Creating Dataset-JSON Using proc JSON and Extended Attribute in SAS

Presentation by Yutaka Morioka and Yuki Nakagawa; team also includes Satoru Orii, Li Zhang, and Hiroshige Takata

Meet the Speakers

Yutaka Morioka

Title: SAS Programmer

Organization: EPS Corporation

A SAS programmer based in Japan. Besides my work as a clinical data scientist, actively engaging in disseminating various SAS programming techniques from introductory level to advanced. Working as an organizer of the SAS User's Group in Japan.

Yuki Nakagawa

Title: SAS Programmer / Biostatistician

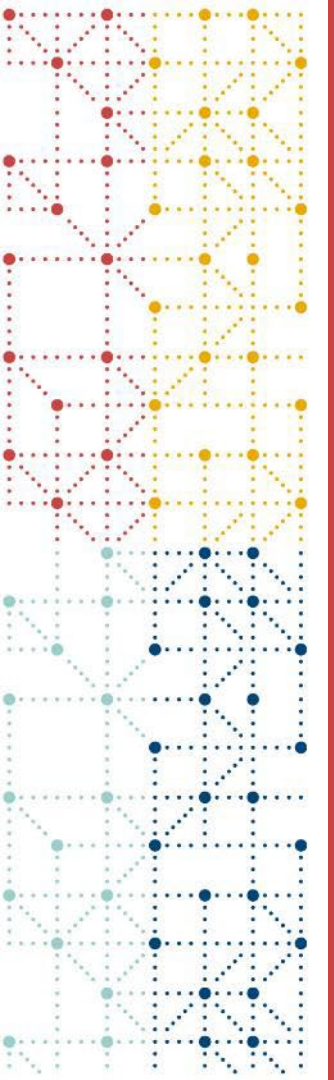
Organization: EPS Corporation

Since joining the Statistical Analysis Department of EPS Corporation in 2019, Yuki Nakagawa has been engaged in clinical trials service from SDTM to statistical analyses as the SAS Programmer and biostatistician.

Disclaimer and Disclosures

- *The views and opinions expressed in this presentation are those of the authors and do not necessarily reflect the official policy or position of CDISC.*

- *The authors have no real or apparent conflicts of interest to report*



Agenda

1. Limitations & Problems of XPT
2. Nice to meet Dataset-JSON.
3. JSON？？
4. Dataset-JSON Specification
5. Creating Dataset-JSON Using proc JSON
6. Import Dataset-JSON to SAS
7. SAS Extended attributes
8. Harmonization with define.xml
9. Conclusion

Limitations & Problems of XPT

Limitations & Problems of XPT

The SAS® Version 5 (V5) transport file format is an open standard developed by SAS to support data transfers between systems, especially those running different operating systems." SAS V5, being an open standard, allowed FDA to specify it as the standard required for data submission

(Source:A Short History of CDISC and SAS Transport Files)

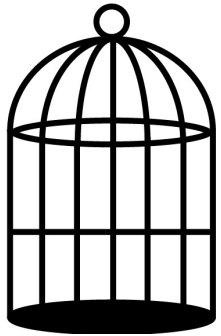
But nearly 25 years have passed since then!

Limitations & Problems of XPT

Limitations

- Variable names are limited to a maximum length of **8** byte.
- Variable labels are limited to a maximum length of **40** byte.
- Character variable data are limited to a maximum length of **200** byte.
- Cannot have Encoding information.
- Only ASCII characters can be used, Numeric type is Integer or double only.

etc.



Limitations & Problems of XPT

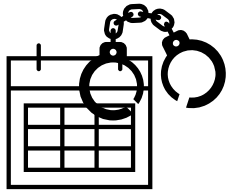
Problems

- Fixed length character makes file size heavier, because blank are filled up to the maximum length.

  -> Unsuitable for ePRO or wearable device data.

- Limit data to a two-dimensional structure.

  -> Impoverishing SEND, SDTM, and ADaM structures.



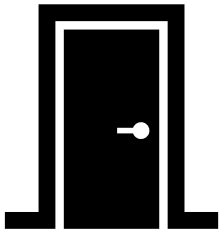
- Incompatible with modern systems.

  -> It is unwieldy from EDC, R, Python, etc.

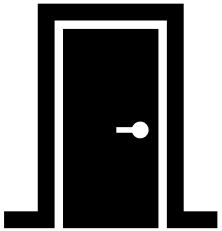
Nice to meet Dataset-JSON.

Next Stage??

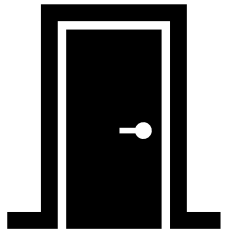
SAS Version 8
Transport format

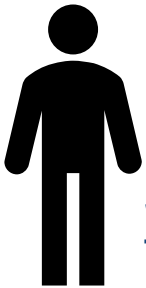


Dataset-XML



Dataset-JSON





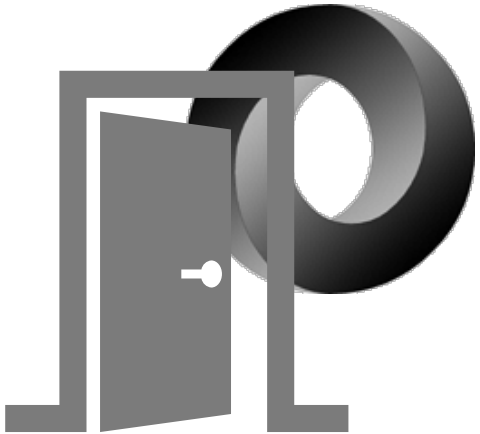
SAS Version 5
Transport format

Nice to meet Dataset-JSON.

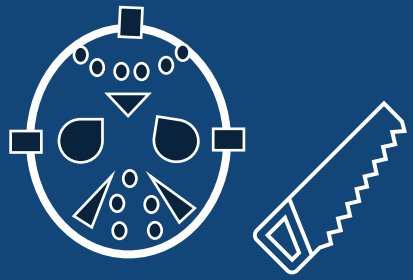
JSON (JavaScript Object Notation)

Dataset-JSON was released as part of ODM version 2.0 in 2023.
Dataset-JSON version 1.1 is currently under development and will be published as an independent standard.

✓ In this presentation, Dataset-JSON version 1.0 is focused.

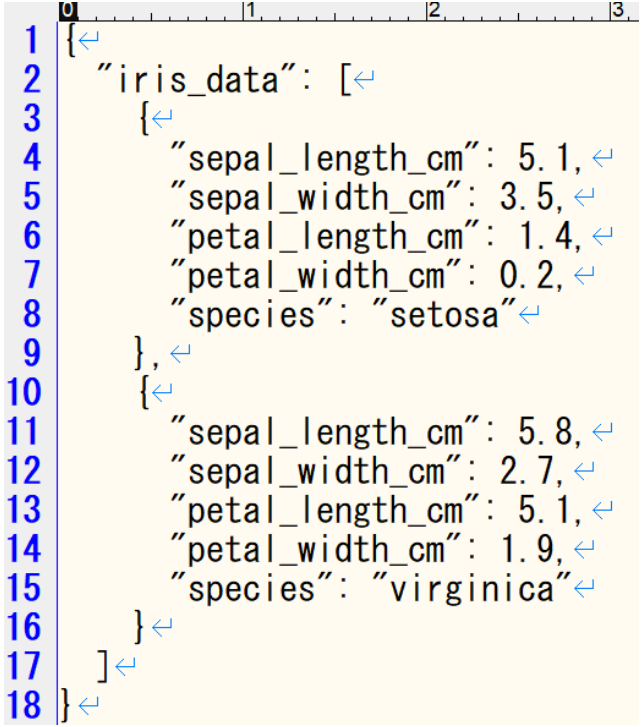


JSON？？



JSON

- Lightweight, text-based, language-independent data interchange format.
- Most modern apps and web services support Dataset-JSON.
- Consisted of Objects (key–value pairs) and Arrays.
- The same expressive power as Dataset-XML.
- Easy to read for both humans and machines.
- Simple structure.



```
{
  "iris_data": [
    {
      "sepal_length_cm": 5.1,
      "sepal_width_cm": 3.5,
      "petal_length_cm": 1.4,
      "petal_width_cm": 0.2,
      "species": "setosa"
    },
    {
      "sepal_length_cm": 5.8,
      "sepal_width_cm": 2.7,
      "petal_length_cm": 5.1,
      "petal_width_cm": 1.9,
      "species": "virginica"
    }
  ]
}
```

JSON?? - Data Types

Data Type	Point	Example
String	Written in double quotes	{**"First Name"** : **"Yutaka"**}
Number	integer or float	{"Age" : **40**}
Object	Enclosed in {} Each name is followed by colon key/value pairs are separated by comma	**{** "presenter" : **{**"First Name" : "Yutaka", "Age" : 40**}** **}**

Array	Enclosed in []
	Values are separated by comma
		{
		"presenters" : **[**"Yutaka", "Yuki"**]**
		}

Boolean	true/false	{"Age>=30" : **true**}, {"Age<30" : **false**}
null	Double quotes are unneeded	{"Middle Name" : **null**}, {"Salary" : **null**}

JSON?? - Object

Key/Value pairs in { } are separated by comma.
The order of Key/Value pairs can be interchange.

```
{
"String1" : Value
,
"String2" : "Value"
,
"String3" : Value
}
```

```
{
"First Name":"Yutaka"
,
"Last Name":"Morioka"
,
"Age":40
}
```

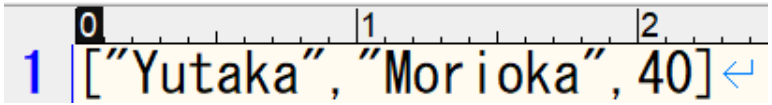
```
{
"First Name" : "Yutaka",
"Last Name" : "Morioka",
"Age" : 40
}
```

JSON?? - Array

Values (Objects) in [ ] separated by comma.
All types do not need to be the same.
The order can be interchange.

