

Retrospective clinical data harmonisation reporting

Jeremy Selva 

@JauntyJJS    

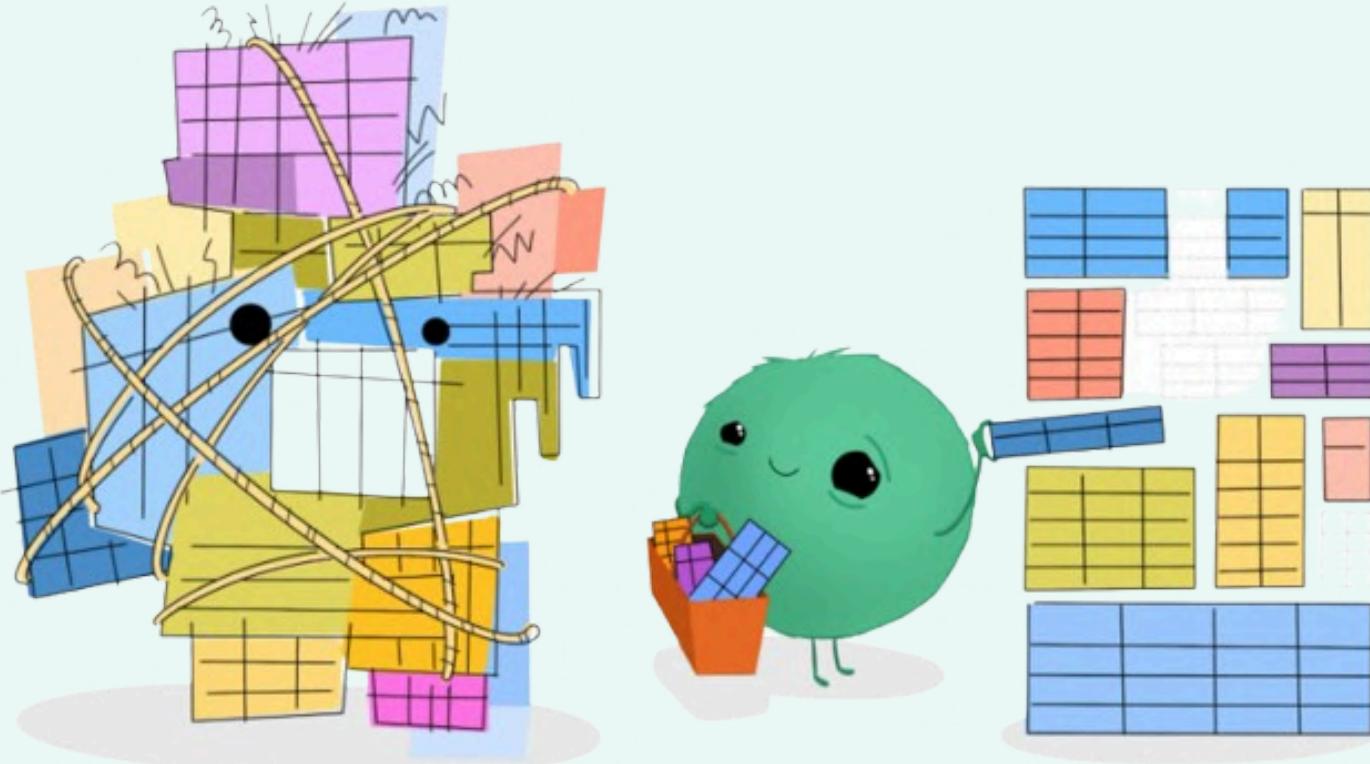
<https://jeremy-selva.netlify.app> 

For R/Medicine 2025 

12th June 2025

whoami

Research Officer from [National Heart Centre Singapore](#) who collects, cleans and harmonises clinical data.



Taming the Data Beast from “[Cleaning Medical Data with R](#)” workshop by Shannon Pileggi, Crystal Lewis and Petter Higgins presented at R/Medicine 2023. Illustrated by [Allison Horst](#).

Outline

Data harmonisation overview strategy

Counter small but annoying issues during retrospective data harmonisation.

Suggested reports and diagram to create for different clients.



Photo by [Sixteen Miles Out on Unsplash](#)

About Data Harmonisation

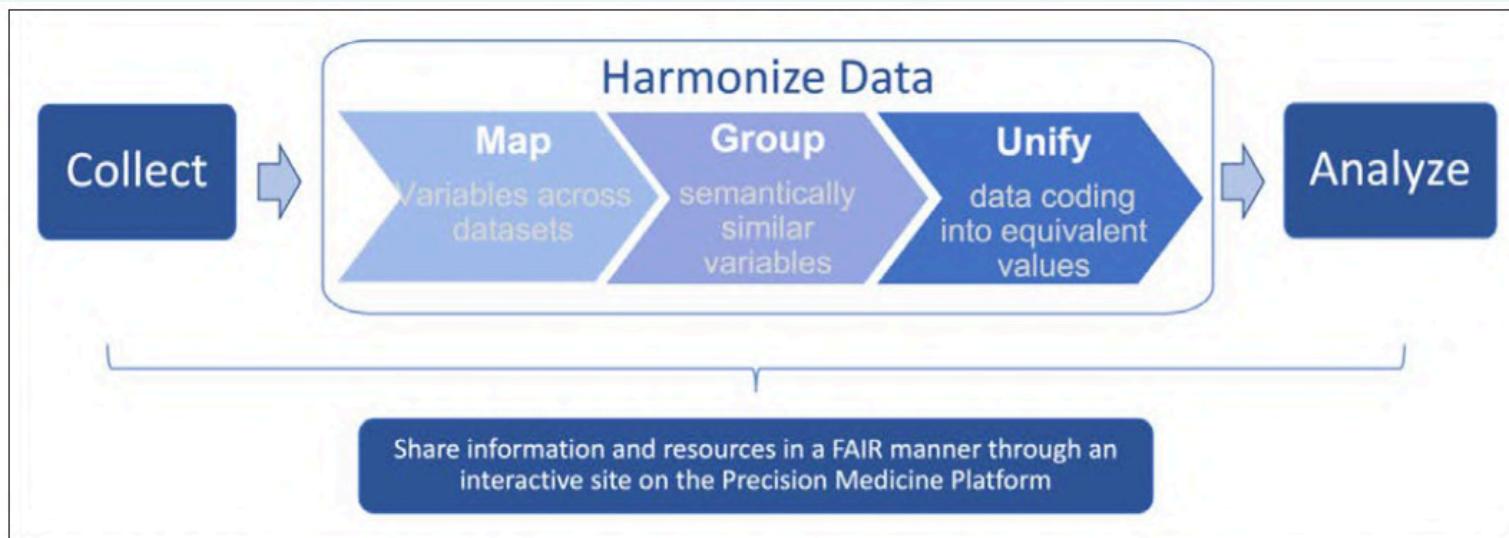


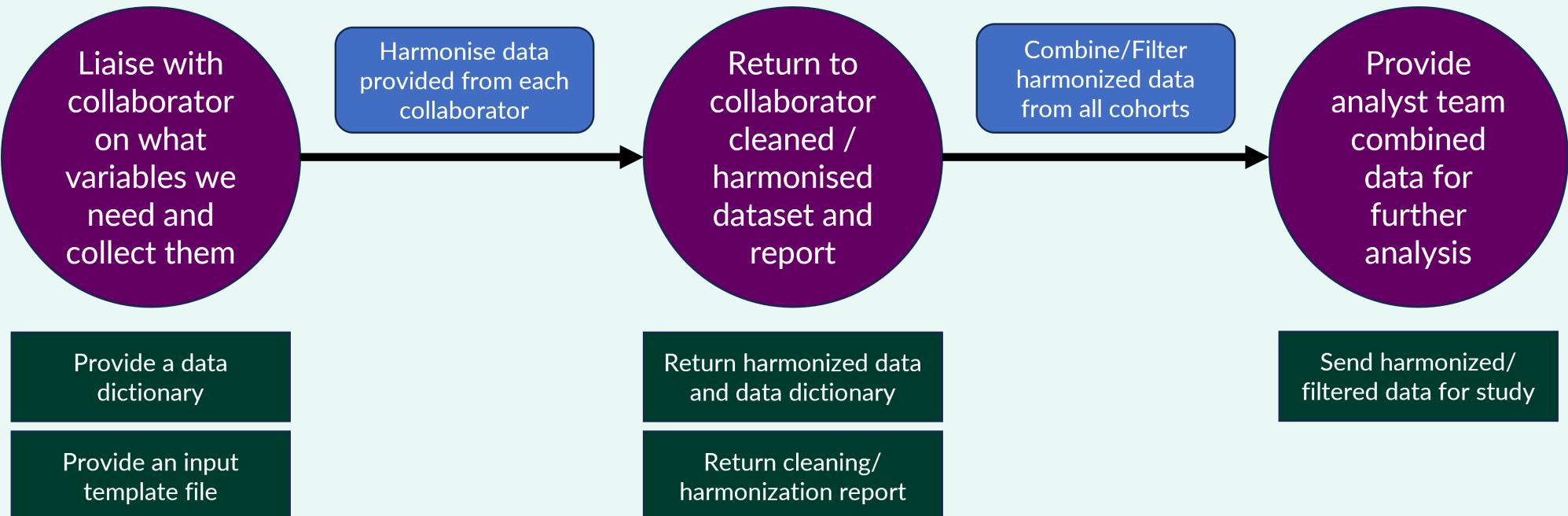
Figure 1. The data harmonization process.

Study data variables collected from different sources need to be mapped to one another (step 1), classified into the generalized concepts they represent (step 2), and transformed into unified harmonized variables (step 3) for analysis.

Image from Mallya et al. Circ Cardiovasc Qual Outcomes. 2023 Nov; 16(11):e009938 doi: 10.1161/CIRCOUTCOMES.123.009938.

Tasks

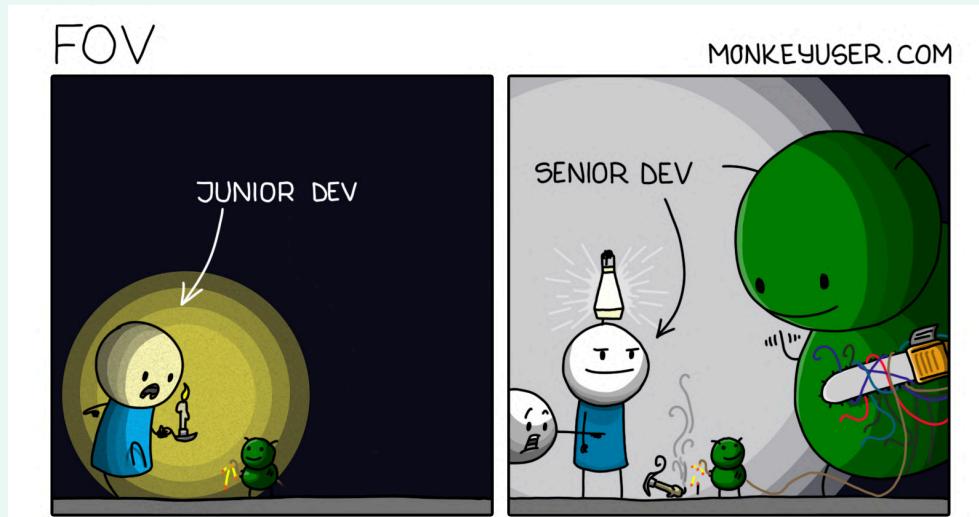
Here is a summary of my task for an international study.



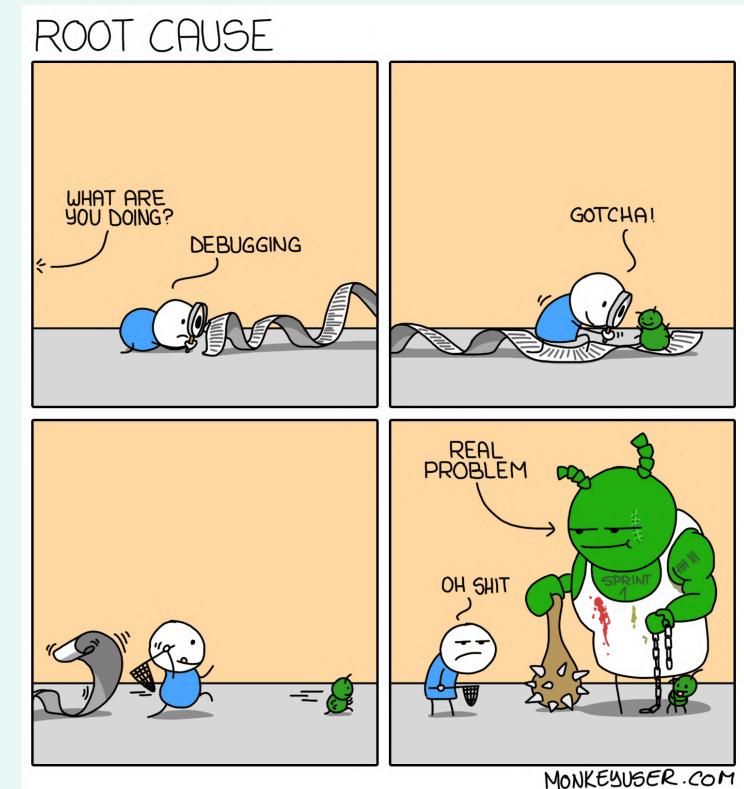
Why Retrospective Data Harmonisation ?

Collaborators may not have resources (time/funding) to match their data to the data dictionary and input template provided.

Data provided may have issues that needs relevant experience and may be hard to solve.



