R Lesson 5: Data Wrangling (part 2)

vanderbi.lt/r

Steve Baskauf

Download ICPSR data for later

- Instructions link on the lessons homepage (vanderbi.lt/r)
- Create ICPSR account
- Download 2 files for National Longitudinal Study of Adolescent to Adult Health, 1994-2008 (**`21600-0001-Data.tsv`** and **`21600-0022-Data.tsv`**)

Modifying tibbles (dplyr)

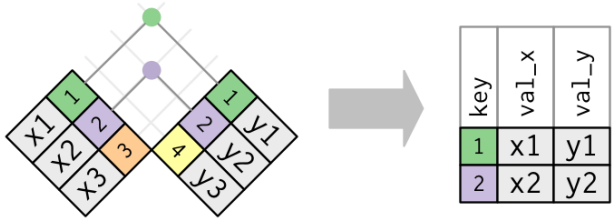
dplyr functions

- **`filter()`** subsets rows
- **`select()`** subsets columns
- **`mutate()`** calculates new columns or changes existing ones

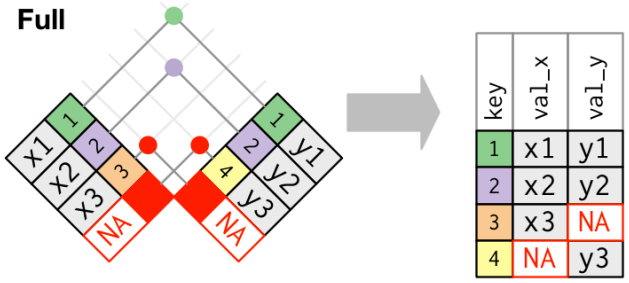
Examples with schools data

Joins

- Joins merge data from multiple tables (tibbles)
- **Keys** are the columns used to match table rows
- **Inner join** only outputs rows with matching keys



- **Full outer join** includes rows that don't match (with NA values inserted)



- Many other permutations
- See https://r4ds.had.co.nz/relational-data.html for explanation and examples (diagrams from there)

Join format

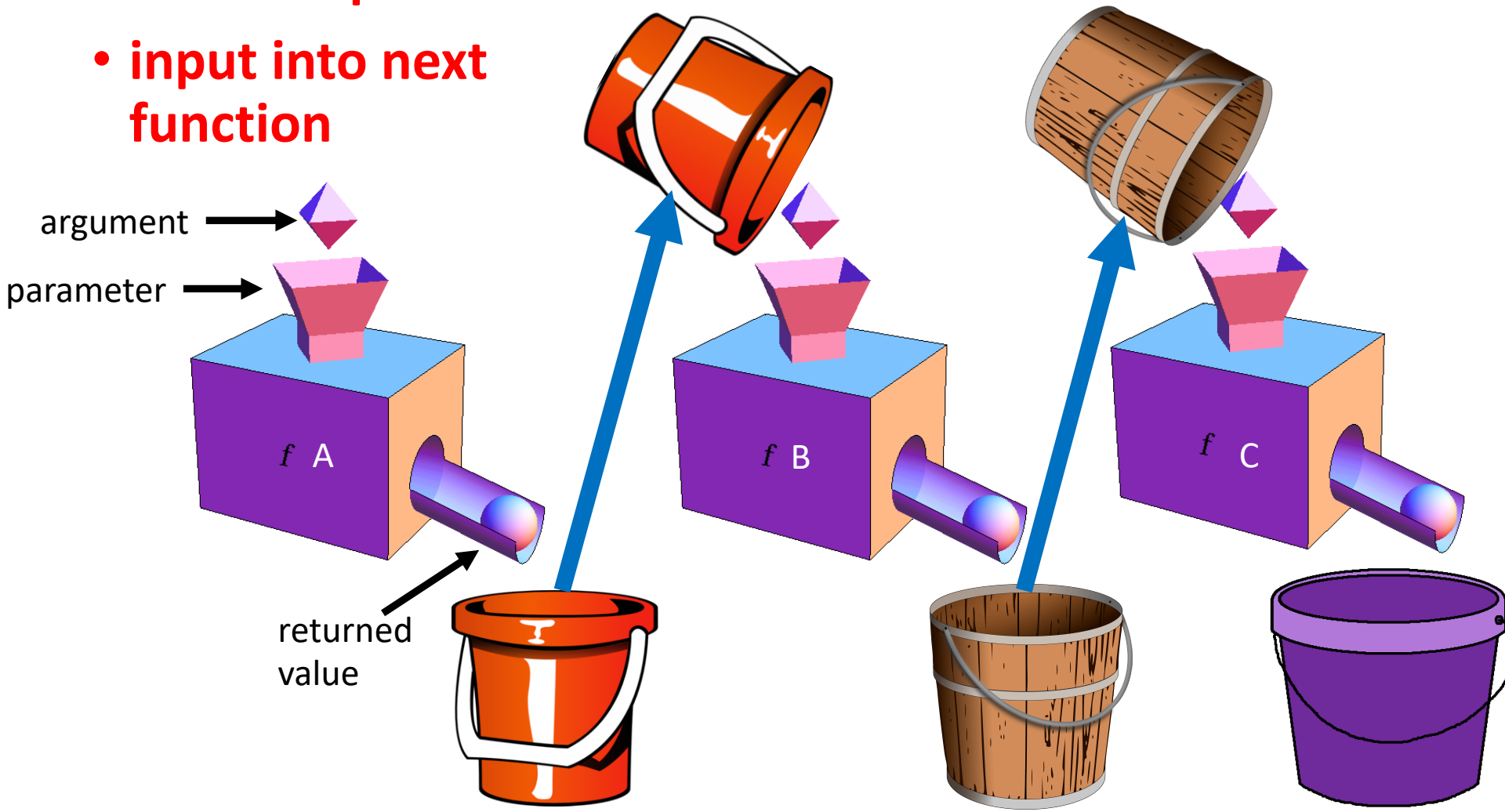
```
full_join(womens_data, poverty_data,
          by = c("country"="Country Name"),
          copy = FALSE,
          suffix = c(".wom", ".pov") )
```

