



Metadata-driven Creation of TFL Programs by Using JSON Data File

Linshan Yuan, Sheng-Wei Wang, Yingshan You

Feb 2023

Sebastian Ferreira, *Untitled 1*

Artwork from the National Art Exhibitions of the Mentally III, Inc (NAEMI).

janssen

PHARMACEUTICAL COMPANY OF
Johnson & Johnson



Agenda

- ❑ Background & Strategies
- ❑ Process flow chart
- ❑ JSON file data structure
- ❑ Two R standard scripts: open code, modularized code
- ❑ Python drivers to auto generate R programs
- ❑ Conclusion

Background & Strategies

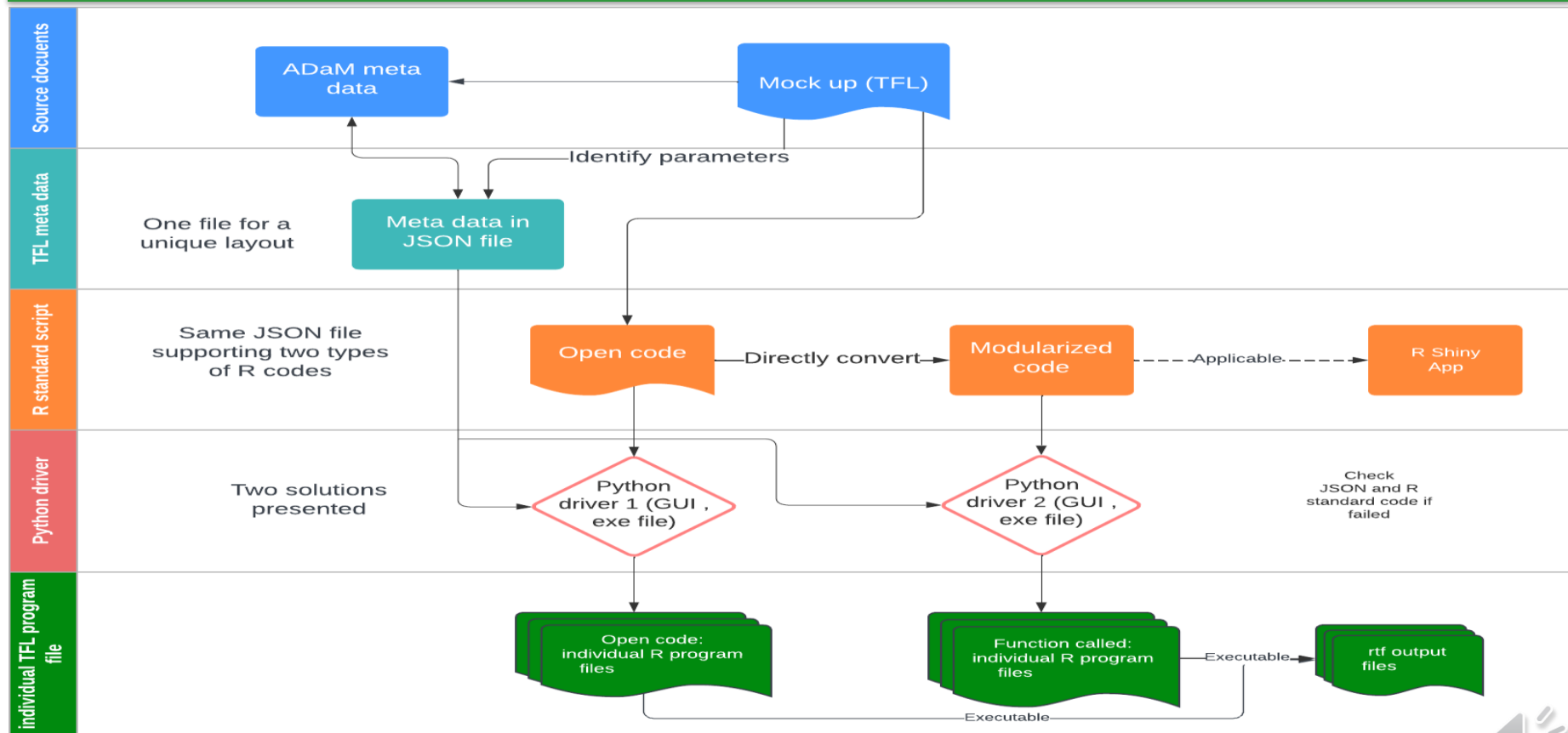
❑ Background:

- ✓ R Adaptive study has large amount of TFLs with over 300.
- ✓ Most team members are the first time to use R for TFL development.
- ✓ Uncertainty for quality or consistence of TFLs.
- ✓ Pressure on ad hoc request for quick turn around.

❑ Strategies:

- ✓ Divide TFL development into two efforts: TFL meta data ; standard R code
- ✓ TFL Meta data: identify parameters/augments per mockup.
- ✓ Standard R code: open code and modularized code
 - I. Exploring two types of R codes style confronting different scenario or stages
 - II. utilizing reporting packages : tidyTLG (in-house development), etc
- ✓ Python Script: integrating both information from the meta data and standard R program and creating the individual R programs automatically.

Process Flow Chart



JSON file data structure

- Language-independent data format & Hierarchy information
 - Standard format for data storage and transport for interchange
 - Commonly used in many program languages
 - Easily edited: less formatting
 - Good alternative for XML: carry more information than flat file (excel, etc)
 - Supporting various data structure as needed
- Carry specified parameters information of TFLs for
 - Key: table ID
 - Value: set of parameters and meta data information
 - Included the meta data of all outputs having same layout
 - In this presentation: one JASON file for one unique layout

JSON file: layout #1

```
{
  "tefpgis02d": {
    "author": "Linshan Yuan",
    "popfl": "FASFL",
    "poplabel": "Analysis set: Full Analysis",
    "treatc": "TRT01P",
    "treatn": "TRT01PN",
    "adslfilter": "expr(FASFL == 'Y' & TRT01PN %in% c(1,2))",
    "data1": "adqsoth.sas7bdat",
    "filter1": "expr(TRT01PN %in% c(1,2) & PARCAT1=='PGIS' & PARAMCD=='PGISSUM' & ANL02FL=='Y' & 'tbltype' : 'CFB1'",
    "tbltype": "CFB1"
  },
  "tefpgis04": {
    "author": "Linshan Yuan",
    "popfl": "FASOLFL",
    "poplabel": "Analysis set: Full(OL) Analysis",
    "treatc": "TRTSEQP",
    "treatn": "TRTSEQPN",
    "adslfilter": "expr(FASOLFL == 'Y' & TRTSEQPN %in% c(1,2,3))",
    "data1": "adqsoth.sas7bdat",
    "filter1": "expr(TRTSEQPN %in% c(1,2,3) & PARCAT1=='PGIS' & PARAMCD=='PGISSUM' & ANL03FL=='Y' & 'tbltype' : 'CFB1'",
    "tbltype": "CFB1"
  }
}
```

TEFPGIS02d: Patient Global Impression of Severity (PGI-S) Total Score: Mean Phase; Full Analysis Set

	Visit Values						
	N	Mean	SD	SE	Med	Min	Max
Analysis set: Full Analysis	###						
Dummy1	###						
Dummy2	###						
Dummy1							
Baseline	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 2	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 4	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 8	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 12	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 16	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 20	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 22	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 24	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX

Dummy2

Baseline	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 2	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 4	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 8	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX
Week 12	###	XX.X	XX.XX	XX.XX	XX.X	XX	XX

This JSON example includes 22 tables and is extendable.