Fov from MonkeyUser.com



Root Cause from MonkeyUser.com

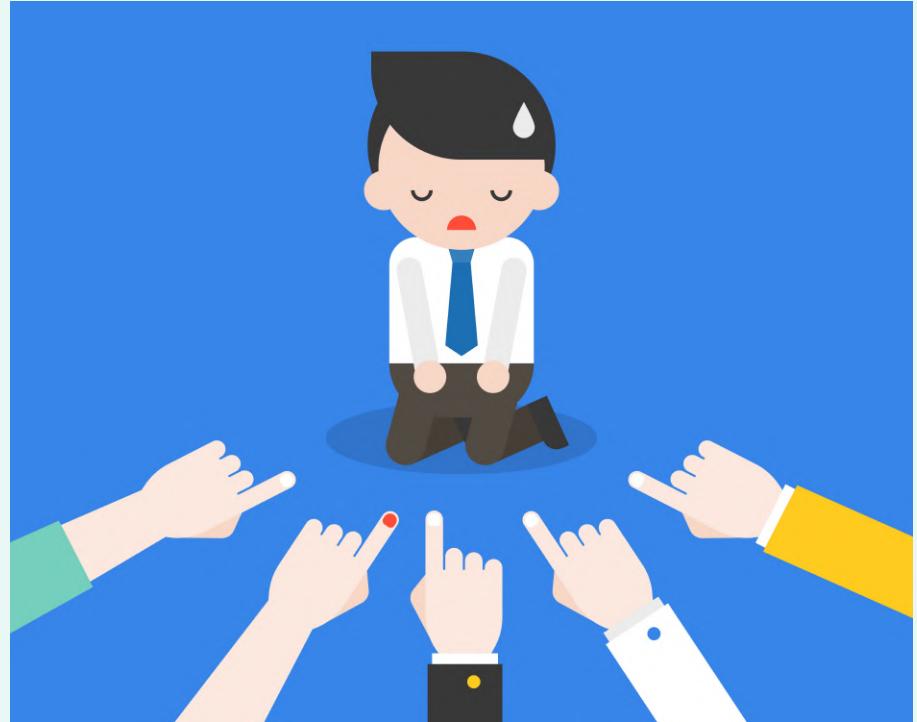
Why Retrospective Data Harmonisation ?

Collaborator wants transparency on how the data is processed.

High risk of getting blame should things don't go well.



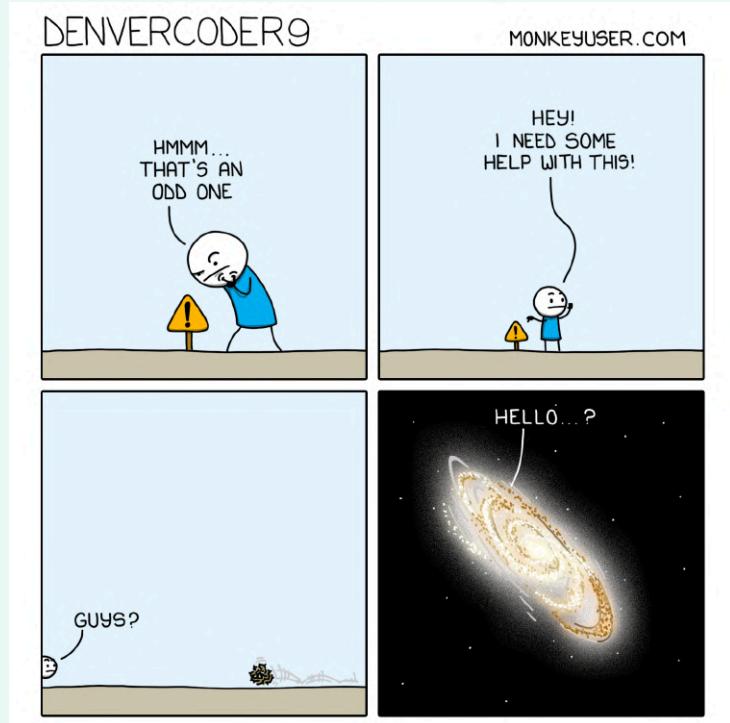
Joy from MonkeyUser.com



Businessman kneel on floor with pointing finger to him by Amonrat Rungreangfangsai

Issues

Limited resources on how to do data harmonisation and to make a report.



DenverCoder9 from [MonkeyUser.com](#)

R packages related to data harmonisation

- `retroharmonize` for survey data.
- `Rmonize` for epidemiological data.
- `psHarmonize` for health and education data.

Issues

retroharmonize works with categorical but not continuous data variables.

Rmonize documentation has too much jargon and difficult for me (and people with limited data science background) to understand.

psHarmonize variable harmonisation process is reported in a Microsoft Excel file. Works for variables with simple harmonisation process.

A Harmonization instructions for height																		
1	study	visit	item	source_dataset	source_item	code1	code_type	coding_notes		id_var	possible_ran_domain	subdomain	final_units	source_units	label	notes		source_labels
2	AGES	1	Height	age733_drrp_first_visit_additional	HEIGHT			No change needed		ID	Clinical Risk F Height	cm	n/a					
3	ARIC	1	Height	VISIT_1	ANTAO1			No change needed		ID	Clinical Risk F Height	cm	cm					STANDING HEIGHT TO NEAR
4	ARIC	3	Height	VISIT_3	ANTC1			No change needed		ID	Clinical Risk F Height	cm	cm					STANDING HEIGHT TO NEAR
5	ARIC	4	Height	VISIT_4	ANTD1			No change needed		ID	Clinical Risk F Height	cm	cm					STANDING HEIGHT TO NEAR
6	ARIC	5	Height	VISIT_5	ANT3			No change needed		ID	Clinical Risk F Height	cm	cm					Standing height (cm)
7	ARIC	6	Height	VISIT_6	ANT3			No change needed		ID	Clinical Risk F Height	cm	cm					Standing Height (cm)
8	ARIC	7	Height	VISIT_7	ANT3			No change needed		ID	Clinical Risk F Height	cm	cm					Standing Height (cm)
9	CHS	2	Height	CHS_main	stht_y2			Imputed 2 height values on visit 18 per CHS instruct idno			Clinical Risk F Height	cm	cm					STANDING HEIGHT - CM
10	CHS	5	Height	CHS_main	stht_y5			Imputed 2 height values on visit 18 per CHS instruct idno			Clinical Risk F Height	cm	cm					STANDING HEIGHT - CM
11	CHS	9	Height	CHS_main	stht_y9			Imputed 2 height values on visit 18 per CHS instruct idno			Clinical Risk F Height	cm	cm					STANDING HEIGHT - CM
12	CHS	18	Height	chs_height_v18	stht_y18			Imputed 2 height values on visit 18 per CHS instruct idno			Clinical Risk F Height	cm	cm					STANDING HEIGHT - CM
13	FHS - Cohort	1	Height	vr_wkthru_ex32_0_0997s_19	HGT1	x * 2.54	function	Converted from inches to cm.		shareid	Clinical Risk F Height	cm	in					Height, Exam 1
14	FHS - Cohort	4	Height	vr_wkthru_ex32_0_0997s_19	HGT4	x * 2.54	function	Converted from inches to cm.		shareid	Clinical Risk F Height	cm	in					Height, Exam 4
15	FHS - Cohort	5	Height	vr_wkthru_ex32_0_0997s_19	HGT5	x * 2.54	function	Converted from inches to cm.		shareid	Clinical Risk F Height	cm	in					Height, Exam 5
16	FHS - Cohort	10	Height	vr_wkthru_ex32_0_0997s_19	HGT10	x * 2.54	function	Converted from inches to cm.		shareid	Clinical Risk F Height	cm	in					Height, Exam 10
17	FHS - Cohort	11	Height	vr_wkthru_ex32_0_0997s_19	HGT11	x * 2.54	function	Converted from inches to cm.		shareid	Clinical Risk F Height	cm	in					Height, Exam 11
18	FHS - Cohort	12	Height	vr_wkthru_ex32_0_0997s_19	HGT12	x * 2.54	function	Converted from inches to cm.		shareid	Clinical Risk F Height	cm	in					Height, Exam 12
19	FHS - Cohort	13	Height	vr_wkthru_ex32_0_0997s_19	HGT13	x * 2.54	function	Converted from inches to cm.		shareid	Clinical Risk F Height	cm	in					Height, Exam 13

Image from Stephen et al. Patterns (N Y). 2024 Jun; 5(8):101003 doi: [10.1016/j.patter.2024.101003](https://doi.org/10.1016/j.patter.2024.101003).

Harmonisation Project Template

<https://jauntyjjs.github.io/harmonisation/>

harmonisation 1.0.0.0 Reference

Search for



Data Harmonisation Project Template

Table of Content

- [Motivation](#)
- [Acknowledgement](#)
- [File Structure](#)
- [Software Installation](#)
- [R Package Installation](#)
- [Using .renv](#)
- [R Functions Management](#)
- [R Packages I Used](#)

Links

[Browse source code](#)

[Report a bug](#)

License

[Full license](#)

[MIT + file LICENSE](#)

Citation

[Citing harmonisation](#)

Developers

Jeremy Selva

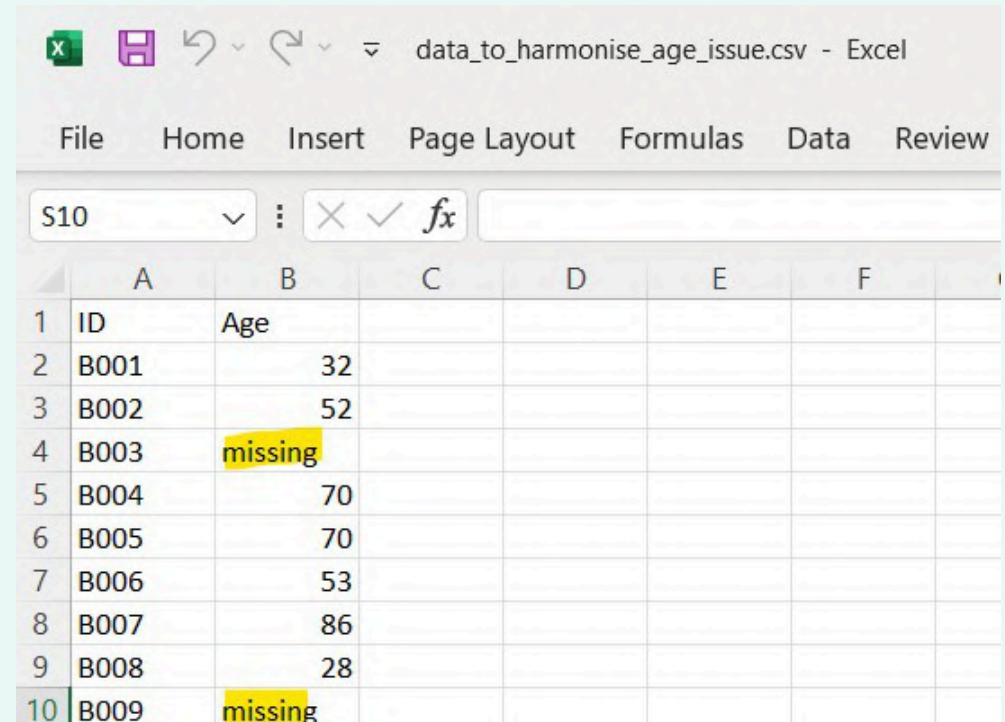
Author, maintainer An ORCID iD icon, which is a green circular badge with the letters "id" in white.

Automated capture of warnings (csv)

When reading files in batches or the same file but in a different version, is there an automated way to catch warnings/issues when reading csv files ?

```
1 cohort_data_csv <- vroom::vroom(  
2   file = here::here("data-raw", "Cohort_csv",  
3     "data_to_harmonise_age_issue.csv"),  
4   delim = ",",  
5   col_select = 1:2,  
6   show_col_types = FALSE,  
7   col_types = list(  
8     ID = vroom::col_character(),  
9     Age = vroom::col_integer()  
10    )  
11  )  
12  
13 head(cohort_data_csv, n = 3)
```

```
# A tibble: 3 × 2  
Warning: One or more parsing issues, call `problems()` on your data frame for  
details,  
e.g.:  
dat <- vroom(...)  
problems(dat)  
  
ID      Age  
<chr> <int>  
1 B001      32  
2 B002      52  
3 B003      NA
```



S10	A	B	C	D	E	F
1	ID	Age				
2	B001		32			
3	B002		52			
4	B003	missing				
5	B004		70			
6	B005		70			
7	B006		53			
8	B007		86			
9	B008		28			
10	B009	missing				

Automated capture of warnings (csv)

If there are issues with the data, the output of `vroom::problems` will be a tibble.

```
1 cohort_data_csv |>
2   vroom::problems()

# A tibble: 4 × 5
  row    col expected    actual   file
  <int> <int> <chr>       <chr>   <chr>
1     2      2 an integer missing D:/Jeremy/PortableR/RPortableWorkDirectory/RMe...
2     4      2 an integer missing D:/Jeremy/PortableR/RPortableWorkDirectory/RMe...
3    10      2 an integer missing D:/Jeremy/PortableR/RPortableWorkDirectory/RMe...
4    17      2 an integer missing D:/Jeremy/PortableR/RPortableWorkDirectory/RMe...
```

To check for this automatically, we can use `pointblank::expect_row_count_match`.

```
1 cohort_data_csv |>
2   vroom::problems() |>
3   pointblank::expect_row_count_match(count = 0)

Error: Row counts for the two tables did not match.
The `expect_row_count_match()` validation failed beyond the absolute threshold level (1).
* failure level (1) >= failure threshold (1)
```

