NEEDBios

R Lab

Victor Ritter

UNC-CH

Jan 2020

Part I-a

R Packages and the tidyverse

R packages

- power of R is on its packages

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

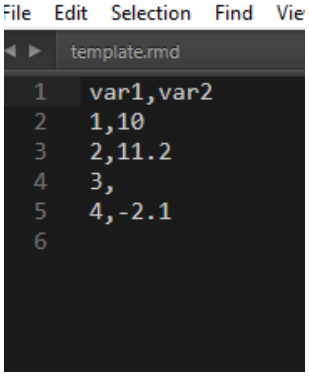
R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in

- Think of a task and there is probably a package that can help you do it.
- All packages are open-source
- Some packages are better then others (check for updates and quality of its documentation)
- Easy to install packages via RStudio

Importing external data - txt




```
read.table(file = "./data/dummy_data.csv",
           header = T, sep = ",")
```

```
>   var1 var2
> 1    1 10.0
> 2    2 11.2
> 3    3   NA
> 4    4 -2.1
```

--

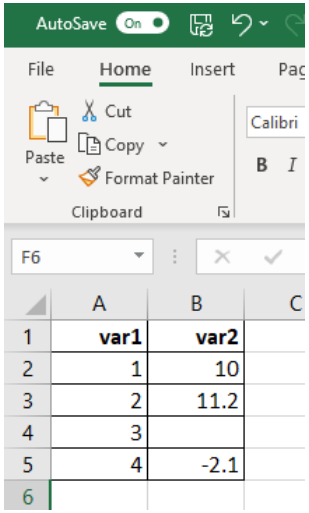
```
read.csv(file = "./data/dummy_data.csv")
```

```
>   var1 var2
> 1    1 10.0
> 2    2 11.2
> 3    3   NA
> 4    4 -2.1
```

Importing external data - excel

- Requires external packages: e.g. **readxl**

```
install.packages("readxl")
library(readxl)
```




```
read_xlsx(path = "./data/dummy_data.xlsx")
```

```
> # A tibble: 4 x 2
>    var1  var2
>   <dbl> <dbl>
> 1     1  10
> 2     2  11.2
> 3     3  NA
> 4     4  -2.1
```

Importing external data - spss/sas/stata

- Requires package **haven**

```
library(haven)

data_sas <- read_sas("dados/datafile.sas7bdat")
data_spss <- read_spss("dados/datafile.sav")
```

Importing external data - from the internet

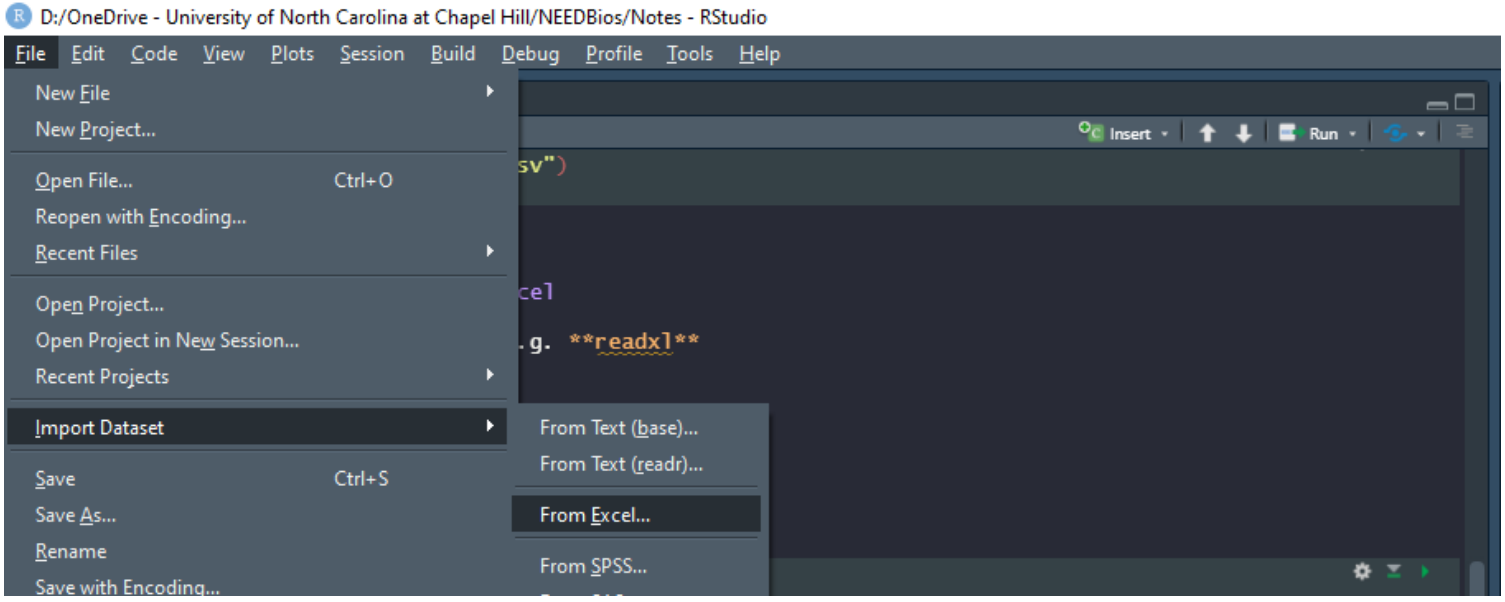
- For a URL that has the data file, just use the file address

```
read.csv("https://mywebsite.com/iris.csv")
```