- First two arguments are the two tibbles to join
- **`by`** value are columns to join by; use = if names differ
- **`suffix`** value is added to columns with duplicate names
- other join types: **`inner_join()`**, **`left_join()`**, …

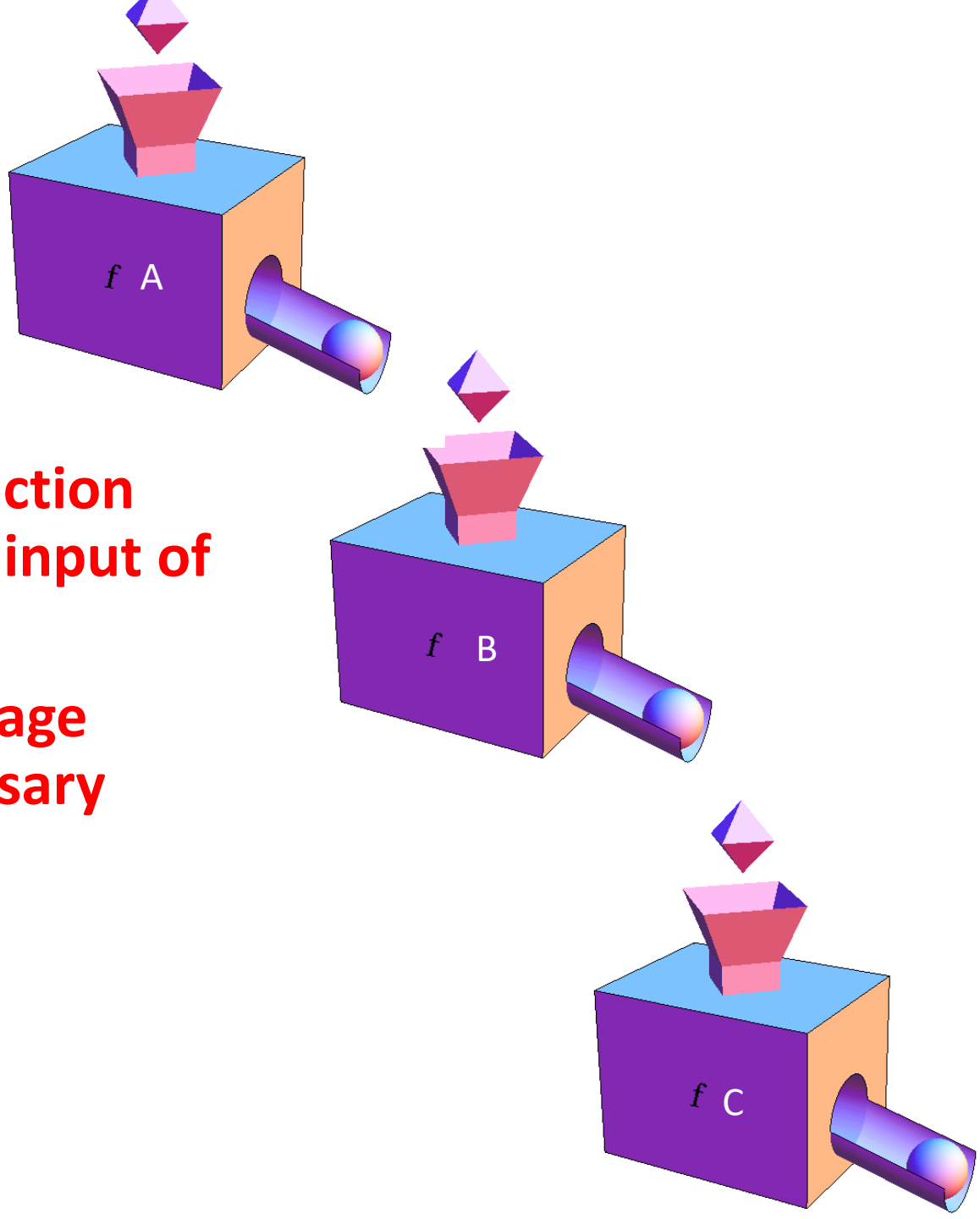
piplines (magrittr)