Automated capture of warnings (csv)

Here is a case with no issues.

```
1 cohort_data_csv <- vroom::vroom(  
2   file = here::here("data-raw", "Cohort_csv",  
3     "data_to_harmonise_age_issue_fixed.csv"),  
4   delim = ",",  
5   col_select = 1:2,  
6   show_col_types = FALSE,  
7   col_types = list(  
8     ID = vroom::col_character(),  
9     Age = vroom::col_integer()  
10    )  
11  )  
12  
13 cohort_data_csv |>  
14   vroom::problems()
```

```
# A tibble: 0 × 5  
# i 5 variables: row <int>, col <int>, expected <chr>, actual <chr>, file <chr>
```

```
1 cohort_data_csv |>  
2   vroom::problems() |>  
3   pointblank::expect_row_count_match(count = 0)
```

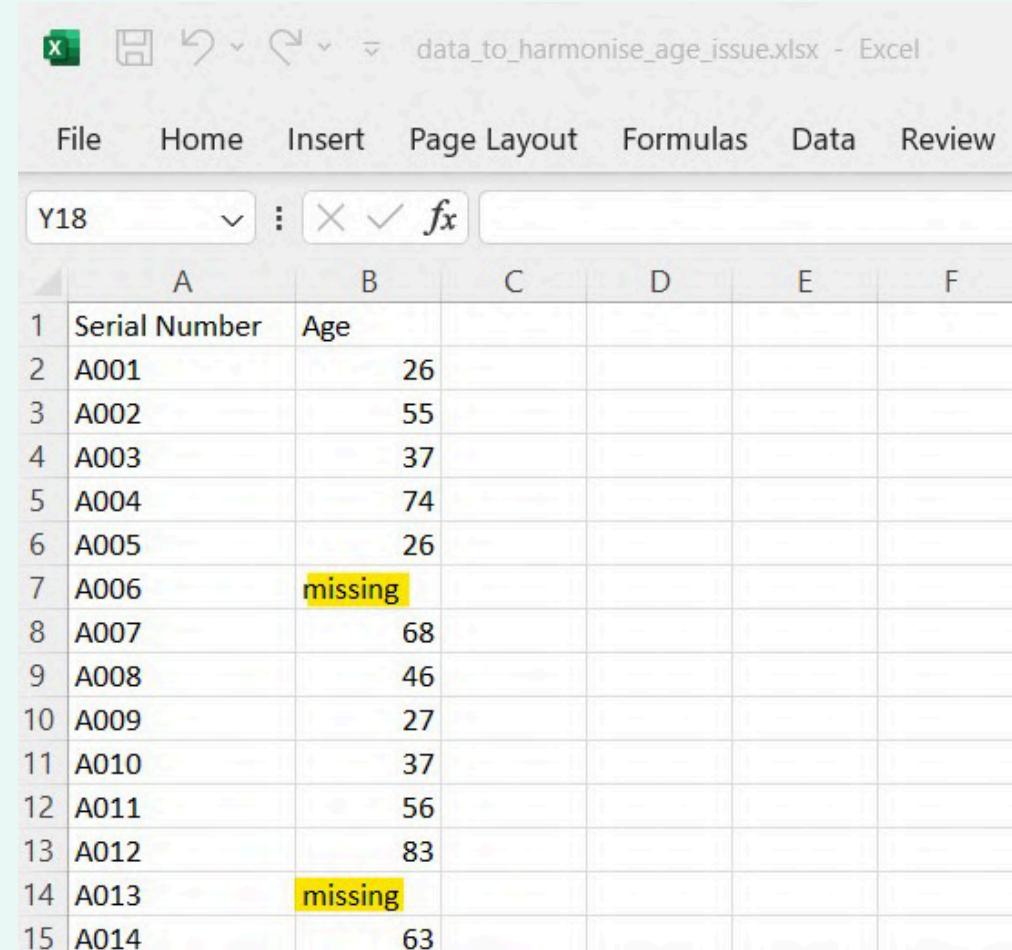
	A	B	C	D	E	F	G
1	ID	Age					
2	B001	32					
3	B002	52					
4	B003	80					
5	B004	70					
6	B005	70					
7	B006	53					
8	B007	86					
9	B008	28					
10	B009	60					

Automated capture of warnings (Excel)

Is there an automated way to catch warnings/issues when reading Excel files ?

```
1 cohort_data_excel <- readxl::read_excel(  
2   path = here::here("data-raw", "Cohort_Excel",  
3     "data_to_harmonise_age_issue.xlsx"),  
4   sheet = "Sheet1",  
5   col_types = c(  
6     "text", "numeric"  
7   )  
8 )
```

```
Warning: Expecting numeric in B7 / R7C2: got 'missing'  
Warning: Expecting numeric in B14 / R14C2: got 'missing'
```



The screenshot shows a Microsoft Excel spreadsheet titled "data_to_harmonise_age_issue.xlsx". The data is organized into two columns: "Serial Number" (Column A) and "Age" (Column B). The rows are numbered from 1 to 15. Cells B7 and B14 contain the text "missing", which is highlighted with a yellow background. The rest of the data is numerical.

	A	B	C	D	E	F
1	Serial Number	Age				
2	A001	26				
3	A002	55				
4	A003	37				
5	A004	74				
6	A005	26				
7	A006	missing				
8	A007	68				
9	A008	46				
10	A009	27				
11	A010	37				
12	A011	56				
13	A012	83				
14	A013	missing				
15	A014	63				

Automated capture of warnings (Excel)

We can read the Excel file with `testthat::expect_no_condition`.

```
1 testthat::expect_no_condition()
2 cohort_data_excel <- readxl::read_excel(
3   path = here::here("data-raw", "Cohort_Excel",
4     "data_to_harmonise_age_issue.xlsx"),
5   sheet = "Sheet1",
6   col_types = c("text", "numeric")
7 )
8 )
```

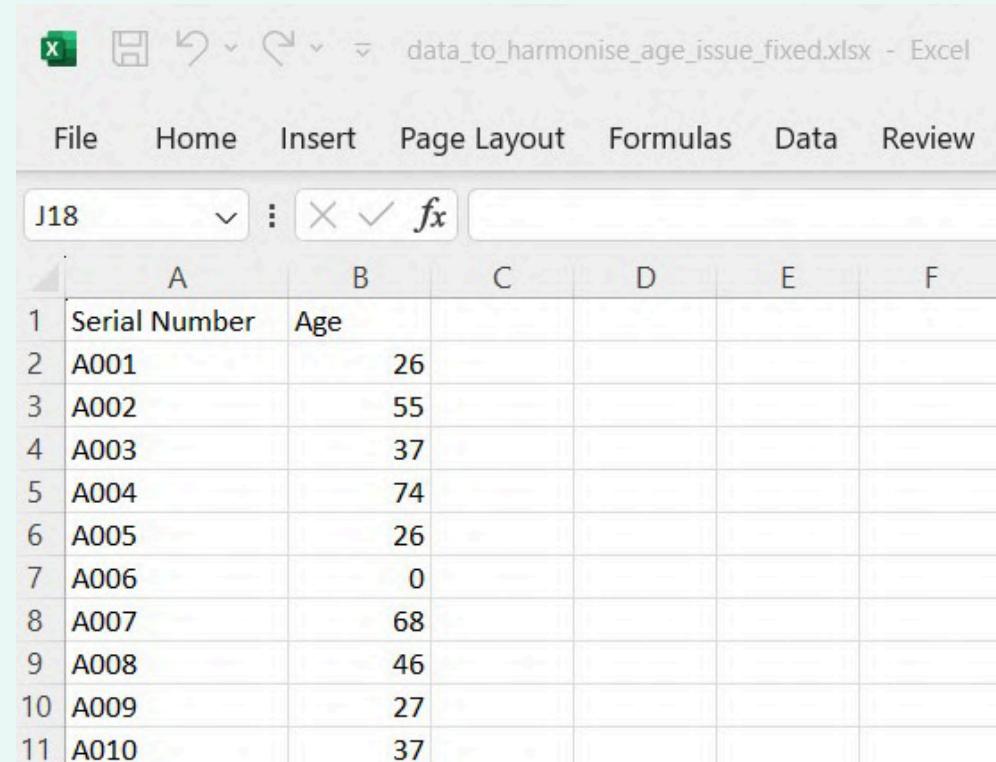
```
Error: Expected `... <- NULL` to run without any conditions.
i Actually got a <simpleWarning> with text:
  Expecting numeric in B7 / R7C2: got 'missing'
```

	A	B	C	D	E	F
1	Serial Number	Age				
2	A001	26				
3	A002	55				
4	A003	37				
5	A004	74				
6	A005	26				
7	A006	missing				
8	A007	68				
9	A008	46				
10	A009	27				
11	A010	37				
12	A011	56				
13	A012	83				
14	A013	missing				
15	A014	63				

Automated capture of warnings (Excel)

However, this method means that you will lose the pipe workflow.

```
1 testthat::expect_no_condition()
2 cohort_data_excel <- readxl::read_excel(
3   path = here::here("data-raw", "Cohort_Excel",
4     "data_to_harmonise_age_issue_fixed.xlsx"),
5   sheet = "Sheet1",
6   col_types = c("text", "numeric"))
7 )
8 )
9
10 cohort_data_excel <- cohort_data_excel |>
11   # Check if Serial Number is unique
12   pointblank::rows_distinct(
13     columns = "Serial Number",
14   )
```



	A	B	C	D	E	F
1	Serial Number	Age				
2	A001	26				
3	A002	55				
4	A003	37				
5	A004	74				
6	A005	26				
7	A006	0				
8	A007	68				
9	A008	46				
10	A009	27				
11	A010	37				

Automated capture of warnings (Excel)

We can use the tee pipe operator `%T>%`.

With Issues

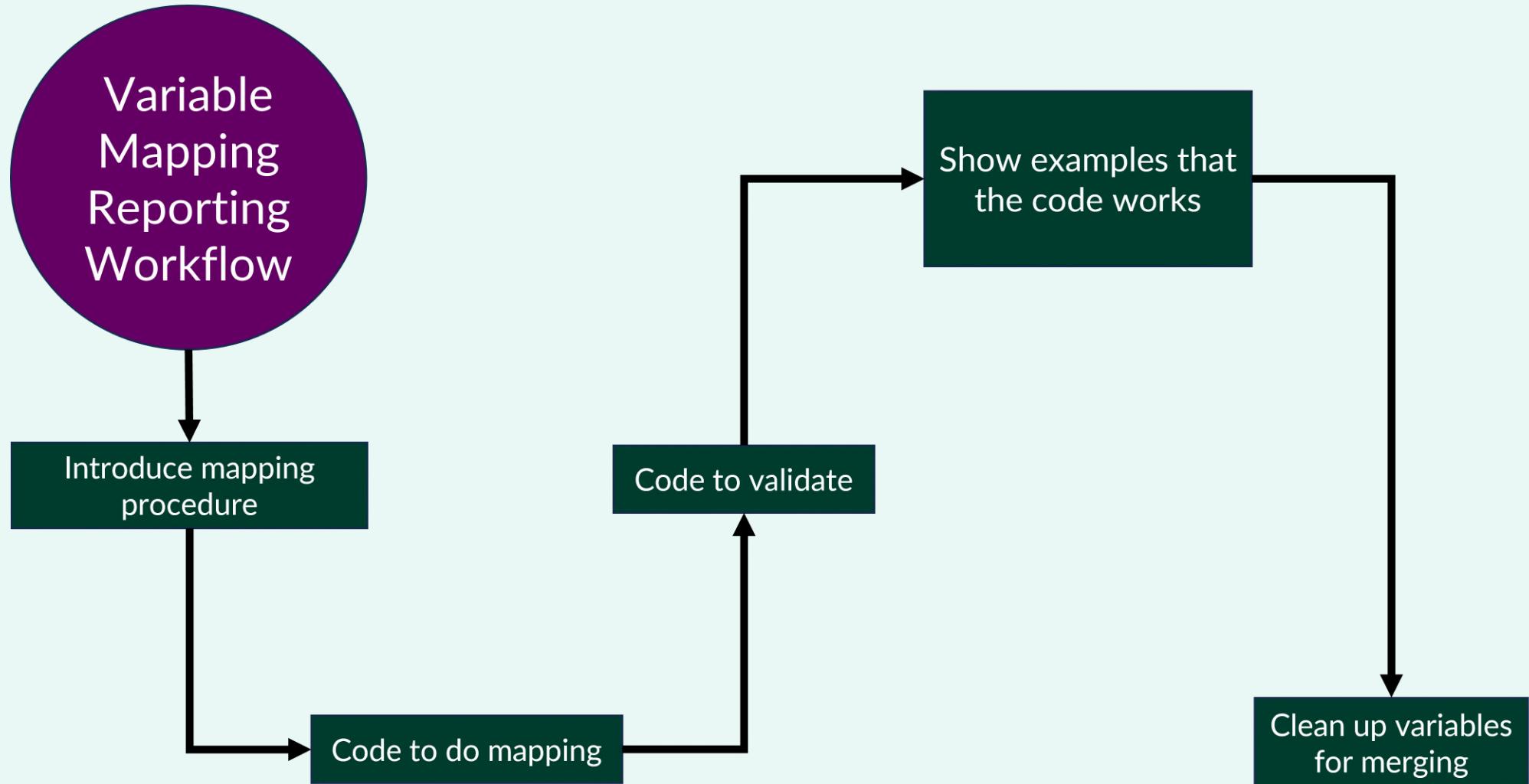
```
1 cohort_data_excel <- readxl::read_excel(  
2   path = here::here("data-raw", "Cohort_Excel",  
3     "data_to_harmonise_age_issue.xlsx"),  
4   sheet = "Sheet1",  
5   col_types = c(  
6     "text", "numeric"  
7   )  
8 ) %T>%  
9 testthat::expect_no_condition()
```

```
Error: Expected `.` to run without any conditions.  
i Actually got a <simpleWarning> with text:  
  Expecting numeric in B7 / R7C2: got 'missing'
```