[Value , Object , Value]

["Yutaka" , "Morioka" , 40]



JSON?? - Objects in Array


```
[
 {
 "First Name":"Yutaka",
 "Last Name":"Morioka",
 "Age":40
 }
,
 {
 "First Name":"Yuki",
 "Last Name":"Nakagawa",
 "Age":29
 }
]
```

JSON?? - Data Types

Data Type	Point	Example
String	Written in double quotes	{**"First Name"** : **"Yutaka"**}
Number	integer or float	{"Age" : **40**}
Object	Enclosed in {} Each name is followed by colon key/value pairs are separated by comma	**{** "presenter" : **{**"First Name" : "Yutaka", "Age" : 40**}** **}**

Array	Enclosed in []
	Values are separated by comma
		{
		"presenters" : **[**"Yutaka", "Yuki"**]**
		}

Boolean	true/false	{"Age>=30" : **true**}, {"Age<30" : **false**}
null	Double quotes are unneeded	{"Middle Name" : **null**}, {"Salary" : **null**}

Dataset-JSON Specification

File-size simulation

		SAS v5 XPT	Dataset-XML	Dataset-JSON
SDTM.AE	16 obs	8.6 KB	14.7 KB	5.4 KB
SDTM.VS	720000 obs	189 MB	634 MB	96.7 MB

No zip or other compression was performed.

Conversion to Dataset-XML using “SAS® Macros to support Dataset-XML v1.0.0”

AE used the sample in “SAS® Macros to support Dataset-XML v1.0.0” v1.0.0 as is.

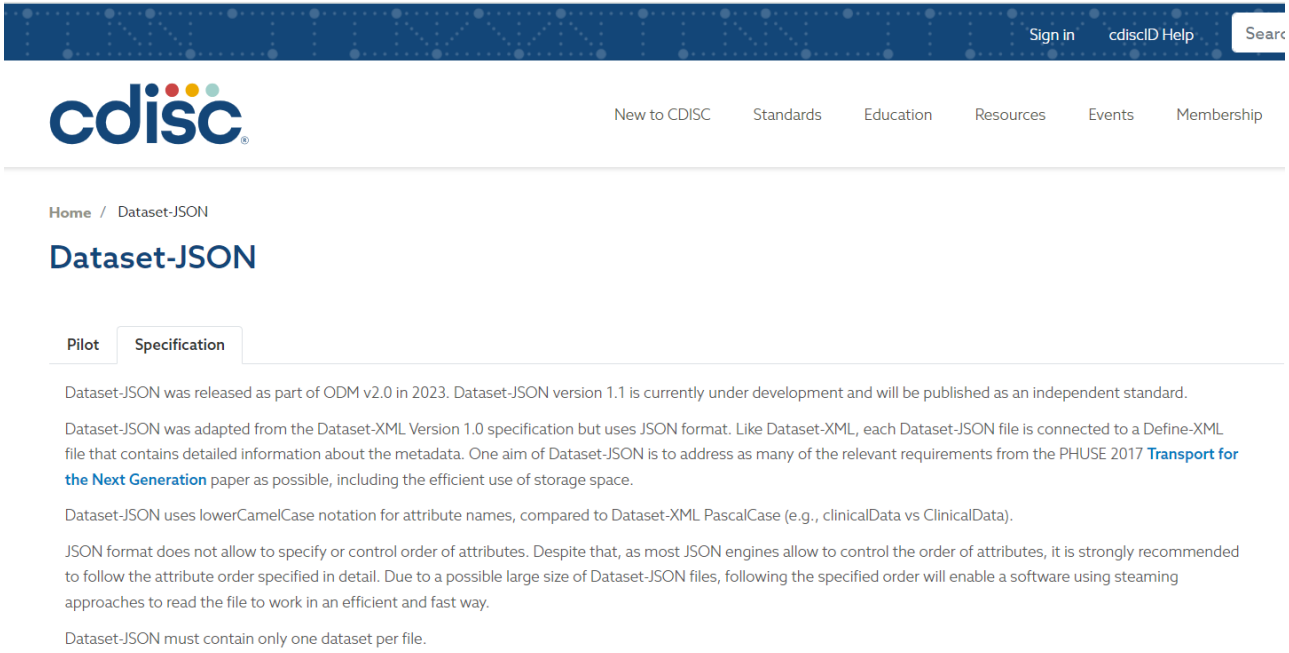
VS used 10,000 times more samples in “SAS® Macros to support Dataset-XML v1.0.0”.