Classic function/variable interaction

- **store output**
- **input into next function**



Piping

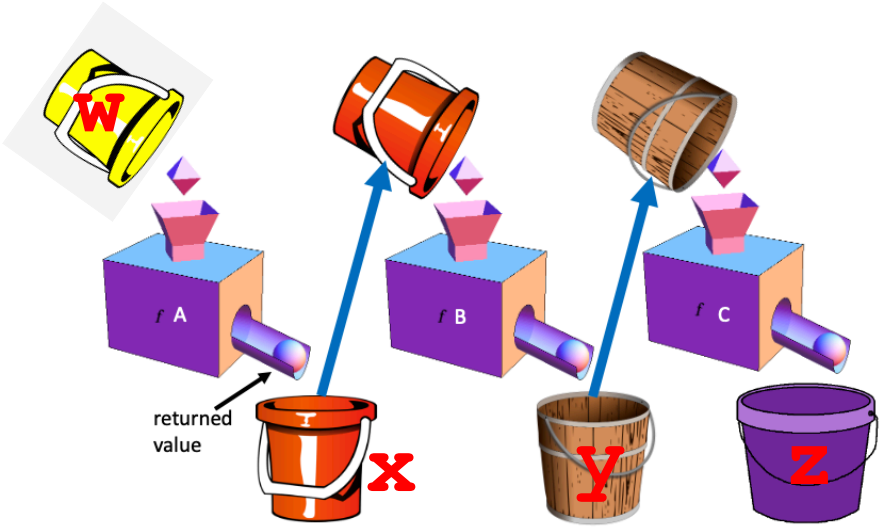


- **output of one function goes directly into input of next**
- **intermediate storage objects not necessary**

Examples

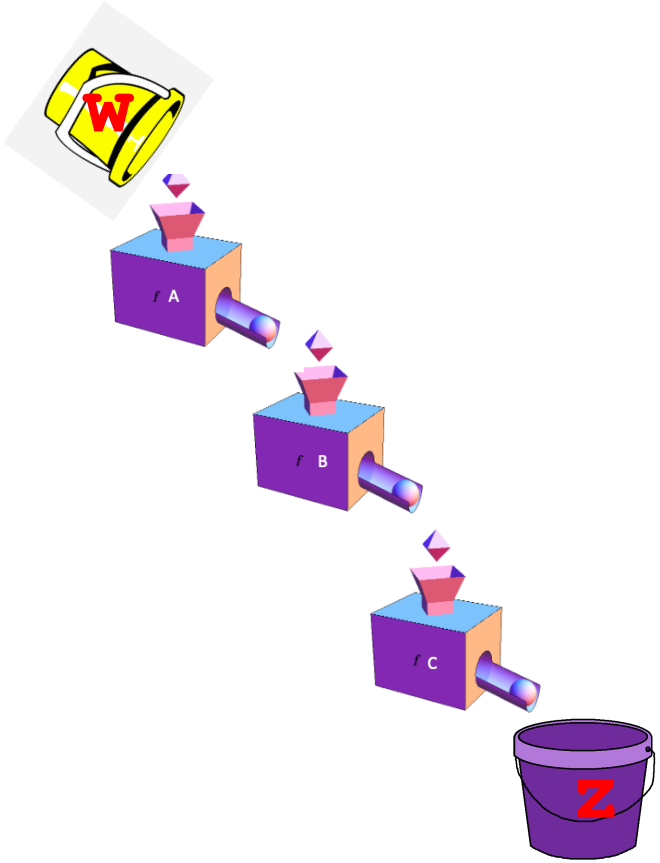
- Classic

```
x <- function_a(w, p)
y <- function_b(x, q)
z <- function_c(y, r)
```



- Piping

```
z <- w %>%
     function_a(p) %>%
     function_b(q) %>%
     function_c(r)
```



- Notice that no intermediate storage object needs to be input into the piped function

Examples with schools data

Issues with the NLSAAH dataset

- **National Longitudinal Study of Adolescent to Adult Health**, 1994-2008 = big longitudinal study
- Datasets are huge and difficult to work with, so need to **extract** subset of data
- Data are coded using numbers – need to **transform** to other forms.
- Data are in separate CSV files that need to be **joined**
- We might want to **create new data fields by calculation** from others.