No Issues

```
1 cohort_data_excel_2 <- readxl::read_excel(  
2   path = here::here("data-raw", "Cohort_Excel",  
3     "data_to_harmonise_age_issue_fixed.xlsx"),  
4   sheet = "Sheet1",  
5   col_types = c("text", "numeric")  
6 ) %T>%  
7 testthat::expect_no_condition() |>  
8 # Check if Serial Number is unique  
9 pointblank::rows_distinct(  
10   columns = "Serial Number",  
11 )
```

Variable Mapping



Variable Mapping

Let take this data set as an example.

```
1 cohort_csv_data <- vroom::vroom(  
2   file = here::here("data-raw",  
3     "Cohort_csv",  
4     "data_to_harmonise.csv"),  
5   delim = ",",  
6   col_select = 1:8,  
7   show_col_types = FALSE,  
8   col_types = list(  
9     ID = vroom::col_character(),  
10    Age = vroom::col_integer(),  
11    Sex = vroom::col_character(),  
12    Height = vroom::col_double(),  
13    Weight = vroom::col_double(),  
14    `Smoke History` = vroom::col_character(),  
15    `Chest Pain Character` = vroom::col_character(),  
16    Dyspnea = vroom::col_character()  
17  )  
18 ) |>  
19 dplyr::rename(cohort_unique_id = "ID") |>  
20 # Remove rows when the ID value is NA  
21 dplyr::filter(!is.na(.data[["cohort_unique_id"]])) |>  
22 # Remove white spaces in column names  
23 dplyr::rename_all(stringr::str_trim) |>  
24 # Check if cohort id is unique  
25 pointblank::rows_distinct(  
  columns = "cohort_unique_id".
```

cohort_uniq ue_id	Age	Sex	Height	Weight	Si H
B001	32	Female	170	63	ne H
B002	52	Female	167	71	cu sr
B003	80	Male	184	77	ne sr
B004	70	Male	160	83	pa sr
B005	70	Female	155	61	cu sr

1-5 of 20 rows

Previous 1 of 4 Next

[Download as CSV](#)

Variable Mapping

Let the reader know how the collaborator's data **Smoke History** is going to be mapped.

Introduce mapping procedure

```
### Smoking History
`smoke_current` is grouped as follows:

```{r}
#| label: smoke current table
#| echo: false
#| message: false
#| warnings: false
#| results: asis

tbl1 <- "
+-----+
| Smoke History | smoke_current |
+=====+=====
| non-smoker | 0 |
+-----+
| past smoker | 0 |
+-----+
| current smoker| 1 |
+-----+
| NA | -1 |
+-----+
"
cat(tbl1)
```

```

2.3 Smoking History

`smoke_current` is grouped as follows:

| Smoke History | smoke_current |
|----------------|---------------|
| non-smoker | 0 |
| past smoker | 0 |
| current smoker | 1 |
| NA | -1 |

`smoke_past` is grouped as follows:

| Smoke History | smoke_past |
|----------------|------------|
| non-smoker | 0 |
| past smoker | 1 |
| current smoker | 0 |
| NA | -1 |

We do a check to ensure that we can only have these scenarios

- `smoke_current` as 1 and `smoke_past` as 0 for current smokers
- `smoke_current` as 0 and `smoke_past` as 1 for past smokers
- `smoke_current` as 0 and `smoke_past` as 0 for non-smokers
- `smoke_current` as -1 and `smoke_past` as -1 for unknown

Variable Mapping

Code to do mapping

```
1 smoking_data <- cohort_csv_data |>
2   dplyr::select(c("cohort_unique_id",
3                 "Smoke History")) |>
4   dplyr::mutate(
5     smoke_current = dplyr::case_when(
6       is.na(.data[["Smoke History"]]) ~ "-1",
7       .data[["Smoke History"]] == "non-smoker" ~ "0",
8       .data[["Smoke History"]] == "past smoker" ~ "0",
9       .data[["Smoke History"]] == "current smoker" ~ "1",
10      .default = NA_character_
11    ),
12    smoke_current = forcats::fct_relevel(
13      .data[["smoke_current"]],
14      c("0", "1")),
15    smoke_past = dplyr::case_when(
16      is.na(.data[["Smoke History"]]) ~ "-1",
17      .data[["Smoke History"]] == "non-smoker" ~ "0",
18      .data[["Smoke History"]] == "past smoker" ~ "1",
19      .data[["Smoke History"]] == "current smoker" ~ "0",
20      .default = NA_character_
21    ),
22    smoke_past = forcats::fct_relevel(
23      .data[["smoke_past"]],
24      c("0", "1")),
25    `Smoke History` = forcats::fct(
26      .data[["Smoke History"]]
```

2.3 Smoking History

`smoke_current` is grouped as follows:

| Smoke History | smoke_current |
|----------------|---------------|
| non-smoker | 0 |
| past smoker | 0 |
| current smoker | 1 |
| NA | -1 |

`smoke_past` is grouped as follows:

| Smoke History | smoke_past |
|----------------|------------|
| non-smoker | 0 |
| past smoker | 1 |
| current smoker | 0 |
| NA | -1 |

Variable Mapping

Code to validate

```
1 smoking_data <- smoking_data |>
2   pointblank::col_vals_in_set(
3     columns = c("smoke_current", "smoke_past"),
4     set = c("0", "1", "-1")
5   ) |>
6   pointblank::col_vals_expr(
7     expr = pointblank::expr(
8       (.data[["smoke_current"]] == "1" & .data[["smoke_past"]] == "0") |
9       (.data[["smoke_current"]] == "-1" & .data[["smoke_past"]] == "-1") |
10      (.data[["smoke_current"]] == "0" & .data[["smoke_past"]] %in% c("0",
11        )
12    )
```

We do a check to ensure that we can only have these scenarios

- `smoke_current` as 1 and `smoke_past` as 0 for current smokers
- `smoke_current` as 0 and `smoke_past` as 1 for past smokers
- `smoke_current` as 0 and `smoke_past` as 0 for non-smokers
- `smoke_current` as -1 and `smoke_past` as -1 for unknown

Reference: <https://github.com/rstudio/pointblank/issues/578>

Variable Mapping

Show examples that the code works

```
```{.content-visible when-format="html"}  
```{r}  
#| label: smoking_data_html  
#| eval: !expr out_type == "html"  
  
if (params$show_table && knitr::is_html_output()) {  
  smoking_data |>  
  harmonisation::reactable_with_download_csv_button()  
}  
```  
```
```

Html Output

cohort_unique_id	Smoke History	smoke_current	smoke_past
	All	All	All
B001	non-smoker	0	0
B002	current smoker	1	0
B003	non-smoker	0	0
B004	past smoker	0	1
B005	current smoker	1	0

1-5 of 20 rows

Previous

1 of 4 Next

 Download as CSV

Variable Mapping

Show examples that the code works

```
### {.content-visible unless-format="html"}  
```{r}  
#| label: smoking data not html
#| eval: !expr out_type != "html"

if (params$show_table) {
 smoking_data |>
 dplyr::distinct(.data[["Smoke History"]],
 .keep_all = TRUE) |>
 knitr::kable()
}
...
```
```

Pdf Output

```
if (params$show_table) {  
  smoking_data |>  
    dplyr::distinct(.data[["Smoke History"]],  
    .keep_all = TRUE) |>  
  knitr::kable()  
}
```

| cohort_unique_id | Smoke History | smoke_current | smoke_past |
|------------------|----------------|---------------|------------|
| B001 | non-smoker | 0 | 0 |
| B002 | current smoker | 1 | 0 |
| B004 | past smoker | 0 | 1 |
| B017 | NA | -1 | -1 |

Variable Mapping

Clean up variables
for merging

```
1 smoking_data <- smoking_data |>  
2 dplyr::select(-c("Smoke History"))
```

| cohort_unique_id | smoke_current | smoke_past |
|------------------|---------------|------------|
| | All | All |
| B001 | 0 | 0 |
| B002 | 1 | 0 |
| B003 | 0 | 0 |
| B004 | 0 | 1 |
| B005 | 1 | 0 |

1-5 of 20 rows

Previous 1 of 4 Next

 Download as CSV

Merging Harmonised Data

Suppose we have completed harmonising a batch of clinical data.

```
1 age_gender_data |>
2   reactable_with_download_csv_button(
3     defaultPageSize = 5,
4     paginationType = "jump",
5     style = list(fontSize = "1rem"),
6   )
```

| cohort_unique_id | age_years | sex |
|------------------|-----------|-----|
| B001 | 32 | 0 |
| B002 | 52 | 0 |
| B003 | 80 | 1 |
| B004 | 70 | 1 |
| B005 | 70 | 0 |

1-5 of 20 rows Previous 1 of 4 Next

 Download as CSV

```
1 body_measurement_data |>
2   reactable_with_download_csv_button(
3     defaultPageSize = 5,
4     paginationType = "jump",
5     style = list(fontSize = "1rem"),
6   )
```

| cohort_unique_id | height_cm | weight_kg | bsa_m2 | bmi |
|------------------|-----------|-----------|--------|-------|
| B001 | 170 | 63 | 1.72 | 21.8 |
| B002 | 167 | 71 | 1.81 | 25.46 |
| B003 | 184 | 77 | 1.98 | 22.74 |
| B004 | 160 | 83 | 1.92 | 32.42 |
| B005 | 155 | 61 | 1.62 | 25.39 |

1-5 of 20 rows Previous 1 of 4 Next

 Download as CSV

How can we merge them without issues of missing rows or additional columns ?

Merging Harmonised Data

`unmatched = "error"` in `dplyr::inner_join` helps to avoid patients with no match.

```
1 join_specification <- dplyr::join_by("cohort_unique_id")
2
3 demo_behavior_data <- cohort_csv_data |>
4   dplyr::select(c("cohort_unique_id")) |>
5   dplyr::inner_join(age_gender_data,
6     by = join_specification,
7     unmatched = "error",
8     relationship = "one-to-one") |>
9   dplyr::inner_join(body_measurement_data,
10    by = join_specification,
11    unmatched = "error",
12    relationship = "one-to-one") |>
13   dplyr::inner_join(smoking_data,
14     by = join_specification,
15     unmatched = "error",
16     relationship = "one-to-one") |>
17   dplyr::relocate(c("bsa_m2", "bmi"),
18     .after = "sex")
```

```
1 three_penguins <- tibble::tribble(
2   ~samp_id, ~species, ~island,
3   1, "Adelie", "Torgersen",
4   2, "Gentoo", "Biscoe",
5   )
6
7 weight_extra <- tibble::tribble(
8   ~samp_id, ~body_mass_g,
9   1, 3220,
10  2, 4730,
11  4, 4725
12  )
13
14 three_penguins |>
15   dplyr::inner_join(
16     y = weight_extra,
17     by = dplyr::join_by("samp_id"),
18     unmatched = "error"
19   )
```

```
Error in `dplyr::inner_join()`:
! Each row of `y` must be matched by `x`.
  i Row 3 of `y` was not matched.
```

Reference: https://www.tidyverse.org/blog/2023/08/teach-tidyverse-23/#improved-and-expanded-_join-functionality

Merging Harmonised Data

`unmatched = "error"` in `dplyr::inner_join` helps to avoid patients with no match.

```
1 join_specification <- dplyr::join_by("cohort_unique_id")
2
3 demo_behavior_data <- cohort_csv_data |>
4   dplyr::select(c("cohort_unique_id")) |>
5   dplyr::inner_join(age_gender_data,
6     by = join_specification,
7     unmatched = "error",
8     relationship = "one-to-one") |>
9   dplyr::inner_join(body_measurement_data,
10    by = join_specification,
11    unmatched = "error",
12    relationship = "one-to-one") |>
13   dplyr::inner_join(smoking_data,
14     by = join_specification,
15     unmatched = "error",
16     relationship = "one-to-one") |>
17   dplyr::relocate(c("bsa_m2", "bmi"),
18     .after = "sex")
```

```
1 three_penguins <- tibble::tribble(
2   ~samp_id, ~species, ~island,
3   1, "Adelie", "Torgersen",
4   2, "Gentoo", "Biscoe",
5   3, "Chinstrap", "Dream"
6 )
7
8 weight_extra <- tibble::tribble(
9   ~samp_id, ~body_mass_g,
10  1, 3220,
11  3, 4725
12 )
13
14 three_penguins |>
15   dplyr::inner_join(
16     y = weight_extra,
17     by = dplyr::join_by("samp_id"),
18     unmatched = "error"
19 )
```

```
Error in `dplyr::inner_join()`:
! Each row of `x` must have a match in `y`.
i Row 2 of `x` does not have a match.
```

Reference: https://www.tidyverse.org/blog/2023/08/teach-tidyverse-23/#improved-and-expanded-_join-functionality