JSON file: layout #2

```

{
  "tefpgis01d": {
    "author": "Linshan Yuan",
    "popfl": "FASFL",
    "poplabel": "Analysis set: Full Analysis",
    "treatc": "TRT01P",
    "treatn": "TRT01PN",
    "catvar": "AVALCAT1",
    "adslfilter": "exp((FASFL == 'Y' & TRT01PN %in% c(1,2)) ",
    "data1": "adqsoth.sas7bdat",
    "filter1": "exp(TRT01PN %in% c(1,2) & PARCAT1='PGTS' & ANL02FL='Y' & !is_na(AVALCAT1) & ! str_detect(AVISIT, 'Unsc') & ! str_detect(AVISIT, 'Day 1'))",
    "lookup": "c('Very Severe'=5, 'Severe'=4, 'Moderate'=3, 'Mild'=2, 'None'=1, 'Total'=99)",
    "alien_col": "exp(case when col == 1 ~ 4, col == 4 ~ 2, TRUE ~ as.double(col))",
    "pad_zero_pop_cols": "c('col1', 'col4')",
    "paramlabel": "c('Severity'='PGI Severity')",
    "tbltype": "EF_CUM"
  }
}

```

This JSON example includes 23 tables and is extendable.

- Different population
- Different source datasets
- Different variables for categories: AVALC, AVALCAT1, CHGCAT1, PCHGCAT1
- Label ordering
- Value format

TEFPGIS01d: Patient Global Impression of Severity (PGI-S): Frequency Distribution Over Time in Double-blind Phase; Full Analysis Set

	dummy1			dummy2		
	n	%	Cum %	n	%	Cum %
Analysis set: Full	###			###		
[Severity of illness]						
[Baseline]						
None	###	XX.X	XX.X	###	XX.X	XX.X
Mild	###	XX.X	XX.X	###	XX.X	XX.X
Moderate	###	XX.X	XX.X	###	XX.X	XX.X
Severe	###	XX.X	XX.X	###	XX.X	XX.X
Very severe	###	XX.X	XX.X	###	XX.X	XX.X
...	###	XX.X	XX.X	###	XX.X	XX.X
-----				----		
Total	###			###		
[Time Point]						
None	###	XX.X	XX.X	###	XX.X	XX.X

Standard R program: solution 1: open code

- Standard open code: parameters at designated area for input from JSON file and main code using only parameters defined by JSON file.
- Individual R programs are generated as open code too.

```
tchq_01.r x tefpgis02d.r x
Source on Save
1 #/*****
2 # Reporting Effort:
3 # Program Name : tefmgadl05a.R
4 # R version : 4.0.5
5 # Short Description :
6 # Author : Linshan Yuan
7 # Date : 03 JUNE 2022
8 # Input : a in.adgsadl
9 # Output : tefmgadl05a.rtf
10 # Remarks :
11 # Macro parameters :
12 # Macro Sample Call :
13 # Modification History :
14 # Rev # Modified By Reporting Effort Date Description
15 #*****/
16
17
18
19 #-----libraries-----
20
21 library(dplyr)
22 library(haven)
23 library(tidytlg)
24 library(stringr)
25
26
27 #@update: define the population flag-----
28 popfl <- "EFFFL"
29 poplabel<- "Analysis set: Primary Efficacy Analysis"
30 treatc<- "TRT01P"
31 treatnc<- "TRT01PN"
32 adslfilter<- expr(!ensym(popfl) == 'Y' & !ensym(treatnc) %in% c(1,2))
33 data1<- 'adgsadl.sas7bdat'
34 filter1<-expr(!ensym(treatnc) %in% c(1,2) & PARCAT1=='MG-ADL' & PARCAT1!='MG-ADL')
35
36 ## define the ID of TIGs
37 tblad <- "TEFMGADL05a"
38 tbltype<- "CFB1"
39
40 #@endupdate
41
42 #-----process data-----
43 ## apply population filter
44 adsl<-read_sas(read_path(a_in,"adsl.sas7bdat"))%>%
45   rename_all(toupper) %>%
46   mutate_if(is.character, ~ na_if(x, c(''))) %>% ## important
47   filter(eval(adslfilter)) %>%
48   mutate(!ensym(treatc) := forcats::fct_reorder(!ensym(treatc))
```

```
{
  "tefpgis02d":
  {
    "author": "Linshan Yuan",
    "popfl": "FASFL",
    "poplabel": "Analysis set: Full Analysis",
    "treatc": "TRT01P",
    "treatnc": "TRT01PN",
    "adslfilter": "expr(FASFL == 'Y' & TRT01PN %in% c(1,2))",
    "data1": "adqsoth.sas7bdat",
    "filter1": "expr(TRT01PN %in% c(1,2) & PARCAT1=='PGIS' & PARCAT1!='PGIS')",
    "tbltype": "CFB1"
  }
}
```