Dataset-JSON created without pretty option

Dataset-JSON Specification

Dataset-JSON Specification

https://www.cdisc.org/dataset-json



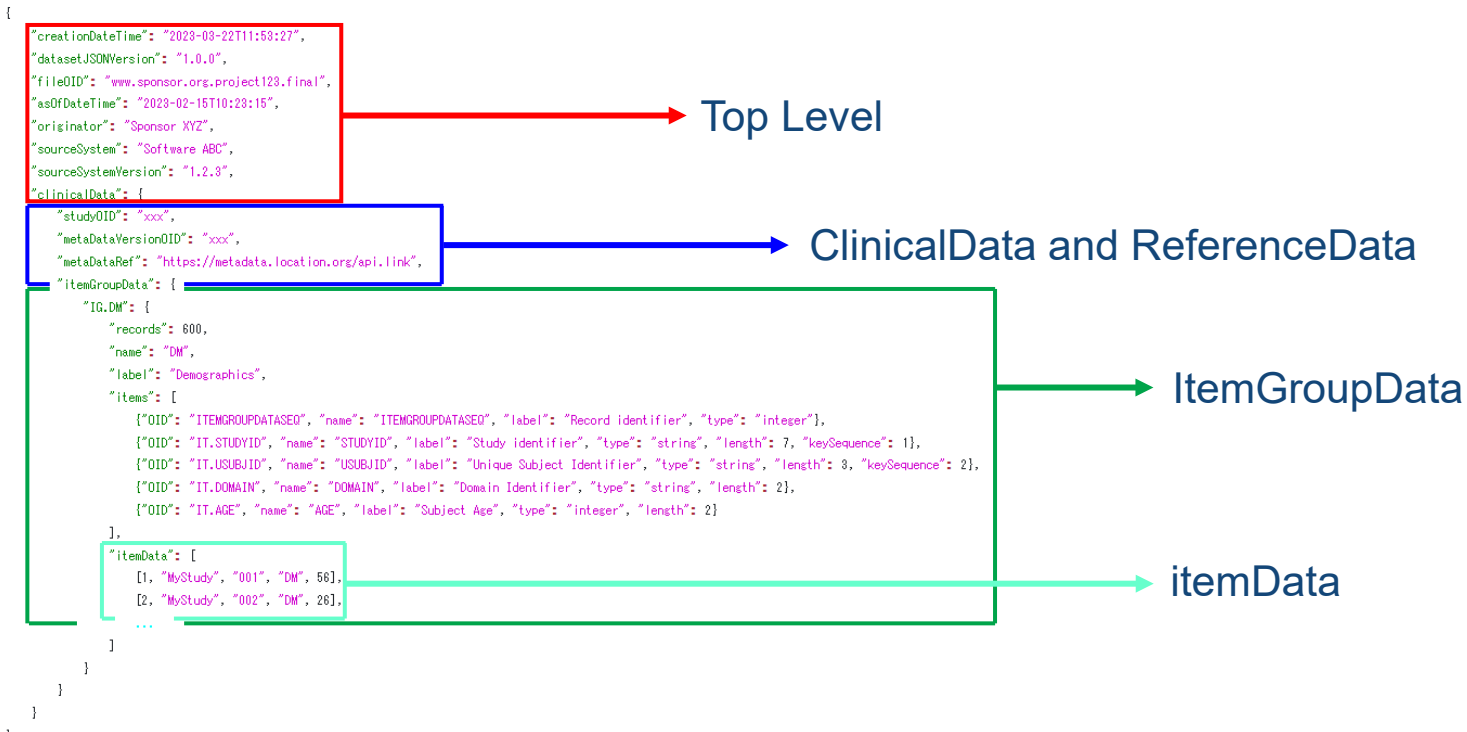
Dataset-JSON Specification

https://www.cdisc.org/dataset-json

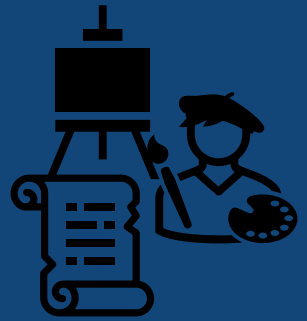

```
{
   "creationDateTime": "2023-03-22T11:53:27",
   "datasetJSONVersion": "1.0.0",
   "fileOID": "www.sponsor.org.project123.final",
   "asOfDateTime": "2023-02-15T10:23:15",
   "originator": "Sponsor XYZ",
   "sourceSystem": "Software ABC",
   "sourceSystemVersion": "1.2.3",
   "clinicalData": {
      "studyOID": "xxx",
      "metaDataVersionOID": "xxx",
      "metaDataRef": "https://metadata.location.org/api.link",
      "itemGroupData": {
         "IG.DM": {
            "records": 600,
            "name": "DM",
            "label": "Demographics",
            "items": [
               {"OID": "ITEMGROUPDATASEQ", "name": "ITEMGROUPDATASEQ", "label": "Record identifier", "type": "integer"},
               {"OID": "IT.STUDYID", "name": "STUDYID", "label": "Study identifier", "type": "string", "length": 7, "keySequence": 1},
               {"OID": "IT.USUBJID", "name": "USUBJID", "label": "Unique Subject Identifier", "type": "string", "length": 3, "keySequence": 2},
               {"OID": "IT.DOMAIN", "name": "DOMAIN", "label": "Domain Identifier", "type": "string", "length": 2},
               {"OID": "IT.AGE", "name": "AGE", "label": "Subject Age", "type": "integer", "length": 2}
            ],
            "itemData": [
               [1, "MyStudy", "001", "DM", 56],
               [2, "MyStudy", "002", "DM", 26],
               ...
            ]
         }
      }
   }
}
```

Dataset-JSON Specification

https://www.cdisc.org/dataset-json



Creating Dataset-JSON Using proc JSON



Creating Dataset-JSON Using proc JSON

Proc JSON

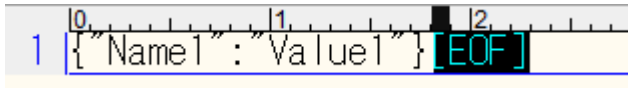
PROC JSON OUT=fileref | "external-file" <options>;
WRITE OPEN type ;
WRITE CLOSE ;
WRITE VALUES value(s)< /options > ;
EXPORT<libref.>SAS-data-set< (SAS-data-set-options)> < /options > ;

Statement	Task
WRITE OPEN	Open and nest a JSON container in the output file
WRITE CLOSE	Close a JSON container that is open in the output file
WRITE VALUES	Write one or more values to the output file
EXPORT	Identify the SAS data set to export and control the resulting output

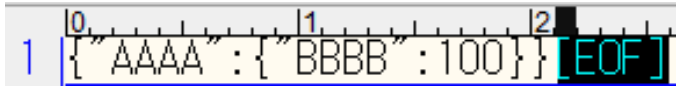
Creating Dataset-JSON Using proc JSON

write values, write open object～write close

```
proc json out="xxxx¥Output1.json";
   write values "Name1" "Value1";
run;
```




```
proc json
out="xxxx¥Output2.json";
 write open object ;
 write values "AAAA";
  write open object ;
   write values "BBBB" 100;
  write close;
write close;
run;
```

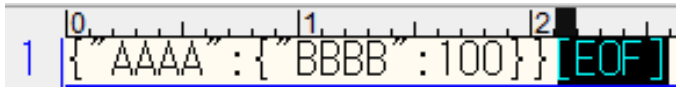


Creating Dataset-JSON Using proc JSON

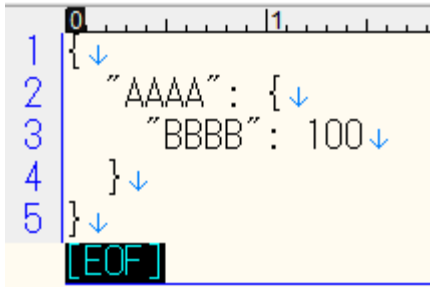
pretty option

```
proc json
out="xxxx¥Output2.json" pretty;
  write open object ;
  write values "AAAA";
   write open object ;
    write values "BBBB" 100;
   write close;
  write close;
run;
```

No pretty option



pretty option



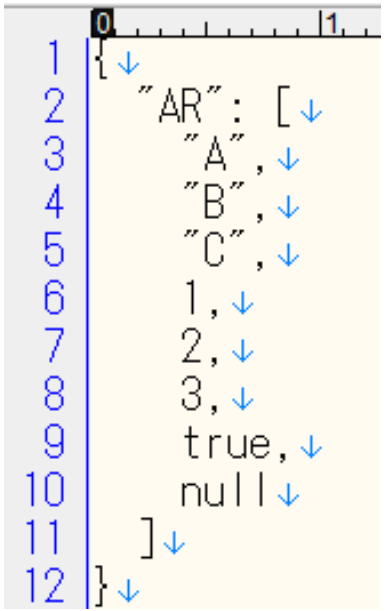
Human Readable

"pretty" creates a more human-readable format using indentation to illustrate the JSON container structure.

Creating Dataset-JSON Using proc JSON

write open array～write close

```
proc json out="xxxx¥Output3.json" pretty;
  write open object;
    write values "AR";
    write open array;
      write values "A" "B" "C" 1 2 3 true null;
    write close;
  write close;
run;
```



Creating Dataset-JSON Using proc JSON

test dataset

```
data dm;
attrib
STUDYID label="Study Identifier"           length=$7.
USUBJID label="Unique Subject Identifier" length=$3.
DOMAIN  label="Domain Abbreviation"        length=$2.
AGE     label="Age"                        length= 8.
;
input STUDYID USUBJID DOMAIN AGE;
cards;
MyStudy, 001, DM, 56
MyStudy, 002, DM, 26
;
run;
```

Variables in Creation Order

#	Variable	Type	Len	Label
1	STUDYID	Char	7	Study Identifier
2	USUBJID	Char	3	Unique Subject Identifier
3	DOMAIN	Char	2	Domain Abbreviation
4	AGE	Num	8	Age

VIEWTABLE: Work.Dm

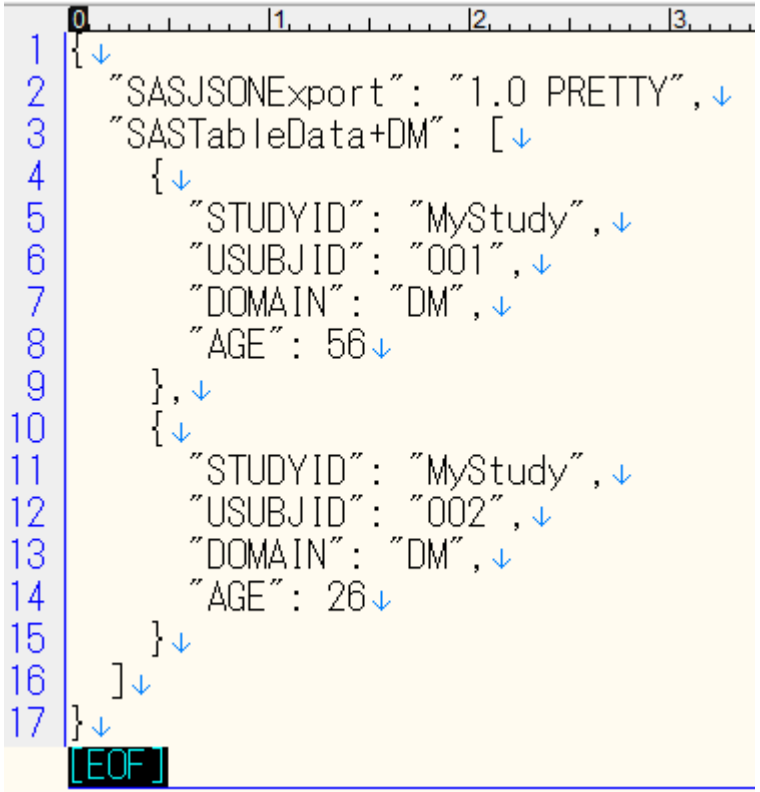
	STUDYID	USUBJID	DOMAIN	AGE
1	MyStudy	001	DM	56
2	MyStudy	002	DM	26

Creating Dataset-JSON Using proc JSON

export

```
proc json
out="xxxx¥Output4.json" pretty;
    export dm;
run;
```

Export SAS data sets to the JSON output file. The variable name/value objects in each observation are output as an array.