Merging Harmonised Data

`relationship = "one-to-one"` in `dplyr::inner_join` helps to avoid patients with multiple match.

```
1 join_specification <- dplyr::join_by("cohort_unique_id")
2
3 demo_behave_data <- cohort_csv_data |>
4   dplyr::select(c("cohort_unique_id")) |>
5   dplyr::inner_join(age_gender_data,
6     by = join_specification,
7     unmatched = "error",
8     relationship = "one-to-one") |>
9   dplyr::inner_join(body_measurement_data,
10    by = join_specification,
11    unmatched = "error",
12    relationship = "one-to-one") |>
13  dplyr::inner_join(smoking_data,
14    by = join_specification,
15    unmatched = "error",
16    relationship = "one-to-one") |>
17  dplyr::relocate(c("bsa_m2", "bmi"),
18    .after = "sex")
```

```
1 three_penguins <- tibble::tribble(
2   ~samp_id, ~species, ~island,
3   1, "Adelie", "Torgersen",
4   2, "Gentoo", "Biscoe",
5   3, "Chinstrap", "Dream"
6 )
7
8 weight_extra <- tibble::tribble(
9   ~samp_id, ~body_mass_g,
10  1, 3220,
11  2, 4730,
12  2, 4725,
13  3, 4000
14 )
15
16 three_penguins |>
17   dplyr::inner_join(
18     y = weight_extra,
19     by = dplyr::join_by("samp_id"),
20     relationship = "one-to-one"
21 )
```

```
Error in `dplyr::inner_join()`:
! Each row in `x` must match at most 1 row in `y`.
i Row 2 of `x` matches multiple rows in `y`.
```

Reference: https://www.tidyverse.org/blog/2023/08/teach-tidyverse-23/#improved-and-expanded-_join-functionality

Merging Harmonised Data

Use `pointblank::has_columns` to ensure we only have harmonised variables.

```
1 testthat::expect_false(
2   pointblank::has_columns(
3     demo_behave_data,
4     columns = c(
5       dplyr::ends_with(".x"),
6       dplyr::ends_with(".y")
7     )
8   )
9 )
10
11 testthat::expect_equal(
12   ncol(demo_behave_data), 9
13 )
14
15 testthat::expect_true(
16   pointblank::has_columns(
17     demo_behave_data,
18     columns = c(
19       "age_years", "sex",
20       "height_cm", "weight_kg", "bsa_m2", "bmi",
21       "smoke_current", "smoke_past"
22     )
23   )
24 )
```

```
1 three_penguins <- tibble::tribble(
2   ~samp_id, ~species, ~island,
3   1, "Adelie", "Torgersen",
4   2, "Gentoo", "Biscoe",
5   3, "Chinstrap", "Dream"
6 )
7
8 weight_extra <- tibble::tribble(
9   ~samp_id, ~island,
10  1, "Torgersen",
11  2, "Biscoe",
12  3, "Dream"
13 )
14
15 three_penguins <- three_penguins |>
16   dplyr::inner_join(
17     y = weight_extra,
18     by = dplyr::join_by("samp_id"),
19     unmatched = "error",
20     relationship = "one-to-one"
21 )
22
23 three_penguins |>
24   pointblank::has_columns(
25     columns = c(
26       dplyr::ends_with(".x")).
```

```
[1] TRUE
```

```
1 colnames(three_penguins)
```

```
[1] "samp_id"  "species"  "island.x" "island.y"
```

Automated Report Challenge

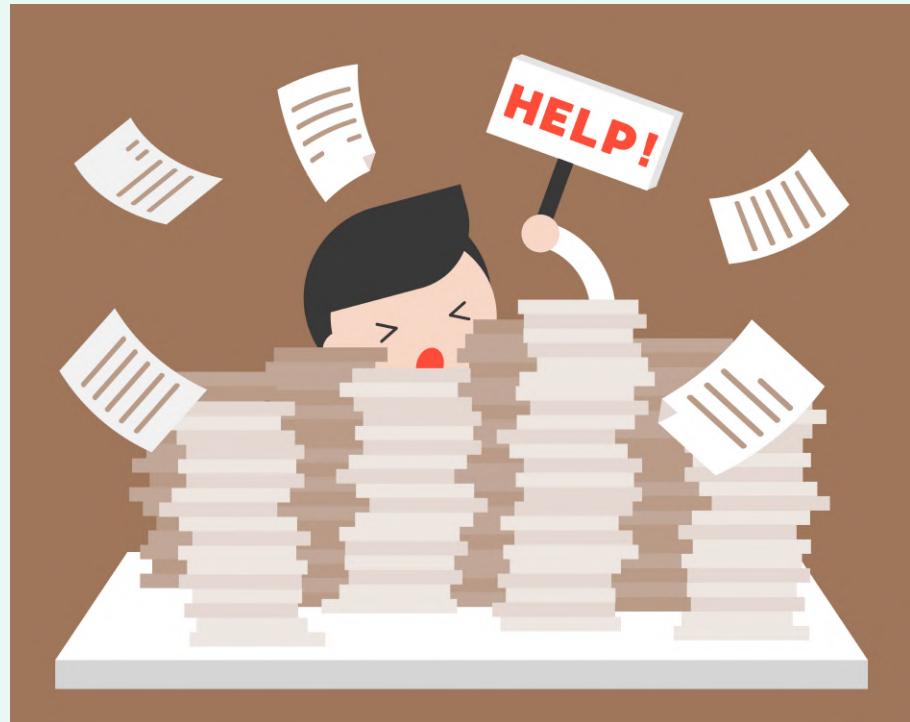
One variable mapping report takes at least one page.

On average, a clinical trial will have a few hundred variables.

- One hundred columns for clinical and demographics.
- Two hundred columns for medication.

Harmonisation report can have at least a few hundreds pages for each cohort.

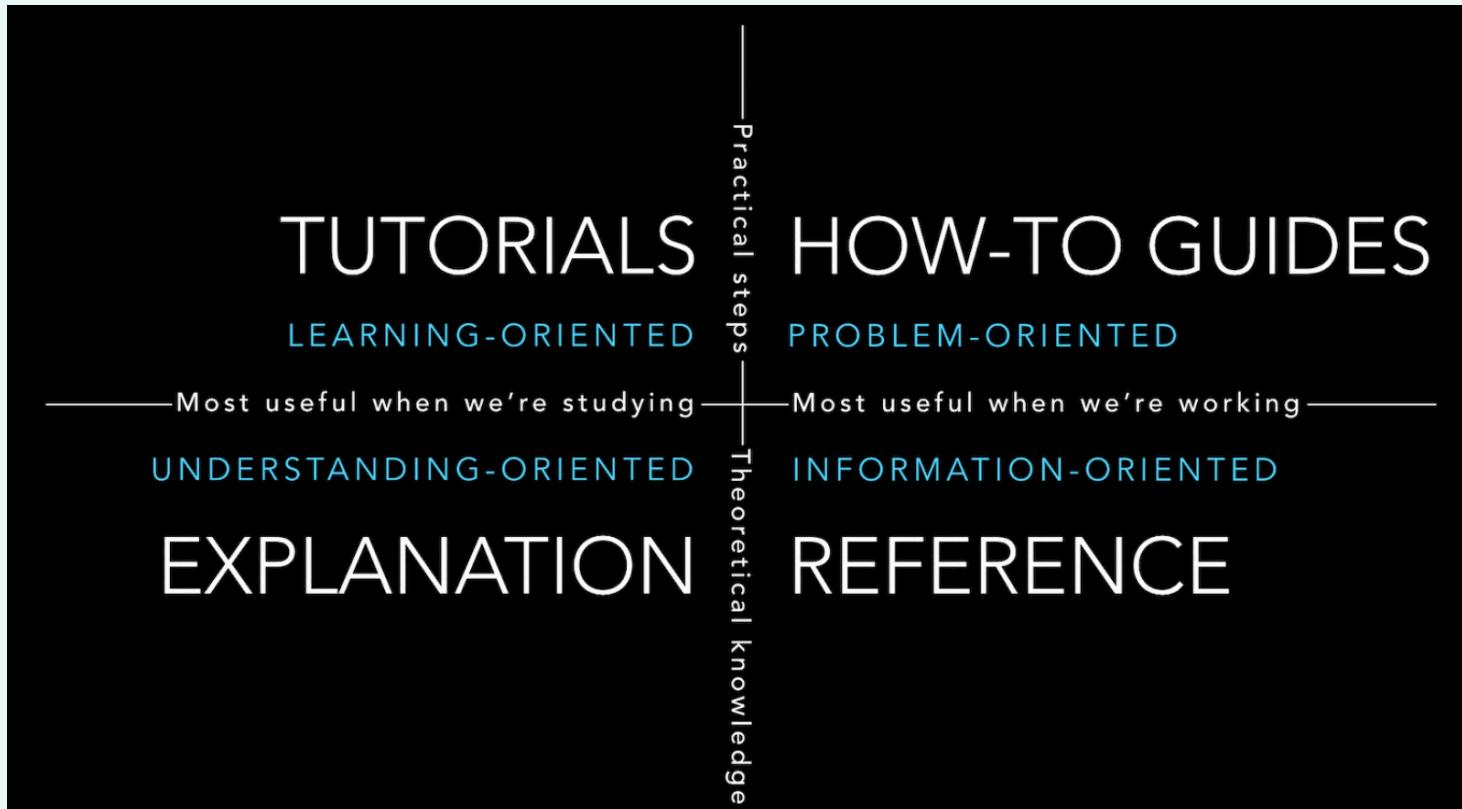
There is a need to automate the creation of these reports.



Businessman in pile of documents asking for help by [Amonrat Rungreangfangsai](#)

Automated Report Challenge

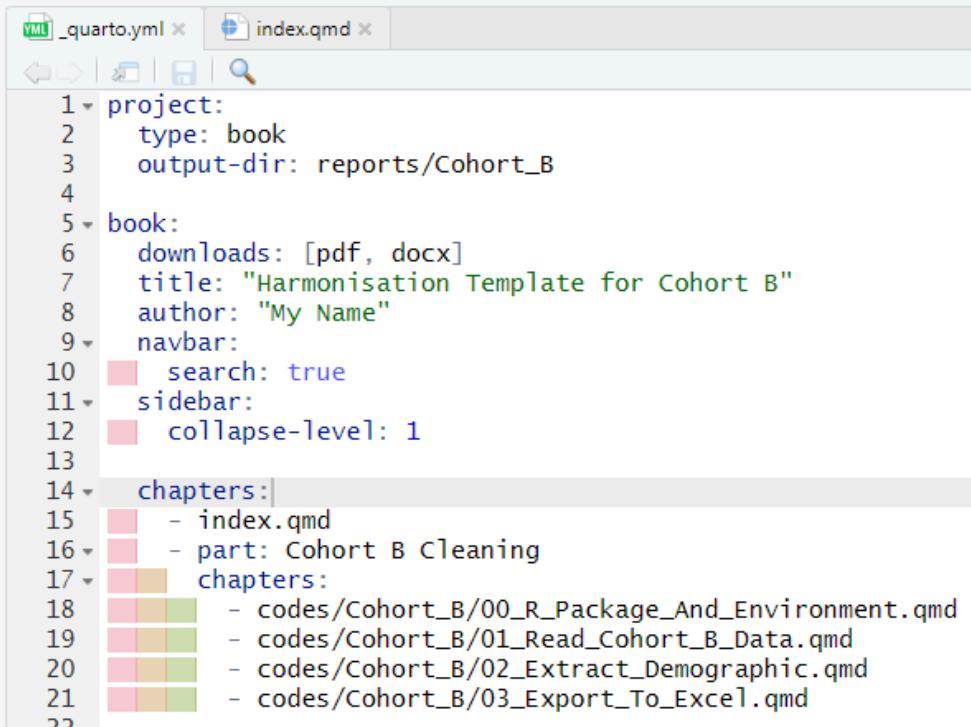
Collaborator wants different ways to report how data harmonisation is done.



