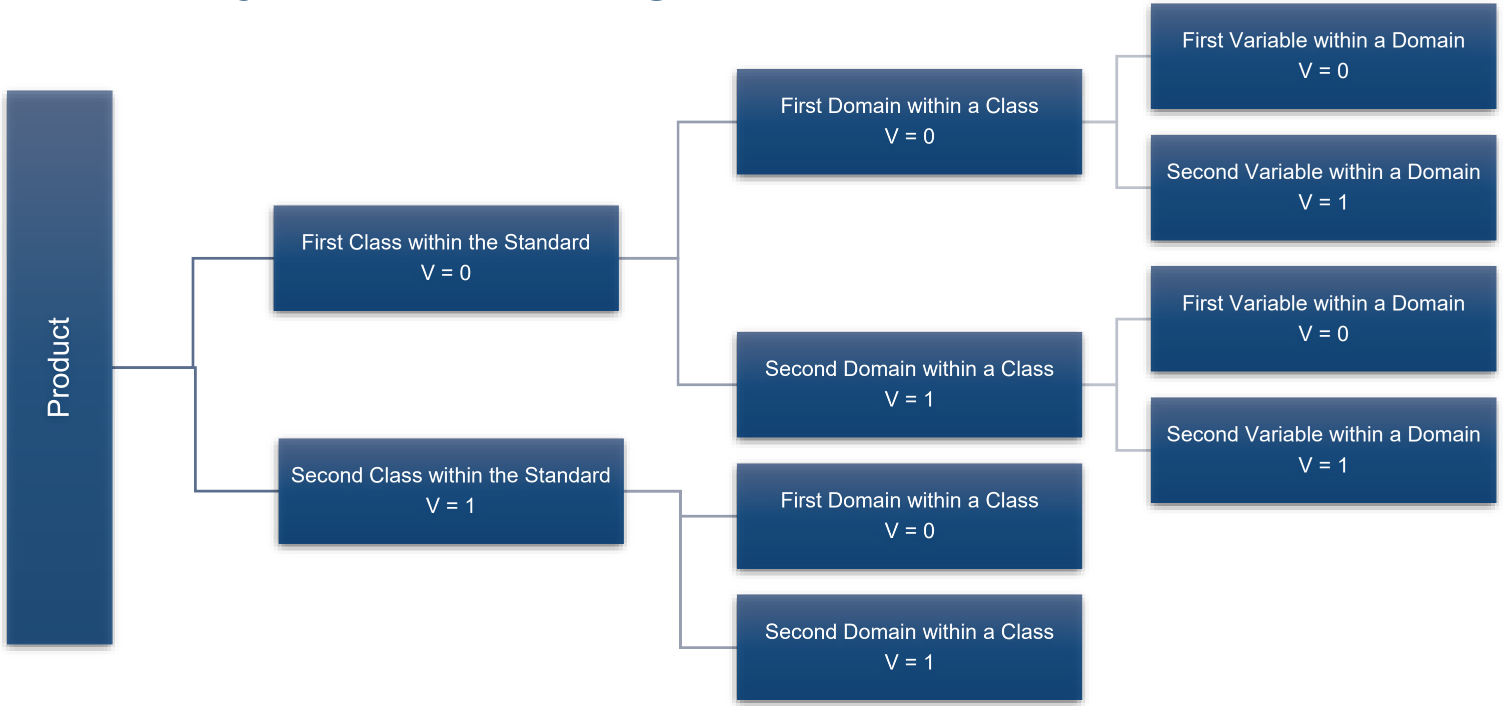
- Each product (or version of a product) may have a distinct hierarchy
- The mapping utility doesn't need to know any of the details beforehand
- The algorithm will process any and all structures

2: Determine the Number of Levels

- Identify the number of variables in the full extract that being with P and have a numeric suffix
- The actual number of object levels will be N – 1
- The last “level” will always be the direct attributes of the lowest-level objects (Name, Description, Ordinal, Role, etc.)