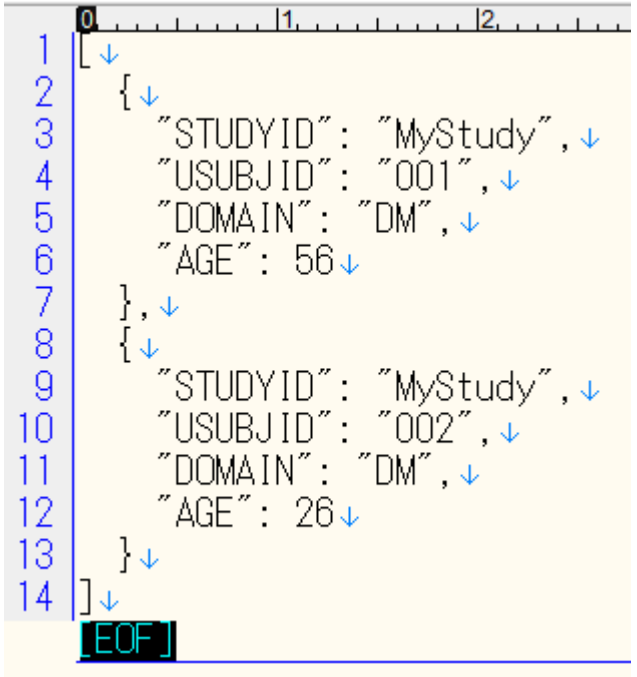


Creating Dataset-JSON Using proc JSON

export

```
proc json
out="xxxx¥Output5.json" pretty;
    export dm / nosastags;
run;
```

"nonsastags" suppresses the SAS export version, exported SAS data set name, and any non-default option specification.

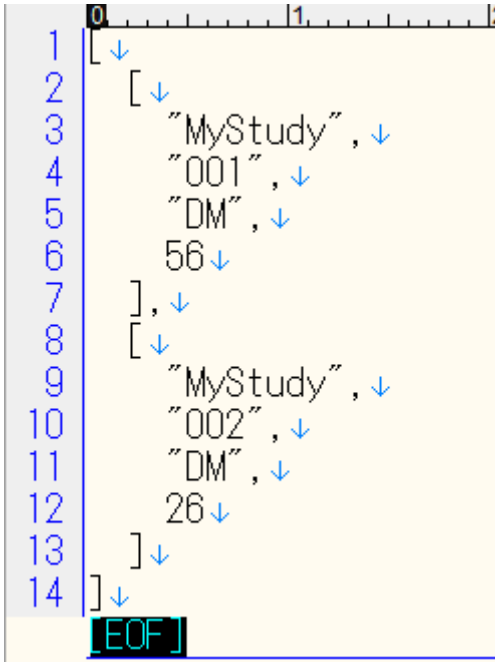


Creating Dataset-JSON Using proc JSON

export

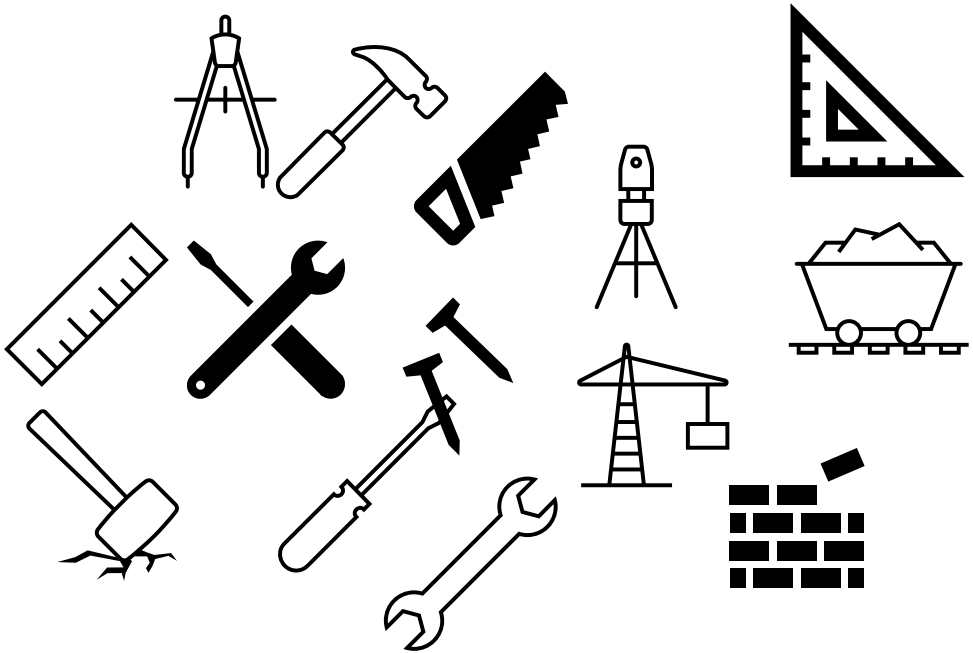
```
proc json
out="xxxx¥Output6.json" pretty;
    export dm / nosastags nokeys;
run;
```

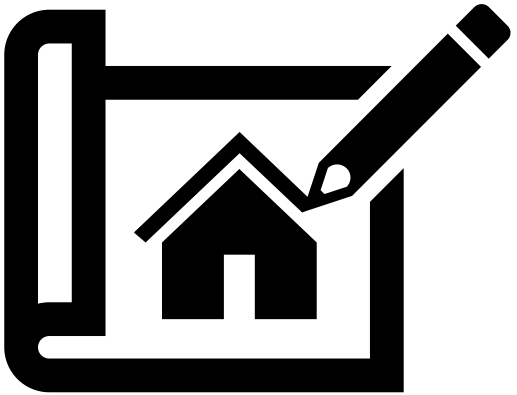
"nokeys" suppresses SAS variable names in observation data and export observations are written as JSON arrays.



Creating Dataset-JSON Using proc JSON

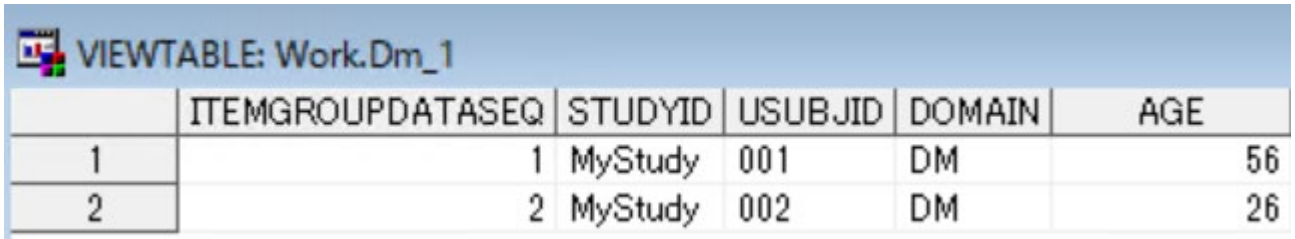
You can create Dataset-JSON with the tools introduced so far.





Creating Dataset-JSON Using proc JSON

```
data dm_1;
attrib ITEMGROUPDATASEQ
label="Record identifier"
length=8.;
 set dm;
 ITEMGROUPDATASEQ=_N_;
run;
```



VIEWTABLE: Work.Dm_1

	ITEMGROUPDATASEQ	STUDYID	USUBJID	DOMAIN	AGE
1	1	MyStudy	001	DM	56
2	2	MyStudy	002	DM	26

Creating Dataset-JSON Using proc JSON

```
ods noresults;
ods output Position=Position;
proc contents data=dm_1 varnum ;
run;

ods output EngineHost=EngineHost;
ods output Attributes=Attributes;
proc contents data= dm_1  ;
run;
ods results;
```

VIEWTABLE: Work.Position (Varnum)

	Member	Num	Variable	Type	Len	Label
1	WORK.DM_1	1	ITEMGROUPDATASEQ	Num	8	Record identifier
2	WORK.DM_1	2	STUDYID	Char	7	Study Identifier
3	WORK.DM_1					
4	WORK.DM_1					
5	WORK.DM_1					

VIEWTABLE: Work.Attributes (Attributes)

	Label1	cValue1
1	Data Set Name	WORK.DM_1
2	Member Type	DATA
3	Engine	
4	Created	
5	Last Modified	
6	Protection	
7	Data Set Type	
8	Label	
9	Data Representation	
10	Encoding	

ABLE: Work.Enginehost (Engine/Host Information)

Label1	cValue1
Data Set Page Size	65536
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	2039
Obs in First Data Page	2
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	XXXXXX¥dm_1.sas7bdat
Release Created	9.0401M7
Host Created	XX
Owner Name	XXXXXXXXX
File Size	128KB
File Size (bytes)	131072