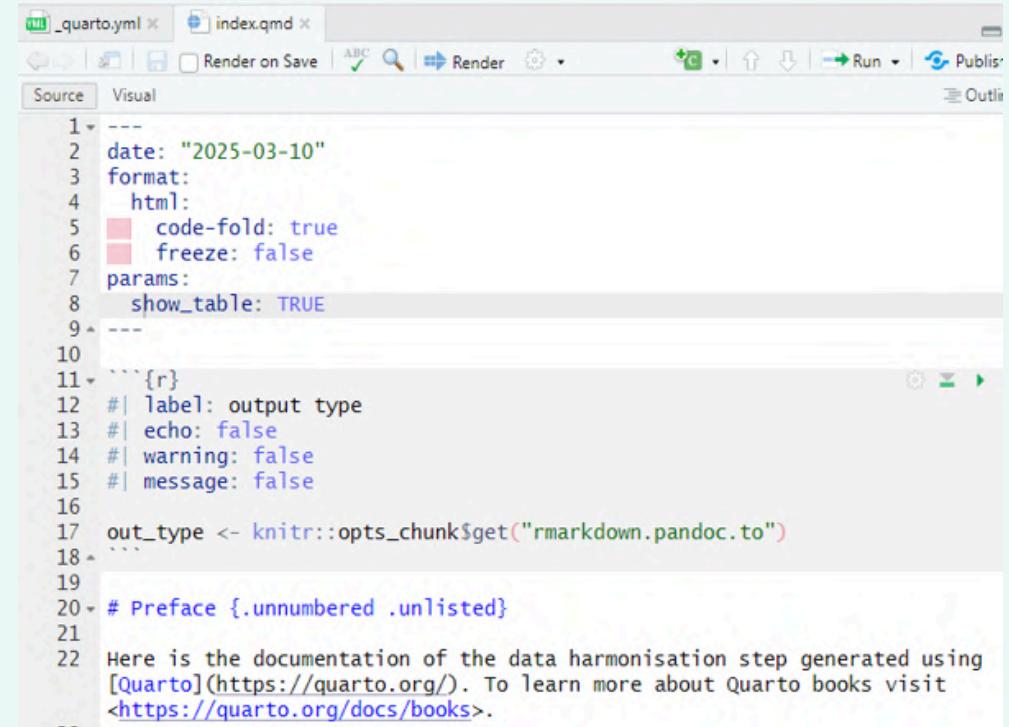
The documentation system by Divio

Quarto Books

To make a Quarto book or website, we need a `_quarto.yml` and `index.qmd` file



```
YAML _quarto.yml x index.qmd x
project:
  type: book
  output-dir: reports/Cohort_B
book:
  downloads: [pdf, docx]
  title: "Harmonisation Template for Cohort B"
  author: "My Name"
  navbar:
    search: true
    sidebar:
      collapse-level: 1
  chapters:
    - index.qmd
    - part: Cohort_B_Cleaning
      chapters:
        - codes/Cohort_B/00_R_Package_And_Environment.qmd
        - codes/Cohort_B/01_Read_Cohort_B_Data.qmd
        - codes/Cohort_B/02_Extract_Demographic.qmd
        - codes/Cohort_B/03_Export_To_Excel.qmd
```

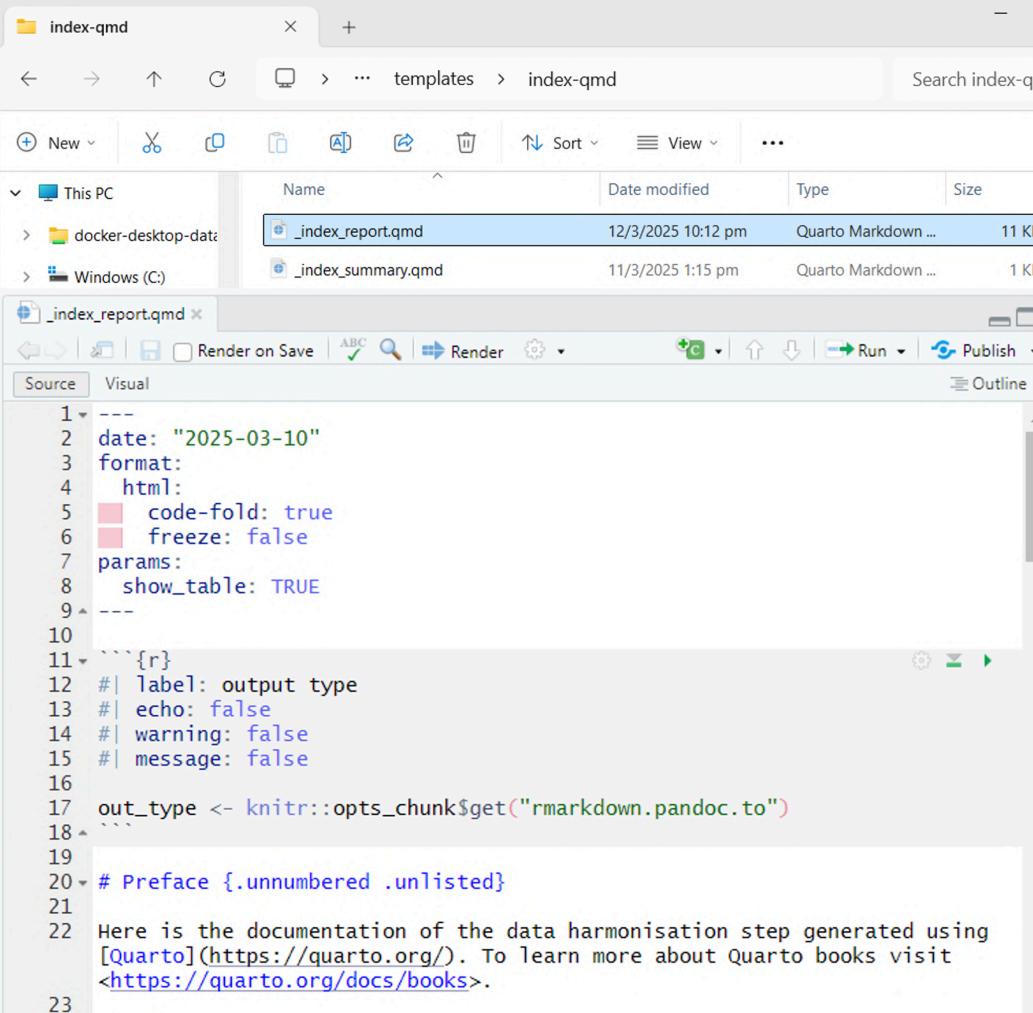


```
YAML _quarto.yml x index.qmd x
Source Visual
--- date: "2025-03-10" format: html
  code-fold: true
  freeze: false
  params:
    show_table: TRUE
---
``{r}
#| label: output type
#| echo: false
#| warning: false
#| message: false
out_type <- knitr::opts_chunk$get("rmarkdown.pandoc.to")
``

# Preface {.unnumbered .unlisted}
Here is the documentation of the data harmonisation step generated using [Quarto](https://quarto.org/). To learn more about Quarto books visit <https://quarto.org/docs/books>.
```

Automated Technical Report (Reference)

We create an `index.qmd` file for technical report generation.

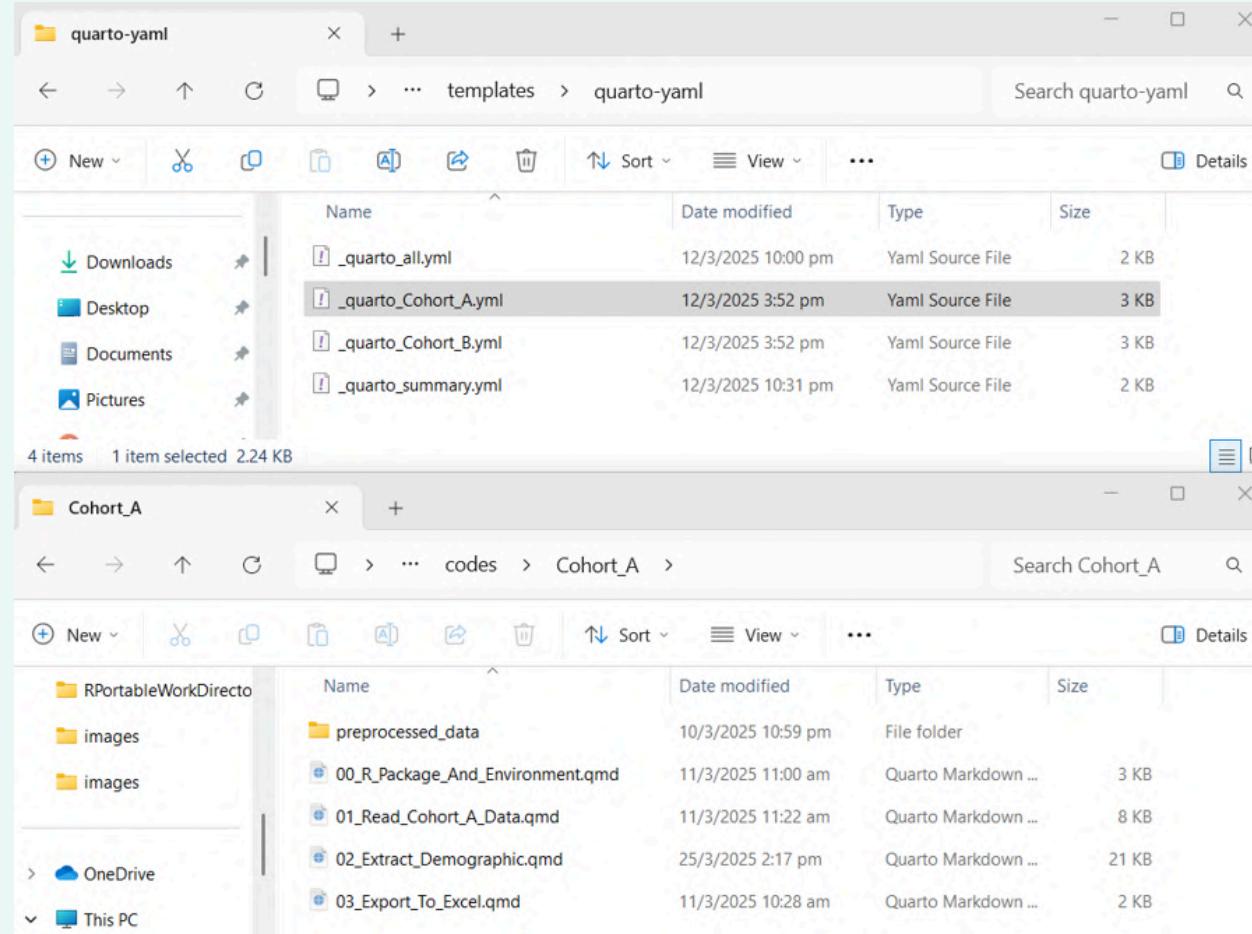


The screenshot shows a Windows File Explorer window titled "index-qmd" containing two files: "_index_report.qmd" and "_index_summary.qmd". Both files are Quarto Markdown files. Below the file list is the Quarto editor interface for the selected file, "_index_report.qmd". The editor has tabs for "Source" and "Visual". The "Source" tab displays the following code:

```
1 ---  
2 date: "2025-03-10"  
3 format:  
4   html:  
5     code-fold: true  
6     freeze: false  
7 params:  
8   show_table: TRUE  
9 ---  
10 ````{r}  
11 #| label: output type  
12 #| echo: false  
13 #| warning: false  
14 #| message: false  
15  
16 out_type <- knitr::opts_chunk$get("rmarkdown.pandoc.to")  
17 ````  
18 # Preface {.unnumbered .unlisted}  
19  
20 Here is the documentation of the data harmonisation step generated using  
21 [Quarto](https://quarto.org/). To learn more about Quarto books visit  
22 <https://quarto.org/docs/books>.  
23
```

Automated Technical Report (Reference)

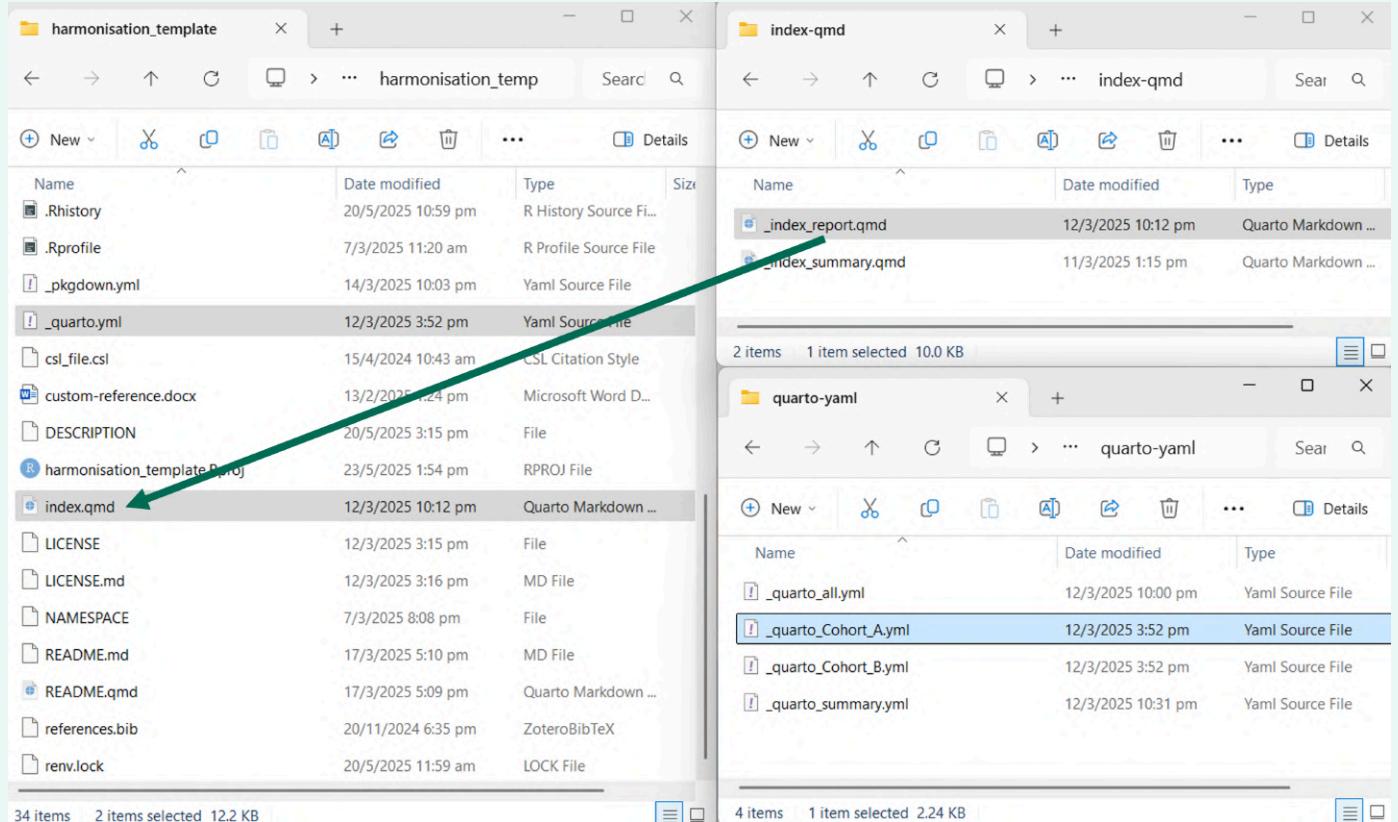
We create a `_quarto.yml` file and relevant Quarto files for each cohort.