Python script (driver)

```
g_01.r x tefpgis02d.r x
Source on Save
1 #/*****
2 # Reporting Effort: jnj-xxxxxx/study-000/re_csr
3 # Program Name : tefpgis02d.R
4 # R version : 4.0.5
5 # Short Description : Patient Global Impression of Severity
6 # Author : Linshan Yuan
7 # Date : 2023-02-20
8 # Input : adqsoth.sas7bdat
9 # Output : tefpgis02d.rtf
10 # Remarks :
11 # Macro Parameters :
12 # Macro Sample Call :
13 # Modification History :
14 # Rev # Modified By Reporting Effort Date Description
15 #*****/
16
17
18
19 #-----libraries-----
20
21 library(dplyr)
22 library(haven)
23 library(tidytlg)
24 library(stringr)
25
26
27 #@update: define the population flag-----
28 popfl<- "FASFL"
29 poplabel<- "Analysis set: Full Analysis"
30 treatc<- "TRT01P"
31 treatnc<- "TRT01PN"
32 adslfilter<- expr(FASFL == 'Y' & TRT01PN %in% c(1,2))
33 data1<- "adqsoth.sas7bdat"
34 filter1<- expr(TRT01PN %in% c(1,2) & PARCAT1=='PGIS' & PARCAT1!='PGIS')
35 tbltype<- "CFB1"
36 tblad <- "TEFPGIS02d"
37
38 #@endupdate
39
40 #-----process data-----
41 ## apply population filter
42 adsl<-read_sas(read_path(a_in,"adsl.sas7bdat"))%>%
43   rename_all(toupper) %>%
44   mutate_if(is.character, ~ na_if(x, c(''))) %>% ## important
45   filter(eval(adslfilter)) %>%
46   mutate(!ensym(treatc) := forcats::fct_reorder(!ensym(treatc))
```

What Python driver does:

- Read table ID parameters from JSON and repeating for each table ID within it
- For each table ID : creating new file by reading R standard program as template line by line and taking actions as below:
 - Header : auto update, such as author, date, source data, program name and output
 - Parameters: auto turn into R statements at the specific location .
 - Rest of R codes: take codes without change.

Python driver 1 (GUI): open code

R TFL program driver panel

Compound/Study/Report: jnj-xxxxx/study-000/re_csr update to program header

TFL meta data path (Jason file): C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01\tchg_01.json

Mockup standard program path (R file): C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01\tchg_01.r

TFL programs path: C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01

Titles path (csv file): C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01\titles.csv update to program header

Generate TFL program

Reset

Generated TFL program list:

TFL programs created under: C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01

```
1: tefmgad105a.r
2: tefmgad105c.r
3: tefmgad105d.r
4: tefmgad127.r
5: tefmgad128.r
6: tefmgqol03a.r
7: tefmgqol03d.r
8: tefmgqol04.r
9: tefmgqol05.r
10: tefnrqol03a.r
11: tefnrqol03d.r
12: tefnrqol10.r
13: tefnrqol11.r
14: tefpgis02a.r
15: tefpgis02d.r
16: tefpgis03.r
17: tefpgis04.r
18: tefqmg05a.r
19: tefqmg05c.r
20: tefqmg05d.r
21: tefqmg10.r
22: tefqmg11.r
```