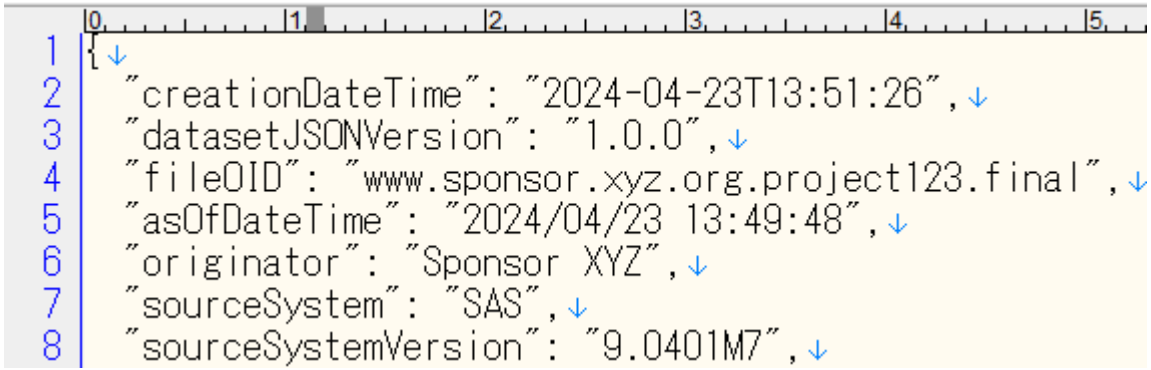
```
data _null_;
set Attributes;
if label1="Label" then call symputx("DS_LABEL",cValue1);
if label1="Last Modified" then call
symputx("asOfDateTime",cValue1); /*or CreationDateTime? */
run;

data _null_;
set Enginehost;
if label1="Release Created" then call
symputx("sourceSystemVersion",cValue1);
run;
```

```
%let originator=Sponsor XYZ;
%let fileOID=
www.sponsor.xyz.org.project123.final;
%let studyOID=M123-111;
%let metaDataVersionOID=1.0;
```

Creating Dataset-JSON Using proc JSON

```
proc json out = "&outpath.¥&L_dataset..json" pretty nofmtdatetime;
 write open object ;
   write values "creationDateTime"  "&creationDateTime";
   write values "datasetJSONVersion"  "1.0.0";
   write values "fileOID"  "&fileOID.";
   write values "asOfDateTime"  "&asOfDateTime";
   write values "originator"  "&originator";
   write values "sourceSystem"  "SAS";
   write values "sourceSystemVersion"  "&sourceSystemVersion";
```



```
{
  "creationDateTime": "2024-04-23T13:51:26",
  "datasetJSONVersion": "1.0.0",
  "fileOID": "www.sponsor.xyz.org.project123.final",
  "asOfDateTime": "2024/04/23 13:49:48",
  "originator": "Sponsor XYZ",
  "sourceSystem": "SAS",
  "sourceSystemVersion": "9.0401M7",
```

Creating Dataset-JSON Using proc JSON

```
write values "clinicalData";
write open object;
  write values "studyOID"  "&studyOID";
  write values "metaDataVersionOID"  "&metaDataVersionOID";
  write values "metaDataRef"  "https://metadata.location.org/api.link";
  write values "itemGroupData";
  write open object;
   write values "IG.&dataset";
    write open object;
     write values "records" &tot_obs. ;
     write values "name" "&dataset" ;
     write values "label" "&DS_LABEL" ;
```

```
    "studyOID": "M123-111",
    "metaDataVersionOID": "1.0",
    "metaDataRef": "https://metadata.location.org/api.link",
    "itemGroupData": {
      "IG.DM": {
        "records": 2,
        "name": "DM",
        "label": "Demographics",
```

Creating Dataset-JSON Using proc JSON

```
write values "items";
  write open array;
  export dataset_item / nosastags;
  write close;
```

BLE: Work.Dataset_item

OID	name	label	type
ITEMGROUPDATASEQ	ITEMGROUPDATASEQ	Record identifier	integer
IT.STUDYID	STUDYID	Study Identifier	string
IT.USUBJID	USUBJID	Unique Subject Identifier	string
IT.DOMAIN	DOMAIN	Domain Abbreviation	string
IT.AGE	AGE	Age	integer


```
{
  "OID": "ITEMGROUPDATASEQ",
  "name": "ITEMGROUPDATASEQ",
  "label": "Record identifier",
  "type": "integer",
  "displayFormat": ""
},
{
  "OID": "IT.STUDYID",
  "name": "STUDYID",
  "label": "Study Identifier",
  "type": "string",
  "displayFormat": ""
},
{
  "OID": "IT.USUBJID",
  "name": "USUBJID",
  "label": "Unique Subject Identifier",
  "type": "string",
  "displayFormat": ""
},
{
  "OID": "IT.DOMAIN",
  "name": "DOMAIN",
  "label": "Domain Abbreviation",
  "type": "string",
  "displayFormat": ""
},
{
  "OID": "IT.AGE",
  "name": "AGE",
  "label": "Age",
  "type": "integer",
  "displayFormat": ""
}
```

Creating Dataset-JSON Using proc JSON

```
write values "itemData";/* record array */
 write open array;
  export &dataset._1 / nokeys nosastags;
 write close;
```

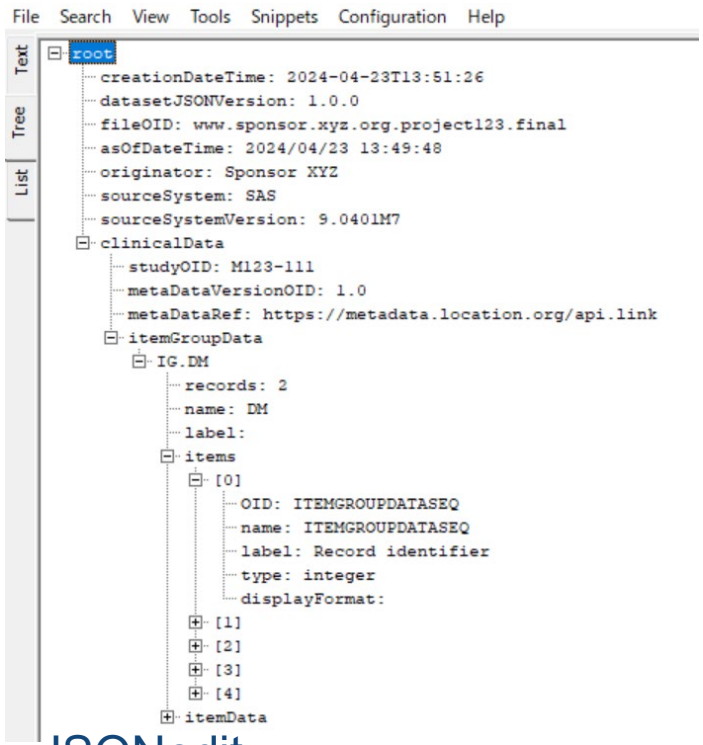
ABLE: Work.Dm_1

ITEMGROUPDATASEQ	STUDYID	USUBJID	DOMAIN	AGE
1	MyStudy	001	DM	56
2	MyStudy	002	DM	26

```
55        "itemData": [
56          [
57            1,
58            "MyStudy",
59            "001",
60            "DM",
61            56
62          ],
63          [
64            2,
65            "MyStudy",
66            "002",
67            "DM",
68            26
69          ]
70        ]
```

Creating Dataset-JSON Using proc JSON

Check the created JSON using a free viewer or browser functions

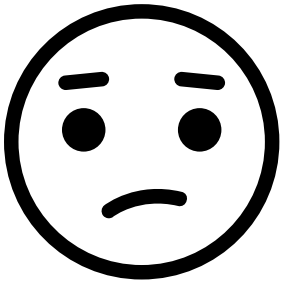


JSONedit

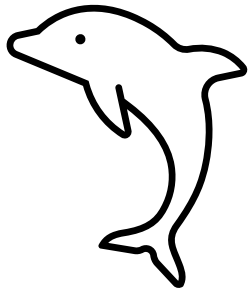
FireFox

Creating Dataset-JSON Using proc JSON

SAS dataset can not have enough metadata.



You need find the place to store metadata.