Automated Technical Report (Reference)

Create an automated script to generate a technical report in pdf, word and html for each cohort

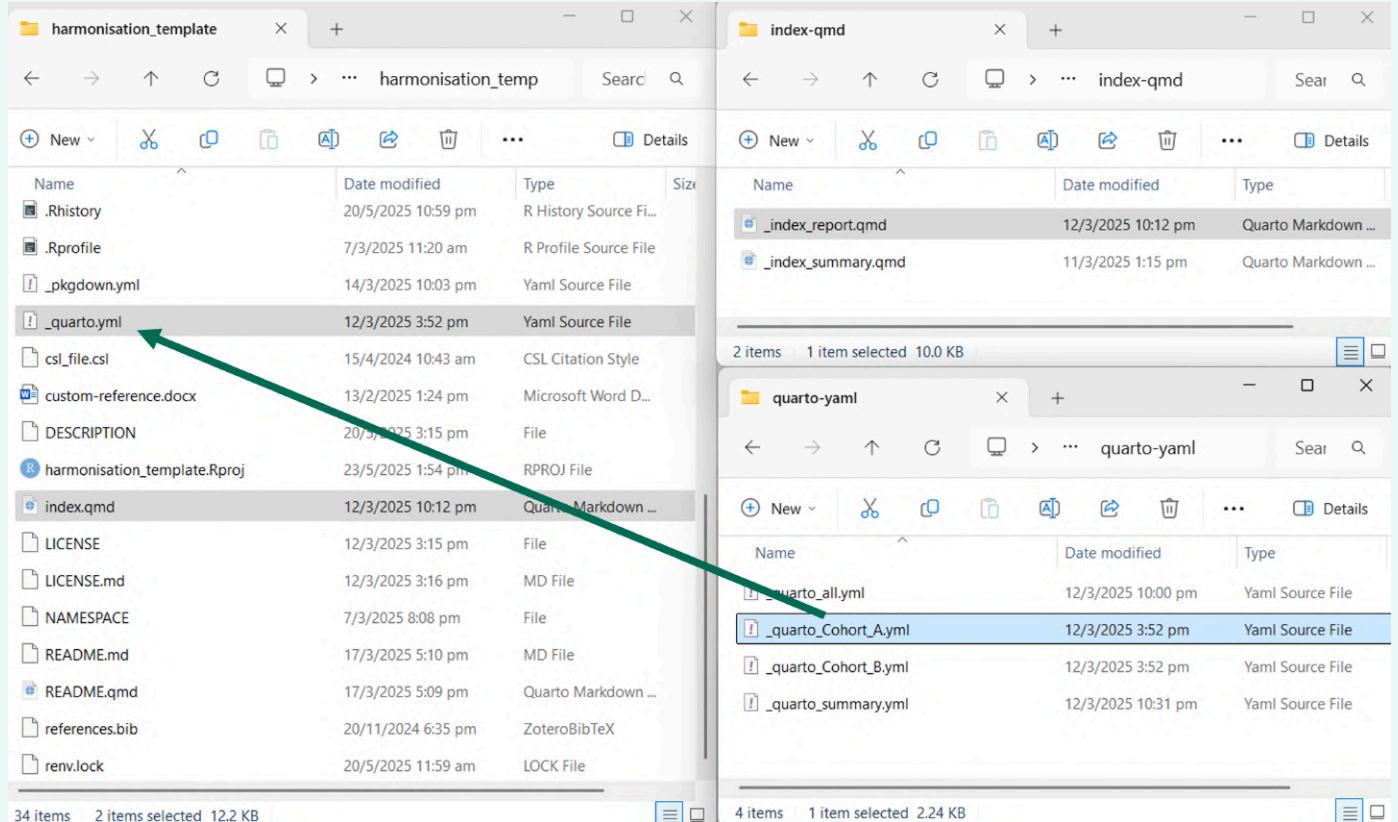
```
1 # Copy the right index.qmd
2 # file
3
4 index_qmd_file <- paste0(
5   "_index_",
6   "report",
7   ".qmd"
8 )
9
10 fs::file_copy(
11   path = here::here(
12     "templates",
13     "index-qmd",
14     index_qmd_file),
15   new_path = here::here(
16     "index.qmd"
17 ),
18   overwrite = TRUE
19 )
```



Automated Technical Report (Reference)

Create an automated script to generate a technical report in pdf, word and html for each cohort

```
1 copy_and_render <- function(
2   cohort
3 ) {
4 
5   # Copy quarto.yml file
6   # for each cohort
7 
8   quarto_yml_file <- paste0(
9     "_quarto_",
10    cohort,
11    ".yml"
12  )
13 
14 fs::file_copy(
15   path = here::here(
16     "templates",
17     "quarto-yaml",
18     quarto_yml_file),
19   new_path = here::here("_quarto.yml"),
20   overwrite = TRUE
21 )
22 
23 # Render each cohort
24 quarto::quarto_render(
25   as_job = FALSE
26 )
```

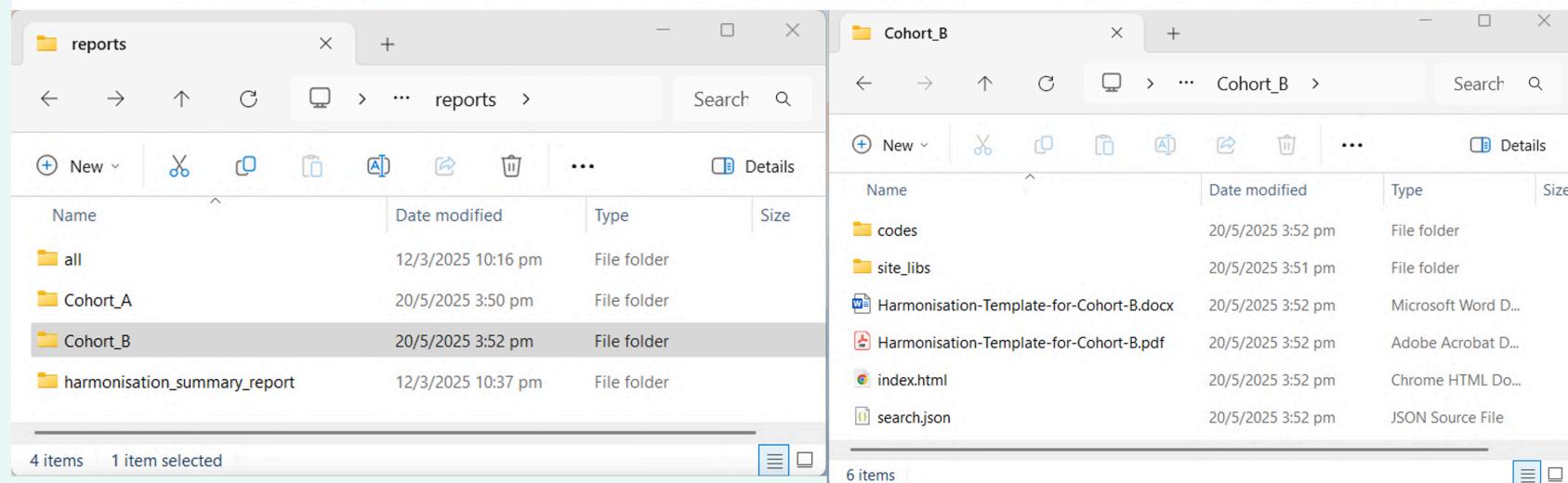


Automated Technical Report (Reference)

Output of these reports are as follows:

Run the R script `cohort_harmonisation_script.R` in `codes` folder to generate:

- Cohort_A Harmonisation Report:
 - HTML: <https://jauntyjjs-harmonisation-cohort-a.netlify.app>
 - PDF : <https://jauntyjjs-harmonisation-cohort-a.netlify.app/Harmonisation-Template-for-Cohort-A.pdf>
 - Word: <https://jauntyjjs-harmonisation-cohort-a.netlify.app/Harmonisation-Template-for-Cohort-A.docx>
- Cohort_B Harmonisation Report:
 - HTML: <https://jauntyjjs-harmonisation-cohort-b.netlify.app>
 - PDF : <https://jauntyjjs-harmonisation-cohort-b.netlify.app/Harmonisation-Template-for-Cohort-B.pdf>
 - Word: <https://jauntyjjs-harmonisation-cohort-b.netlify.app/Harmonisation-Template-for-Cohort-B.docx>



Automated Summary Report (How-to-Guide)

A similar method is done to create a summary report in word using **flextable**.

2.4 Smoking History

smoke_current is the harmonised data field to denote if the patient is a current smoker during the time of the CT scan. *smoke_past* is the harmonised data field to denote if the patient is a past smoker during the time of the CT scan.

They hold the following values:

Table S6: Harmonised values of *smoke_current* and *smoke_past*.

| Value | Description |
|-------|-------------|
| 0 | no |
| 1 | yes |
| -1 | unknown |

They are harmonised as follows:

Table S7: Harmonised process of *smoke_current* and *smoke_past*.

| Cohort ID | Original Response | Harmonisation Response |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| Cohort A | Column <i>smoke_current_good</i> with
0 as no.
1 as yes.
-1 as unknown.
Column <i>smoke_past_good</i> with
0 as no.
1 as yes.
-1 as unknown. | <i>smoke_current</i> will take the values of <i>smoke_current_good</i> .
<i>smoke_past</i> will take the values of <i>smoke_past_good</i> . |
| Cohort B | Column <i>Smoke History</i> with
<i>non-smoker</i> as non-smoker. | Map the values of <i>Smoke History</i> to <i>smoke_current</i> as follows: |

past smoker as a past smoker.

current smoker as a current smoker.

NA as unknown.

non-smoker and *past smoker* as 0.

current smoker as 1.

NA as -1.

Map the values of *Smoke History* to *smoke_past* as follows:

non-smoker and *current smoker* as 0.

past smoker as 1.

NA as -1.

After harmonisation, we validate the values of *smoke_current* and *smoke_past* to ensure that there can only be the following cases:

Table S8: Valid values of *smoke_current* and *smoke_past*.

| Description | <i>smoke_current</i> | <i>smoke_past</i> |
|----------------|----------------------|-------------------|
| Non-smoker | 0 | 0 |
| Past smoker | 0 | 1 |
| Current smoker | 1 | 0 |
| Unknown | -1 | -1 |

Data Variable Justification Report (Explanation)

Initially we have `smoke_history` with values 0 (Non-smoker), 1 (Current smoker), 2 (Past smoker) and -1 (Unknown).

However, in the early stage of the study, some collaborators could only provide if the patient has a smoking history but could not specify if the patient is a current or past smoker.

| Patient | smoke_history | Problem |
|----------------|---------------|----------------------|
| Non-smoker | 0 | No smoking history |
| Current smoker | 1 | Have smoking history |
| Past smoker | 2 | Have smoking history |
| Missing | -1 | Missing |

Data Variable Justification Report (Explanation)

To deal with this case, we have `smoke_history`, `smoke_current` and `smoke_past` to hold values 1 (Yes), 0 (No) and -1 (Unknown).

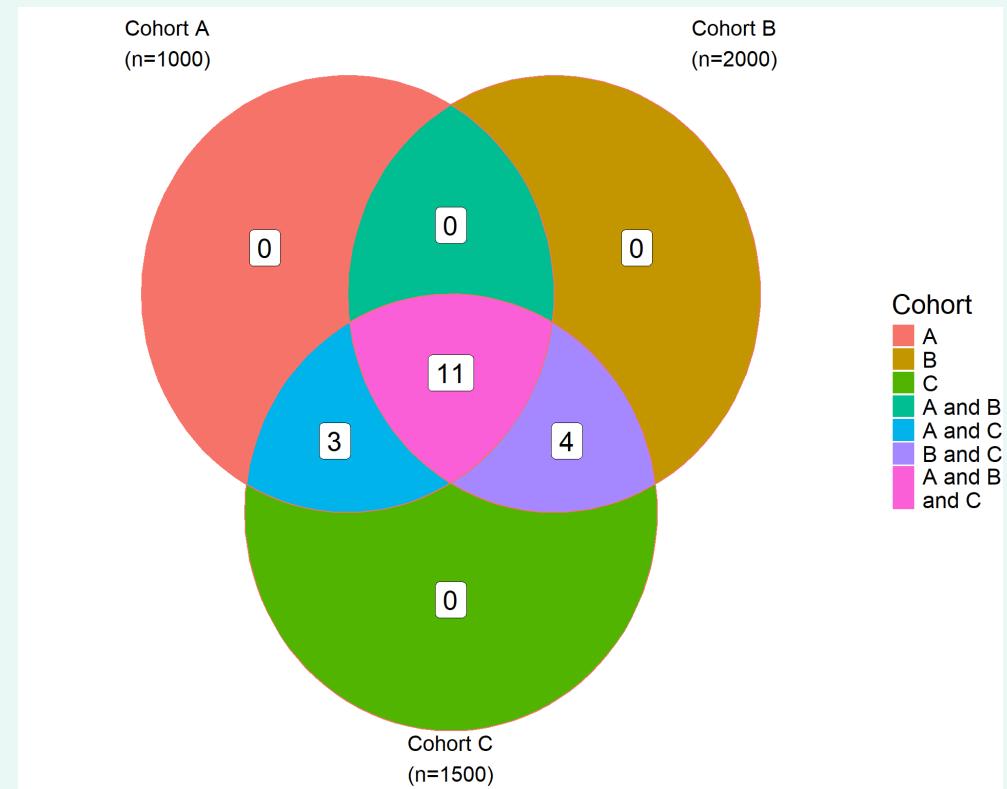
| Patient | smoke_history | smoke_current | smoke_past |
|----------------------|---------------|---------------|------------|
| Current smoker | 1 | 1 | 0 |
| Past smoker | 1 | 0 | 1 |
| Non-smoker | 0 | 0 | 0 |
| Have smoking history | 1 | -1 | -1 |
| Missing | -1 | -1 | -1 |

Overview Diagrams

How many variables can each cohort provide ?