Python driver demo

Standard R program: solution 2: Modularized code

- Convert open code into function definition

```
tchg_01.r x tefpgis02d.r x
Source on Save
37 tblid <- "TEFMGADL05a"
38 tbltype<- "CFB1"
39
40 #'@update
41
42 #-----process data-----
43 ## apply population filter
44 adsl<-read_sas(read_path(a_in,"adsl.sas7bdat"))%>%
45   rename_all(toupper) %>%
46   mutate_if( is.character, ~ na_if(., c(''))) %>%
47   filter(eval(adslfilter)) %>%
48   mutate(!ensym(treatc) := forcats::fct_reorder(
49
```



```
g_01.r x tefpgis02d.r x tchg_01_fundef.r x tefrq_mdd01
Source on Save
tchg_mdd01 <- function (
  #'@update: define the parameters-----
  popfl,
  poplabel,
  treatc,
  treatn,
  adslfilter,
  data1,
  filter1,
  tblid,
  tbltype
  #'@endupdate -----
)
{
  #-----process data-----
  ## apply population filter
  adsl<-read_sas(read_path(a_in,"adsl.sas7bdat"))%>%
    rename_all(toupper) %>%
    mutate_if( is.character, ~ na_if(., c(''))) %>%
    filter(eval(adslfilter)) %>%
    mutate(!ensym(treatc) := forcats::fct_reorder(!

```

Standard R program: solution 2: Modularized code

- Standard modularized code is on left side
- The generated Individual R program will have predefined function with parameters from the same JSON file.

```
tchg_01_fun.r x tchg_01_r x tefpgis02d.r x tchg_01_fundef.r x
Source on Save
1 #/*****
2 # Reporting Effort:
3 # Program Name :
4 # R version : 4.0.5
5 # Short Description:
6 # Author :
7 # Date :
8 # Input :
9 # Output :
10 # Remarks :
11 # Macro Names :
12 # Macro Parameters :
13 # Modification History :
14 # Rev # Modified By Reporting Effort Date Description
15 #****
16
17 #-----libraries-----
18
19 library(dplyr)
20 library(haven)
21 library(tidytlg)
22 library(stringr)
23
24
25 source('C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode
26
27 #-----process data-----
28
29 tchg_mdd01 (
30
31 #'@update: define the parameters-----
32
33
34 #'@endupdate -----
35
36 )
37
```

```
{
  "tefpgis02d":
  {
    "author": "Linsan Yuan",
    "popfl": "FASFL",
    "poplabel": "Analysis set: Full Analysis",
    "treatc": "TRT01P",
    "treatn": "TRT01PN",
    "adslfilter": "expr(FASFL == 'Y' & TRT01PN %in% c(1,2))",
    "data1": "adqsoth.sas7bdat",
    "filter1": "expr(TRT01PN %in% c(1,2) & PARCAT1=='PGIS' & P
    "tbltype": "CFB1"
  }
}
```



Python script (driver)

```
chg_01_fun.r x tefpgis02d.r x
Source on Save
1 #/*****
2 # Reporting Effort: JNJ-xxxxxx/study-000/re csp
3 # Program Name: tefpgis02d.R
4 # R version : 4.0.5
5 # Short Description: Patient Global Impression of Sev
6 # Author: Linsan Yuan
7 # Date: 2023-02-20
8 # Input: adqsoth.sas7bdat
9 # Output: tefpgis02d.rtf
10 # Remarks :
11 # Macro Names: tchg_mdd01
12 # Macro Parameters: tchg_mdd01(popfl, poplabel, treat
13 # Modification History :
14 # Rev # Modified By Reporting Effort Date Description
15 #****
16
17 #-----libraries-----
18
19 library(dplyr)
20 library(haven)
21 library(tidytlg)
22 library(stringr)
23
24 source('C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode
25
26 #-----process data-----
27
28 tchg_mdd01 (
29
30 #'@update: define the parameters-----
31
32 popfl= "FASFL",
33 poplabel= "Analysis set: Full Analysis",
34 treatc= "TRT01P",
35 treatn= "TRT01PN",
36 adslfilter= expr(FASFL == 'Y' & TRT01PN %in% c(1,2)),
37 data1= "adqsoth.sas7bdat",
38 filter1= expr(TRT01PN %in% c(1,2) & PARCAT1=='PGIS' &
39 tbltype= "CFB1"
40 tblid = "TEFPGIS02d"
41
42 #'@endupdate -----
43
44 )
45
```