3: Identify Each Distinct Object

Each object level begins with V = 0


Product
First Class within the Standard
V = 0
Second Class within the Standard
V = 1
First Domain within a Class
V = 0
Second Domain within a Class
V = 1
First Domain within a Class
V = 0
Second Domain within a Class
V = 1
First Variable within a Domain
V = 0
Second Variable within a Domain
V = 1
First Variable within a Domain
V = 0
Second Variable within a Domain
V = 1

3: Identify Each Distinct Object

Each object begins with a changed value of value of P

- Increase of +1 or more is a new object below the prior object
- Decrease of -1 or more is a new object at the same level or above
- All objects have a Name attribute (typically not the first record for the object)

3: Identify Each Distinct Object

Each object begins with a changed value of value of P

- Increase of +1 or more is a new object below the prior object
- Decrease of -1 or more is a new object at the same level or above
- All objects have a Name attribute
  (typically, not the first record for the object)

3: Identify Each Distinct Object

P	P1	P2	P3	P4	P5	P6	Value
1	name						SDTMIG v3.4
2	classes	name					Interventions
3	classes	datasets	name				AG
4	classes	datasets	datasetVariables	name			STUDYID
5	classes	datasets	_links	priorVersion	href		/mdr/sdtmig/3-3 /classes/ GeneralObservations
6	classes	datasets	datasetVariables	_links	Parent Dataset	title	Procedure Agents

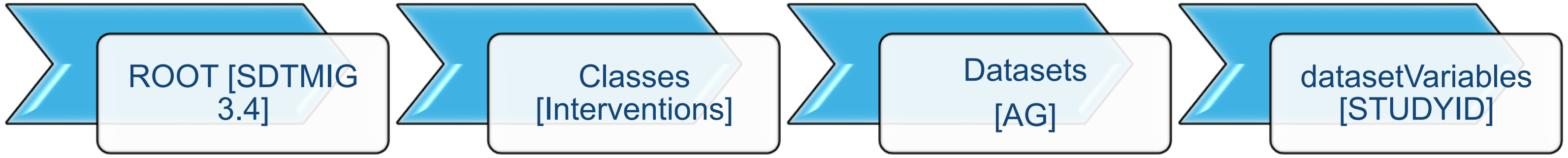
3: Identify Each Distinct Object

P	P1	P2	P3	P4	P5	P6	Value
1	name						[illegible]
2	classes	name					
3	classes	datasets	name				
4	classes	datasets	datasetVariables	name			STUDYID
5	classes	datasets	_links	priorVersion	href		/mdr/sdtmig/3-3 /classes/ GeneralObservations
6	classes	datasets	datasetVariables	_links	Parent Dataset	title	Procedure Agents

Attribute Names are captured in P1 – P{n}
where {n} is the value in P

4: Cascade Object Identifiers to Each Record

- Each object will have a unique path



- Assign the path to each record

- Can be
  - A dynamically created set of key variables
  - Single Composite Key variable with delimited Text

- Requires multiple passes through the content
  - First: Identify the Objects, capturing the Name of each
  - Second: Attach the path to each record of a given object

5: Retain Natural Keys

- As noted early on, attribute Names are captured in P1 – P{n}, where {n} is the value in P

- Attribute names are re-used across object types
  - Classes **Name**
  - Datasets **Name**
  - datasetVariables **Name**

5: Retain Natural Keys

- For each object type, rename Name to the type

- For other attributes occurring in more than a single object type, rename with type (and other path values as necessary) as the prefix
  - Datasets_Label
  - datasetVariables_Label
  - datasetVariables_links_ParentDataset_title

6: Transpose

Convert vertical structure to horizontal based on the path and attribute names

P	Path	Attribute Name	Value
4	ROOT [SDTMIG 3.4] | Classes [Interventions] | Data1sets [AG] | datasetVariables [STUDYID]	Datasetvariables	STUDYID
4	ROOT [SDTMIG 3.4] | Classes [Interventions] | Data1sets [AG] | datasetVariables [STUDYID]	Datasetvariables_Label	Study Identifier
4	ROOT [SDTMIG 3.4] | Classes [Interventions] | Data1sets [AG] | datasetVariables [USUBJID]	Datasetvariables	USUBJID
4	ROOT [SDTMIG 3.4] | Classes [Interventions] | Data1sets [AG] | datasetVariables [USUBJID]	Datasetvariables_Label	Unique Subject Identifier

6: Transpose

Convert vertical structure to horizontal based on the path and attribute names

ROOT	CLASSES	DATASETS	DatasetVariables	DatasetVariables_label
SDTMIG 3.4	Interventions	AG	STUDYID	Study Identifier
SDTMIG 3.4	Interventions	AG	USUBJID	Unique Subject Identifier

Summary

JSON Extracts include far more than Excel

- Cross-links to terminology and other versions
- More atomic, which each attribute distinct

JSON Hierarchy Self-Labeling

- Can be leveraged to auto-structure

Flexible

- Could dynamically carry attributes through the hierarchy
- Dynamically generate the hierarchy in whole or part

Thank You!

Carlo.Radovsky@Immanant.com