How many variables can be harmonised ?

```
1 demographic_list <- list(
2   A = c("Age", "Sex",
3     "Hypertension", "Dyslipidemia", "Family Hx CAD", "Diabetes",
4     "Smoke Current", "Smoke Past",
5     "Have Chest Pain", "Chest Pain Character",
6     "Dyspnea",
7     "BMI", "Height", "Weight"),
8   B = c("Age", "Sex",
9     "Hypertension", "Dyslipidemia", "Family Hx CAD", "Diabetes",
10    "Smoke Current", "Smoke Past",
11    "Have Chest Pain", "Chest Pain Character",
12    "Dyspnea",
13    "HDL", "Total Cholesterol",
14    "Triglyceride", "LDL"),
15   C = c("Age", "Sex",
16     "Hypertension", "Dyslipidemia", "Family Hx CAD", "Diabetes",
17     "Smoke Current", "Smoke Past",
18     "Have Chest Pain", "Chest Pain Character",
19     "Dyspnea",
20     "BMI", "Height", "Weight",
21     "HDL", "Total Cholesterol",
22     "Triglyceride", "LDL")
23 )
24
25 cohort_a_label_name <- "Cohort A\n(n=1000)"
26 cohort_b_label_name <- "Cohort B\n(n=2000)"
```

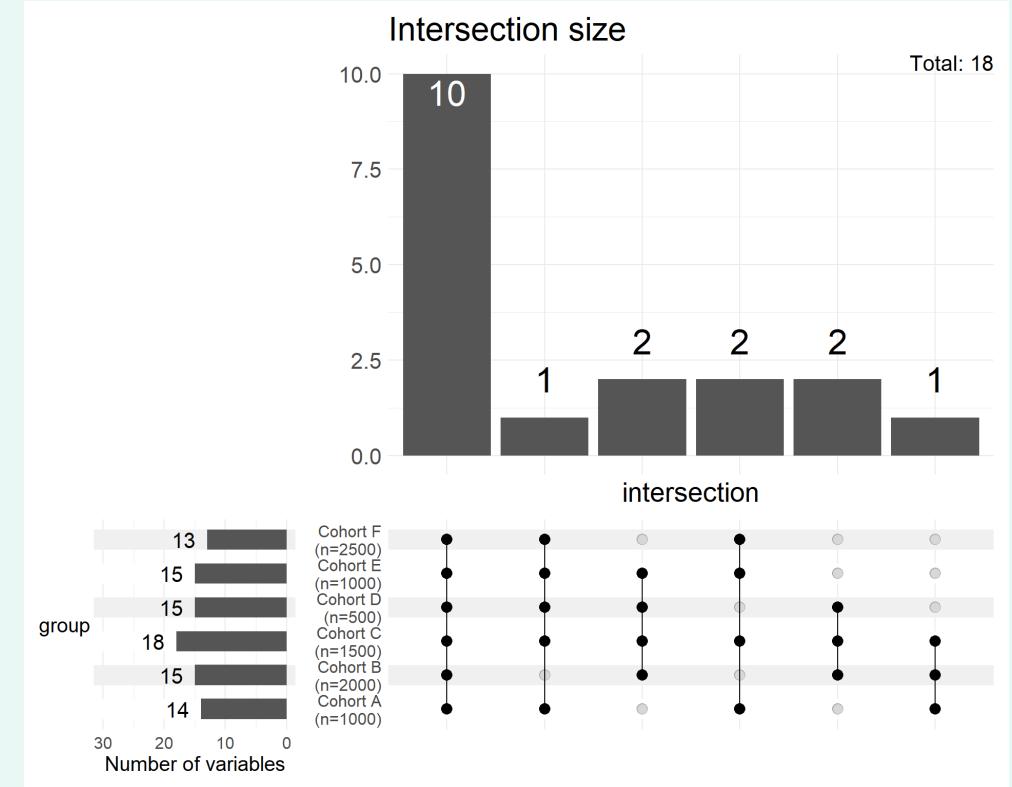


But it does not work for many (> 10) cohorts.

Overview Diagrams

Upset plots are too complicated for clinicians.

```
1 demographic_venn <- tibble::tibble(  
2   column_name = c("Age", "Sex",  
3     "Hypertension", "Dyslipidemia", "Family Hx CAD", "Diabetes",  
4     "Smoke Current", "Smoke Past",  
5     "Have Chest Pain", "Chest Pain Character",  
6     "Dyspnea",  
7     "BMI", "Height", "Weight",  
8     "HDL", "Total Cholesterol",  
9     "Triglyceride", "LDL"),  
10   `Cohort A` = c(1, 1,  
11     1, 1, 1, 1,  
12     1, 1,  
13     1, 1,  
14     1,  
15     1, 1, 1,  
16     0, 0,  
17     0, 0),  
18   `Cohort B` = c(1, 1,  
19     1, 1, 1, 1,  
20     1, 1,  
21     1, 1,  
22     1,  
23     0, 0, 0,  
24     1, 1,  
25     1, 1),  
26   `Cohort C` = c(1, 1).
```



Cannot answer follow-up questions:

How many cohorts provide patient's blood lipid information and how many patients have this information ?

Overview Diagrams

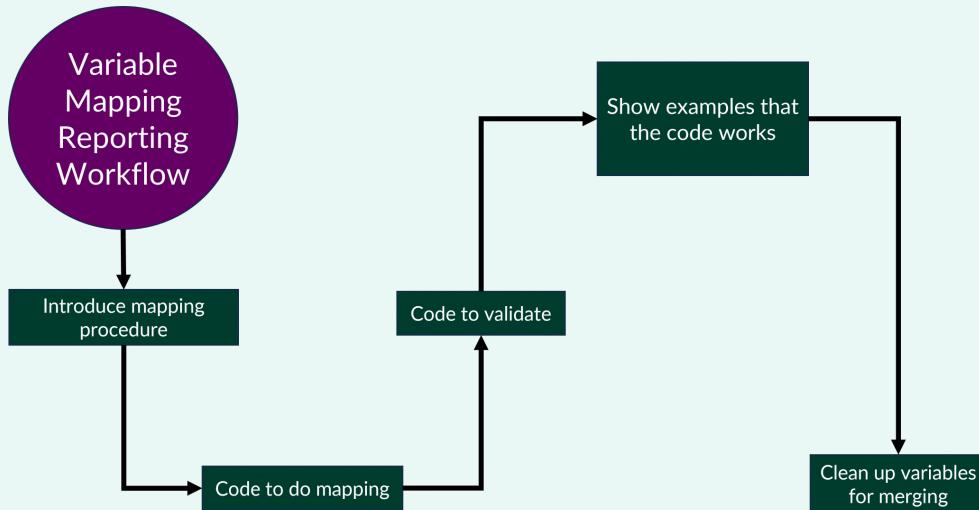
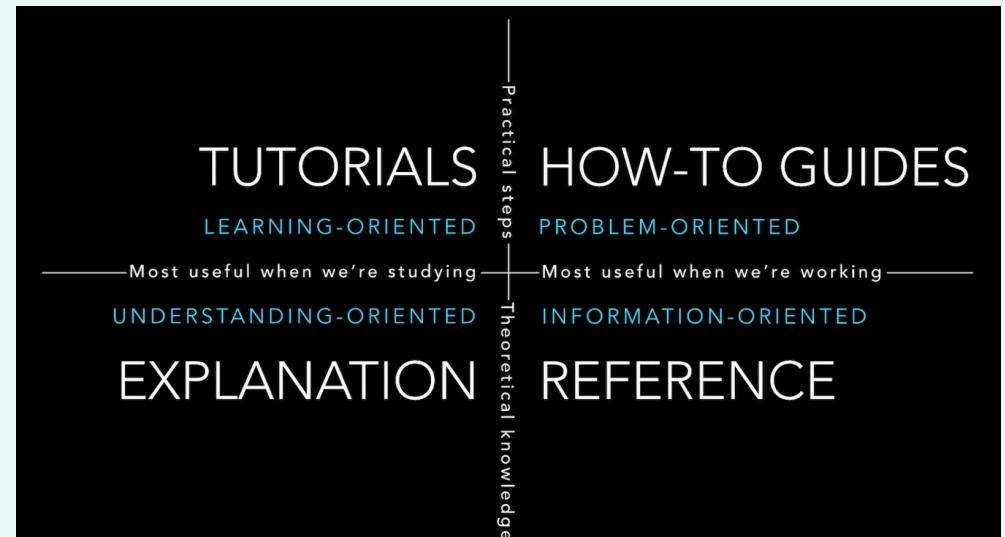
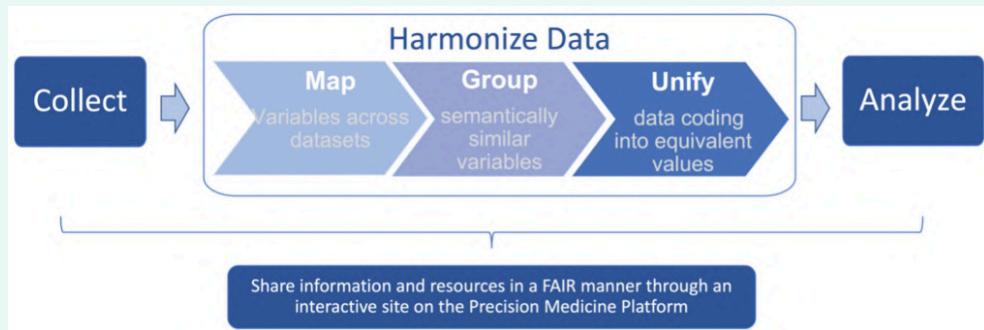
Create a “heatmap” using Microsoft PowerPoint.

| | | | | | | | | | |
|--------------|----------|------|----|----------------------|-------------|-------------|-------------|-------------|--------------|
| | | | | 10 | | | | | |
| | | | | Age | | | | | |
| | | | | Sex | | | | | |
| | | | | Hypertension | | | | | |
| | | | | Dyslipidemia | | | | | |
| | | | | Family Hx CAD | | | | | |
| | | | | Diabetes | | | | | |
| | | | | Smoke Current | | | | | |
| | | | | Smoke Past | | | | | |
| | | | | Have Chest Pain | 1 | 1 | 2 | 2 | 2 |
| | | | | Chest Pain Character | Dyspnea | BMI | Height | HDL | Triglyceride |
| Country A | Cohort A | 1000 | 15 | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ |
| | Cohort B | 2000 | 16 | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ |
| Country B | Cohort C | 1500 | 18 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Country C | Cohort D | 500 | 16 | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ |
| Country D | Cohort E | 1000 | 16 | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ |
| | Cohort F | 2500 | 14 | ✓ | ✗ | ✓ | ✓ | | |
| Total | | | | 8500 | 4500 | 6500 | 6000 | 5000 | 4000 |

| Variable Colour Legend | |
|------------------------|--------------|
| Age | Green |
| Sex | Yellow |
| Comorbidity | Light Purple |
| Smoking history | Grey |
| Symptoms | Cyan |
| Obesity | Pink |
| Blood lipid | Orange |

| Table Legend | |
|--------------|-----------------|
| ✓ | Available |
| ✗ | Not available |
| | Pending arrival |

Summary



Variable Colour Legend

| |
|-----------------|
| Age |
| Sex |
| Comorbidity |
| Smoking history |
| Symptoms |
| Obesity |
| Blood lipid |

Table Legend

| | |
|------|-----------------|
| ✓ | Available |
| ✗ | Not available |
| Grey | Pending arrival |

| Country | Cohort | N | Variables | Chest Pain Character | 10 | | Height | HDL | Triglyceride |
|--------------|----------|------|-----------|----------------------|---------|------|--------|------|--------------|
| | | | | | Dyspnea | BMI | | | |
| Country A | Cohort A | 1000 | 15 | ✓ | ✓ | ✓ | 2 | 2 | 2 |
| | Cohort B | 2000 | 16 | ✓ | ✓ | ✗ | | | |
| Country B | Cohort C | 1500 | 18 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | Cohort D | 500 | 16 | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ |
| Country C | Cohort E | 1000 | 16 | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ |
| | Cohort F | 2500 | 14 | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ |
| Total | | | | | 8500 | 4500 | 6500 | 6000 | 5000 |
| | | | | | | | | | 4000 |

Thank you

Harmonisation project template on GitHub: <https://github.com/JauntyJJS/harmonisation/>

README License MIT license

Data Harmonisation Project Template

R-CMD-check.yaml passing

Table of Content

- [Motivation](#)
- [Acknowledgement](#)
- [File Structure](#)
- [Software Installation](#)
- [R Package Installation](#)
- [Using `renv`](#)
- [R Functions Management](#)
- [R Packages Used](#)
- [R Platform Information](#)
- [Data Harmonisation Report For Each Cohort](#)
- [Combined Data Harmonisation Report For All Cohort](#)
- [Data Harmonisation Summary](#)
- [General Recommendations](#)



Feature Complete from [MonkeyUser.com](https://monkeyuser.com)