Python driver 2 (GUI): modularized code

R TFL program driver pane (modularized code)

Compound/Study/Report: jnj-xxxxxx/study-000/re_csr

TFL meta data path (Jason file): C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01_fun\tchg_01.json

Mockup standard program path (R file): C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01_fun\tchg_01_fun.r

TFL programs path: C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01_fun

Titles path (csv file): C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01_fun\titles.csv

Generate TFL program

Reset

Same JSON file used
for open code

Generated TFL program list:

TFL programs created under: C:\Users\lyuan7\OneDrive - JNJ\Documents\mycode\python\PHUSE2023\Paper1\tchg_01_fun

```
1: tefmgad105a.r
2: tefmgad105c.r
3: tefmgad105d.r
4: tefmgad127.r
5: tefmgad128.r
6: tefmgqo103a.r
7: tefmgqo103d.r
8: tefmgqo104.r
9: tefmgqo105.r
10: tefnrqo103a.r
11: tefnrqo103d.r
12: tefnrqo110.r
13: tefnrqo111.r
14: tefpgis02a.r
15: tefpgis02d.r
16: tefpgis03.r
17: tefpgis04.r
18: tefqmg05a.r
19: tefqmg05c.r
20: tefqmg05d.r
21: tefqmg10.r
22: tefqmg11.r
```



How to update :

TEFMGADL30:

Total Score: Frequency Distribution of Improvement Over Time

Through Open-label Phase; Full Analysis Set (OL) Analysis Set

	dummy1			dummy2			dummy3		
	n	%	Cum %	n	%	Cum %	n	%	Cum %
Analysis set: Full(OL) Analysis	7			4			15		

ADSL

xxxxxxx Total Score

OL Week 2

>=8	2	40.0	40.0	0	0.0	0.0	2	13.3	13.3
7	0	0.0	40.0	0	0.0	0.0	1	6.7	20.0
6	0	0.0	40.0	0	0.0	0.0	0	0.0	20.0
5	1	20.0	60.0	1	33.3	33.3	1	6.7	26.7
4	0	0.0	60.0	0	0.0	33.3	0	0.0	26.7
3	0	0.0	60.0	1	33.3	66.7	2	13.3	40.0
2	1	20.0	80.0	0	0.0	66.7	0	0.0	40.0
<2	1	20.0	100.0	1	33.3	100.0	7	46.7	86.7
Total	5			3			15		

OL Week 4

>=8	1	20.0	20.0	1	25.0	25.0	3	20.0	20.0
7	1	20.0	40.0	1	25.0	50.0	1	6.7	26.7

column_metadata.xlsx

"tbltype" : "EF_CUM2"

"catvar": "CHGCAT1",

"filter1": "expr(TRTSEQPN %in% c(1,2,3) & PARCAT1=='MG-ADL' & PARAMCD=='SUM' & ANL03FL=='Y' & is.na(ABLFL))",
 "lookup": "c('<2'=8,'2'=7,'3'=6,'4'=5,'5'=4,'6'=3,'7'=2,'>=8'=1,'Total'=99)",
 "align_col": "expr(case_when(col == '2' ~ 4, col == '3' ~ 7, col == '4' ~ 2, col == '7' ~ 3, TRUE ~ as.double(col)))",

easy update in JSON for key of 'filter1' with additional condition appended :
 & !is.na(CHGCAT1)

Summary

- The success of generate executable individual R program rely on the completed output meta data in JSON file and well-established standard R code.
- Build up robust standard code will drastically improve the efficiency with auto generation feature supported.
- The stable open code could be the basis to turn into modularized/function which could be useful for Shiny app setup future.
- With this strategy, we have over 300 TFL generated in R in a study and the number is still growing.

Q & A



The opinions in this presentation are my own and do not necessarily reflect the views and policies of J&J