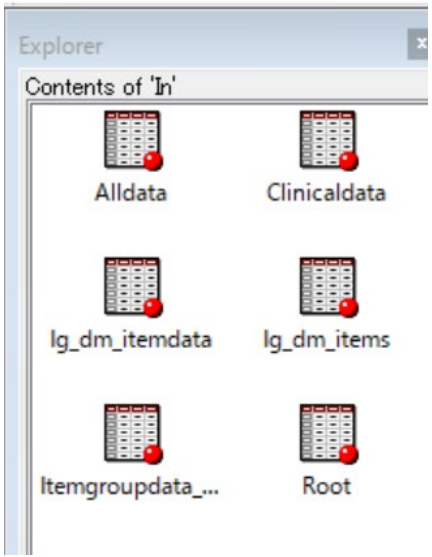
Import Dataset-JSON to SAS

Import Dataset-JSON to SAS

Later SAS 9.4TS1M4, you can use the libname JSON engine to read JSON files into SAS datasets.

```
filename js "XXXX¥dm.json" encoding="utf-8";
libname in json fileref=js;
```

```
proc copy in = in out = work;
run;
```



Import Dataset-JSON to SAS

Contains dataset attribute

LE: Work.Itemgroupdata_ig_dm

ordinal_itemGroupData	ordinal_IG_DM	records	name	label
1	1	2	DM	Demographics

Contains variable attribute

LE: Work.lg_dm_items

ordinal_IG_DM	ordinal_items	OID	name	label	type
1	1	IT.ITEMGROUPDATASE	ITEMGROUPDATASE	Record identifier	integer
1	2	IT.STUDYID	STUDYID	Study Identifier	string
1	3	IT.USUBJID	USUBJID	Unique Subject Identifier	string
1	4	IT.DOMAIN	DOMAIN	Domain Abbreviation	string
1	5	IT.AGE	AGE	Age	integer

Contains data

ABLE: Work.lg_dm_itemdata

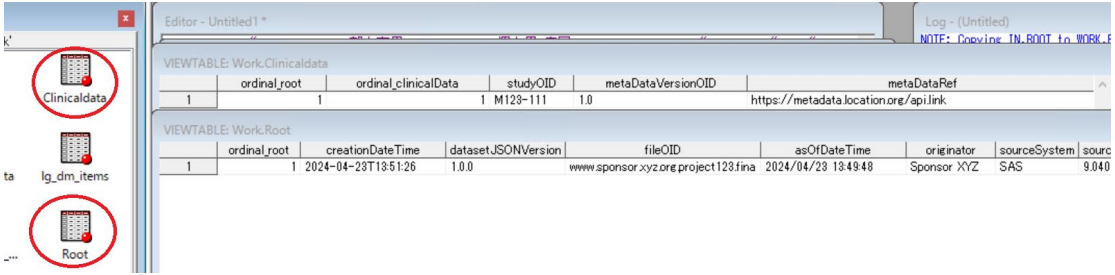
ordinal_IG_DM	ordinal_itemData	element1	element2	element3	element4	element5
1	1	1	MyStudy	001	DM	56
1	2	2	MyStudy	002	DM	26

Import Dataset-JSON to SAS

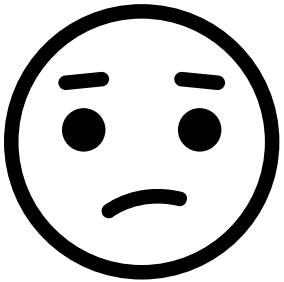
```
%macro imp_json(inpath=XXXX,ds=);
filename js "&inpath¥&ds..json" encoding="utf-8";
libname in json fileref=js ;
proc copy in = in out = work ;
run ;
libname in clear ;
filename js clear;
data _null_;
set Itemgroupdata_ig_&ds.;
call symputx("name",name);
call symputx("label",label);
run;
data _null_;
set Ig_&ds._items end=eof;
 call symputx( cats("rename",ordinal_items), cats("element",ordinal_items)||"="|| name );
 call symputx( cats("label",ordinal_items), cats("element",ordinal_items)||"="||
cats(label,"") );
 if ^missing(displayFormat) then do;
  call symputx( cats("format",ordinal_items), cats("element",ordinal_items)||" "||
cats(displayFormat,"") );
 end;
 else do;
  call symputx( cats("format",ordinal_items),"");
 end;
 if eof then call symputx("last_ordinal_items",ordinal_items);
run;
```

```
%macro create;
data &name(label="&label"
drop=ordinal_IG_&ds ordinal_itemData ITEMGROUPDATASEQ);
 set Ig_&ds._itemdata;
 rename
 %do i = 1 %to &last_ordinal_items;
  &&rename&i
 %end;
 ;
  label
 %do i = 1 %to &last_ordinal_items;
  &&label&i
 %end;
 ;;
 format
 %do i = 1 %to &last_ordinal_items;
  &&format&i
 %end;
 ;
 ;
 run;
 %mend create;
 %create;
%mend imp_json;
```

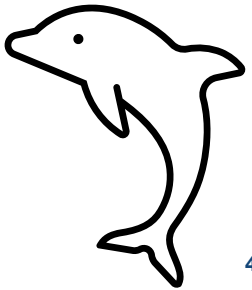
Import Dataset-JSON to SAS



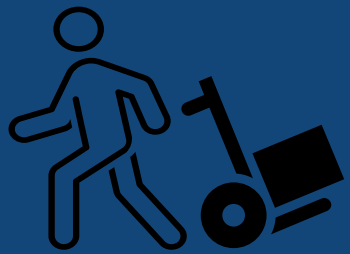
Too much information in Dataset-JSON, no place to store it in SAS dataset



You need find the place to store metadata.

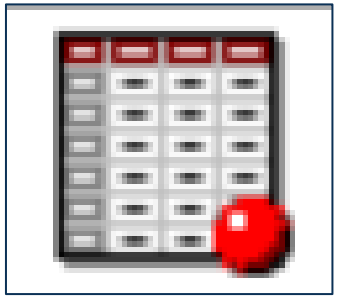


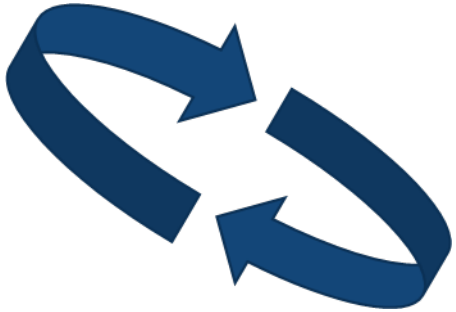
SAS Extended Attributes



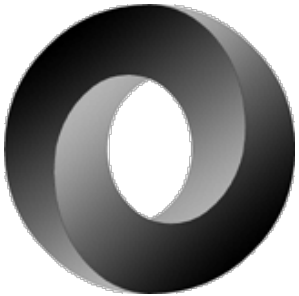
SAS Extended Attributes

SAS Dataset





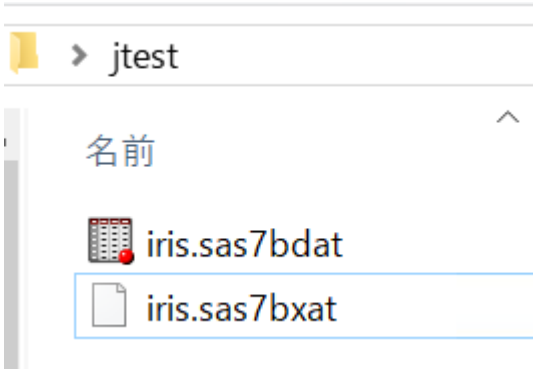
JSON

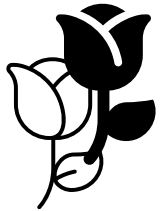


Need a receptacle for data leaked by conversion

SAS Extended Attributes

Starting with SAS 9.4, it is possible to set extended attributes that can be defined freely and arbitrarily by the user for a dataset or each variable. Extended attributes consist of a name/value pair, and the value can be of either a numeric or character type.





VIEWTABLE: Lib1.Iris

	Species	SepalLength	SepalWidth	PetalLength	PetalWidth
1	Setosa	50	33	14	2
2	Setosa	46	34	14	3
3	Setosa	46	36	10	2
4	Setosa	51	33	17	5
5	Setosa	55	35	13	2
6	Setosa	48	31	16	2
7	Setosa	52	34	14	2

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Label
4	PetalLength	Num	8	Petal Length (mm)
5	PetalWidth	Num	8	Petal Width (mm)
2	SepalLength	Num	8	Sepal Length (mm)
3	SepalWidth	Num	8	Sepal Width (mm)
1	Species	Char	10	Iris Species

Alphabetic List of Data Set Extended Attributes

Extended Attribute	Numeric Value	Character Value
disc	.	This data sets consists of 3 different types of irises' (Setosa, Versicolour, and Virginica) petal and sepal length

Alphabetic List of Extended Attributes on Variables

Extended Attribute	Attribute Variable	Numeric Value	Character Value
desc	SepalLength	.	one of the small leaves directly under a flower

The sas7bxat is associated with the dataset body and has extended meta information.

SAS Extended Attributes

I, Yutaka Morioka, have presented a method of using extended attributes for TLF compares at a SAS User's Group in Japan.