Assignment #1: Extract data

- We want data on age, sex, and parental relationships (from DS0001) and on smoking and body characteristics (from DS0022). See top of script.
- The AID unique identifier is the key to join the two tables.
- Want to save extracted data in file so that we don't need to load the big datasets into memory.

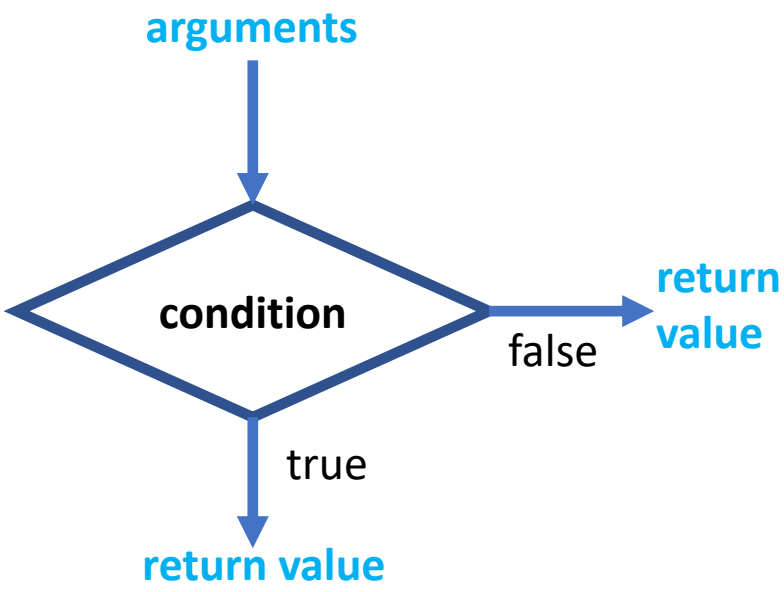
Assignment #2 Calculate BMI

- We want to know BMI to see if it's related to smoking and other factors.
- Need to replace codes for missing values to NA (documentation gives codes for each question)
- Also recode "1" and "2" for sex to "male" and "female" to make interpretation clearer.
- Calculate age
- Convert height and weight to SI (metric)
- Calculate BMI using mass_kg/height_m^2
- Need to use piping because of so many operations

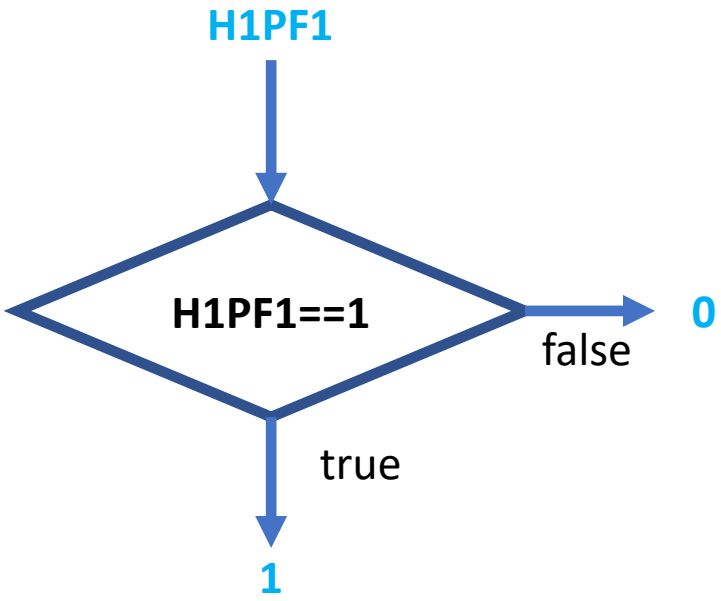
Assignment #3 Calculate "maternal closeness"

- For each maternal relationship question recode 1 to 1 (good) and other values to 0 (1 is most common value).
- Complication: NA values must be preserved during calculation.
- Maternal closeness is 1 for people who have 1 values for every maternal relationship question.
- Need to have an "if" function for this (`ifelse`).

`ifelse()` function



general pattern



specific example

```
ifelse(H1PF1==1, 1, 0)
```

boolean operators for conditions

! is **NOT**

& is **AND**

| is **OR**

- Examples:

`!H1PF5==1` H1PF5 isn't equal to 1

`!is.na(H1PF5)` H1PF5 doesn't have an NA value

`H1PF3==1 & H1PF4==1` both H1PF3 and H1PF4 equal 1

`is.na(H1PF2)|is.na(H1PF3)` either H1PF2 or H1PF3 is NA