解析帳票出力用データセットのコンペア
に拡張属性を利用する方法

○森岡 裕
(イーピーエス株式会社 統計解析1部)

How to use extended attribute to compare datasets for TLF

Yutaka Morioka
Statistics Analysis Department 1, EPS Corporation

How to use extended attribute to compare datasets for TLF

解析帳票への応用

```
proc datasets nolist;
modify T14_1_X_X;
xattr add ds title1="Table 14.1.X.X 服薬遵守率"
             population="安全性解析対象集団"
             footnote1="※---脚注----";
xattr add var out3 (VARHEAD="EPS001#N=100#n(%)")
              out4(VARHEAD="プラセボ#N=105#n(%)");
run;quit;
```

データセットの拡張属性のリスト（アルファベット順）

拡張属性	数値	文字値
footnote1	.	※---脚注----
population	.	安全性解析対象集団
title1	.	Table 14.1.X.X 服薬遵守率

変数の拡張属性のリスト（アルファベット順）

拡張属性	属性変数	数値	文字値
VARHEAD	out3	.	EPS001#N=100#n(%)
VARHEAD	out4	.	プラセボ#N=105#n(%)

OUTPUTFILE.rtf

Table 14.1.X.X 服薬遵守率
安全性解析対象集団

		EPS001 N=100 n(%)	プラセボ N=105 n(%)
服薬率（%）	例数	92	97
	平均	93.29	94.15
	標準偏差	8.54	9.06
	最小値	44.3	52.7
	中央値	95.00	97.10
	最大値	100.0	100.0
	80%未満	0	2 (4.1)
	80%以上	100(100.0)	103 (95.9)

※---脚注----

T14_1_X_X.sas7bdat

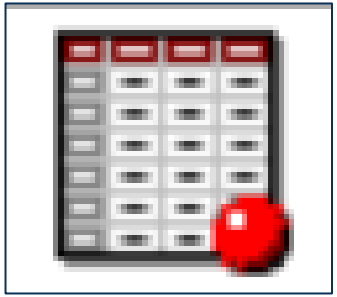
out1	out2	out3	out4
服薬率（%）	例数	92	97
	平均	93.29	94.15
	標準偏差	8.54	9.06
	最小値	44.3	52.7
	中央値	95.00	97.10
	最大値	100.0	100.0
	80%未満	0	2 (4.1)
	80%以上	100(100.0	103 (95.9)

16

SAS Extended Attributes How to Assign

```
proc datasets ;
modify dm;
xattr set ds
originator="Sponsor XYZ"
datasetJSONVersion="1.0.0"
studyOID="M123-111"
;
xattr set var
STUDYID (keySequence=1 type="string")
USUBJID (keySequence=2 type="string")
DOMAIN (type="string")
AGE (type="integer")
;
run;
quit;
```

DM



VIEWTABLE: Work.Dm

	STUDYID	USUBJID	DOMAIN	AGE
1	MyStudy	001	DM	56
2	MyStudy	002	DM	26

Use proc datasets to add extended attributes to SAS datasets

SAS Extended Attributes

How to take out – method 1

```
ods output ExtendedAttributesDS=EXDS;
ods output ExtendedAttributesVar=EXVAR;
proc contents data=dm;
run;
```

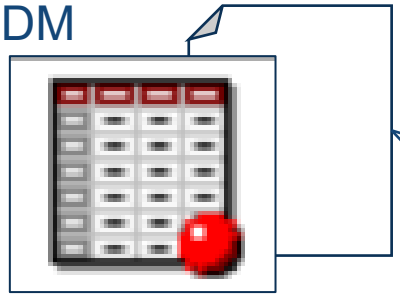


TABLE: Work.Exds (Alphabetic List of Data Set Extended Attributes)

Member	ExtendedAttribute	AttributeCharValue
WORK.DM	datasetJSONVersion	1.0.0
WORK.DM	originator	Sponsor XYZ
WORK.DM	studyOID	M123-111

TABLE: Work.Exvar (Alphabetic List of Extended Attributes on Variables)

Member	ExtendedAttribute	AttributeVariable	AttributeCharValue	AttributeNumValue
WORK.DM	keySequence	STUDYID		
WORK.DM	keySequence	USUBJID		
WORK.DM	type	AGE	integer	
WORK.DM	type	DOMAIN	string	
WORK.DM	type	STUDYID	string	
WORK.DM	type	USUBJID	string	

Alphabetic List of Data Set Extended Attributes

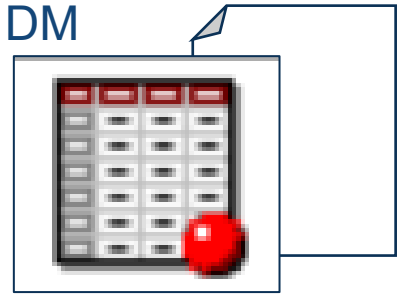
Extended Attribute	Numeric Value	Character Value
datasetJSONVersion	.	1.0.0
originator	.	Sponsor XYZ
studyOID	.	M123-111

Alphabetic List of Extended Attributes on Variables

Extended Attribute	Attribute Variable	Numeric Value	Character Value
keySequence	STUDYID	1	
keySequence	USUBJID	2	
type	AGE	.	integer
type	DOMAIN	.	string
type	STUDYID	.	string
type	USUBJID	.	string

SAS Extended Attributes

How to take out – method 2




```
proc sql;
 create table XATR as
  select *
  from dictionary.xattrs
  where memname ="DM";
quit;
```

```
data XATR;
set sashelp.vxattr;
where memname ="DM";
run;
```

VIEWTABLE: Work.Xatr

	libname	memname	name	xattr	xtype	xoffset	xvalue
1	WORK	DM		datasetJSONVersion	char	0	1.0.0
2	WORK	DM		originator	char	0	Sponsor XYZ
3	WORK	DM		studyOID	char	0	M123-111
4	WORK	DM	STUDYID	type	char	0	string
5	WORK	DM	STUDYID	keySequence	num	0	1
6	WORK	DM	USUBJID	type	char	0	string
7	WORK	DM	USUBJID	keySequence	num	0	2
8	WORK	DM	DOMAIN	type	char	0	string
9	WORK	DM	AGE	type	char	0	integer

Dataset-Extend attribute

Variable-Extend attribute

SAS Extended Attributes

EXTENDED ATTRIBUTES: A New Metadata Creation Feature in SAS® 9.4 for Data Sets and Variables

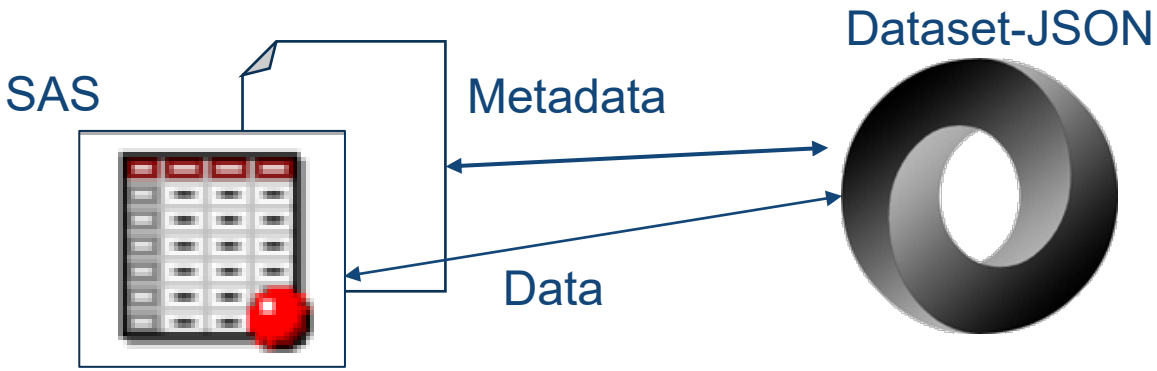
Joseph Hinson, inVentiv Health, Princeton, NJ, USA

The idea of embedding CDISC metadata as SAS extended attributes in SAS® data sets has already been presented in the past.

Harmonization with define.xml

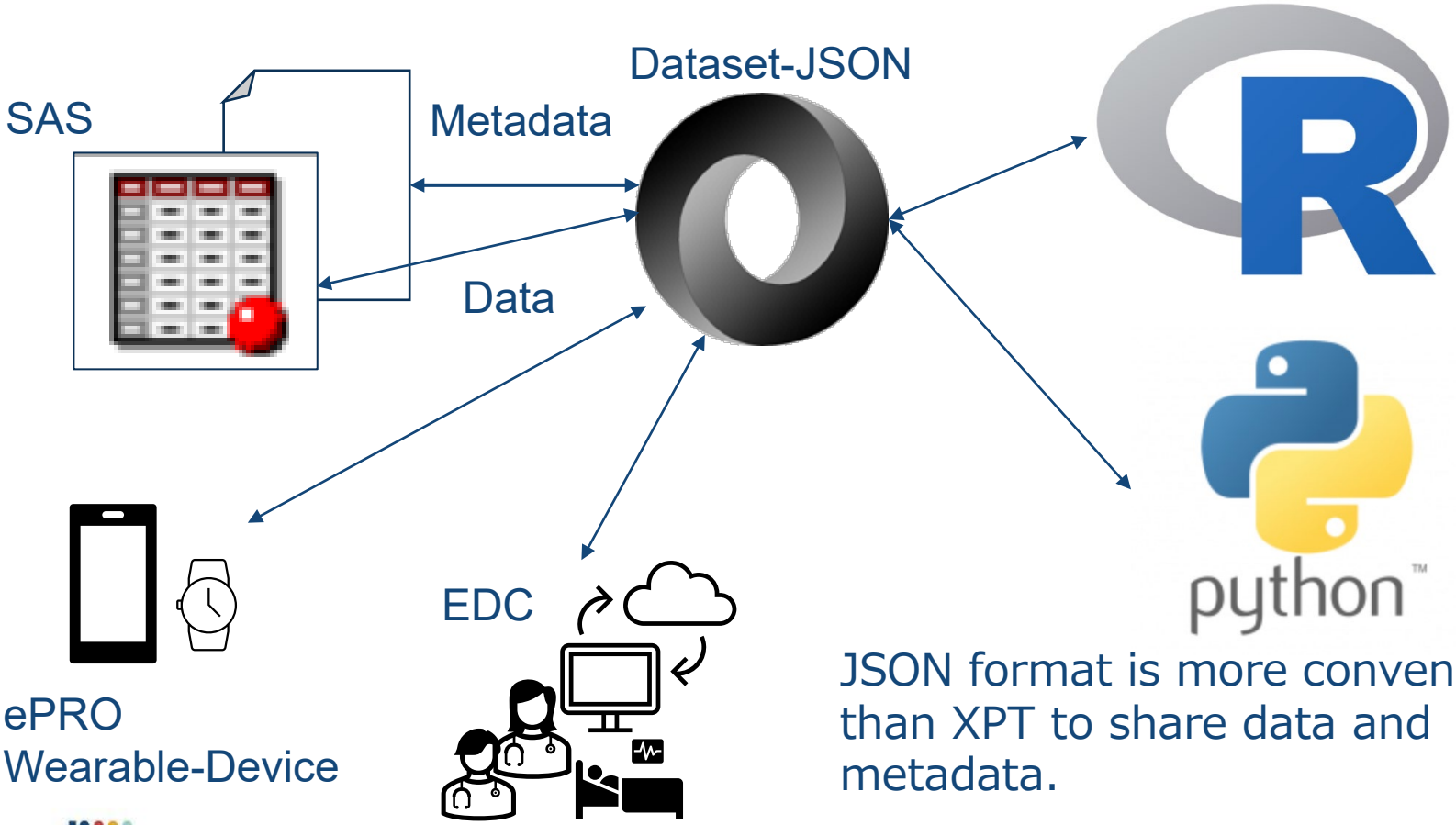
Harmonization with define.xml and other applications

During clinical trial



During clinical trial, the define.xml file is not yet complete.

Harmonization with define.xml and other applications



JSON format is more convenient than XPT to share data and metadata.

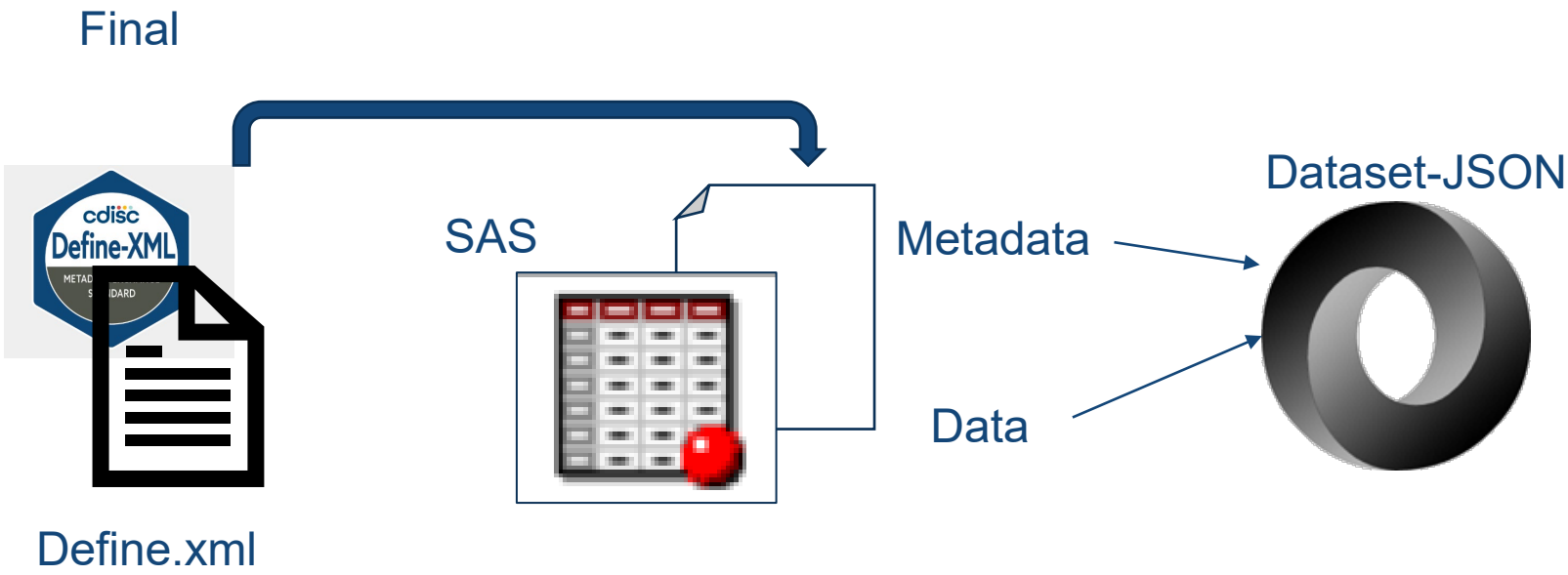
Harmonization with define.xml and other applications

https://atorus-research.github.io/datasetjson/

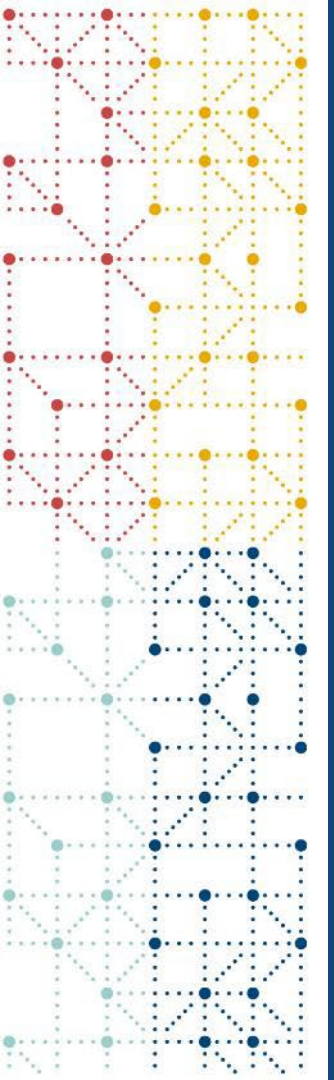
datasetjson

Welcome to **datasetjson**. **datasetjson** is an R package built to read and write CDISC Dataset JSON formatted datasets.

Harmonization with define.xml and other applications



1) Get metadata from the fixed Define.xml and store it as extended attributes.
2) Create Dataset -JSON from SAS dataset with extended attributes.



Conclusion

Conclusion

Dataset-JSON can be...

- a useful alternative to SAS version 5 transport files
- easily created using SAS Proc JSON
- easily converted from SAS datasets using SAS libname's JSON engine

The extended attributes can be a bridge between SAS datasets and Dataset-JSON in the difference in the metadata.

Acknowledgement

- PHUSE OST WG, Dataset-JSON sub team
  - **Yuichi Nakajima**
  - **Ippei Akiya**
  - **Tomoyuki Namai**

- Working groups in EPS
  - **Marie Kono**
  - **Katsumi Tohyama**
  - **Kohei Ohtani**
  - **Toshimune Yamada**
  - **Mai Sugaya**
  - **Terumasa Hyodo**

References

- A Short History of CDISC and SAS Transport Files A Short History of CDISC and SAS Transport Files
- CDISC Dataset-JSON Dataset-JSON | CDISC
- Dataset-JSON v1.1 Dataset-JSON v1.1 - Dataset-JSON v1.1 - Wiki (cdisc.org)
- JSONedit JSONedit - JSON editor
- Yutaka Morioka, EPS Corp.
  How to use extended attribute to compare datasets for TLF
- Joseph Hinson, inVentiv Health, Princeton, NJ, USA
  EXTENDED ATTRIBUTES: A New Metadata Creation Feature in SAS® 9.4 for Data Sets and Variables
- Lex Jansen, Sr. Director, Data Science Development, CDISC
  Working with Dataset-JSON using SAS©

Thank You!