- Web scrapping can be done using packages like **rvest**, **rjason**, **jsonlite**

Importing external data using RStudio

- Can also be done via RStudio's File menu



Checking the data

- From **base R**

```
h1n1 <- read.csv("./data/h1n1_usa.csv")
head(h1n1)
```

```
>         State Cases Deaths Population
> 1     Alabama   477      0    4661900
> 2      Alaska   272      0     686293
> 3     Arizona   947     15    6500180
> 4    Arkansas   131      0    2855390
> 5  California  3161     52   36756666
> 6    Colorado   171      0    4939456
```

- From the **tidyverse** package

```
glimpse(h1n1)
```

```
> Observations: 51
> Variables: 4
> $ State      <fct> Alabama, Alaska, Arizona, Arkansas, Ca…
> $ Cases      <int> 477, 272, 947, 131, 3161, 171, 1713, 3…
> $ Deaths     <int> 0, 0, 15, 0, 52, 0, 8, 0, 0, 23, 1, 3,…
> $ Population <int> 4661900, 686293, 6500180, 2855390, 367…
```

Checking the data

- Can also inspect the dataset using RStudio's Environment tab
- Or do `View(h1n1)`
- Try the `summary()` function from base R

```
summary(h1n1)
```

```
>          State         Cases              Deaths          Population
>  Alabama    : 1   Min.   :  45.0   Min.   : 0.000   Min.   :  532668
>  Alaska     : 1   1st Qu.: 179.5   1st Qu.: 0.000   1st Qu.: 1653624
>  Arizona    : 1   Median : 283.0   Median : 1.000   Median : 4269245
>  Arkansas   : 1   Mean   : 856.7   Mean   : 5.922   Mean   : 5961955
>  California: 1   3rd Qu.: 856.5   3rd Qu.: 5.500   3rd Qu.: 6524702
>  Colorado   : 1   Max.   :6222.0   Max.   :63.000   Max.   :36756666
>  (Other)    :45
```

Exporting data

- As CSV with `write.csv()`
- As an R object with `write.rds()`

Part I-b

Introduction to tidyverse

R before/after Hadley Wickham

- Hadley Wickham, Chief Scientist at RStudio and an adjunct Professor of statistics at the University of Auckland, Stanford University, and Rice University.

--

- *tidyverse*: collection of R packages for data science that share a common philosophy

Pipe operator %>%

UNC

- Allow you to write more readable code

```
x <- c(1, 2, 3, 4)
sqrt(sum(x))
```

```
> [1] 3.162278
```

```
x %>% sum() %>% sqrt()
```

```
> [1] 3.162278
```

- Like a recipe

```
let_cool(bake(put(mix(add(bowl(rep("farinha", 2), "water", "baking_soda",
  "milk", "oil"), "flour", until = "soft"), duration = "3min"),
  where = "pan", type = "pan", grease = TRUE), duration = "50min"),
  "fridge", "20min")

bowl(rep("flour", 2), "water", "baking_soda", "milk", "oil") %>%
  add("farinha", until = "soft") %>%
  mix(duration = "3min") %>%
  put(where = "pan", type = "pan", grease = TRUE) %>%
  bake(duration = "50min") %>%
  let_cool("fridge", "20min")
```

tibbles (tidy tables)

- From the **tibble** package (included in the **tidyverse** package)

```
h1n1 <- as_tibble(h1n1)
h1n1
```

```
> # A tibble: 51 x 4
>    State      Cases Deaths Population
>    <fct>      <int>  <int>      <int>
> 1 Alabama      477      0    4661900
> 2 Alaska       272      0     686293
> 3 Arizona      947     15    6500180
> 4 Arkansas     131      0    2855390
> 5 California  3161     52   36756666
> # … with 46 more rows
```

Shaping data with dplyr

UNC

- Main functions:
  - `filter()` - filter lines
  - `select()` - select columns
  - `arrange()` - sort dataset
  - `mutate()` - create/modify columns
  - `group_by()` - group base
  - `summarise()` - summaris(z)e data

Line filtering

```
h1n1 %>%
  filter(Deaths > 10)
```

```
> # A tibble: 8 x 4
>   State      Cases Deaths Population
>   <fct>      <int>  <int>      <int>
> 1 Arizona      947     15    6500180
> 2 California  3161     52   36756666
> 3 Florida     2915     23   18328340
> 4 Illinois    3404     17   12901563
> 5 New Jersey  1414     15    8682661
> # … with 3 more rows
```

```
h1n1 %>%
  filter(Deaths > 10, Population <= 1e7)
```

```
> # A tibble: 3 x 4
>   State      Cases Deaths Population
>   <fct>      <int>  <int>      <int>
> 1 Arizona      947     15    6500180
> 2 New Jersey  1414     15    8682661
> 3 Utah         988     16    2736424
```

Line filtering

```
h1n1 %>%
  filter(Deaths > 10 & Population <= 1e7)
```

```
> # A tibble: 3 x 4
>   State      Cases Deaths Population
>   <fct>      <int>  <int>      <int>
> 1 Arizona      947     15    6500180
> 2 New Jersey  1414     15    8682661
> 3 Utah         988     16    2736424
```

```
h1n1 %>%
  filter(Deaths > 10 | Cases >= 1000)
```

```
> # A tibble: 13 x 4
>   State       Cases Deaths Population
>   <fct>       <int>  <int>      <int>
> 1 Arizona       947     15    6500180
> 2 California   3161     52   36756666
> 3 Connecticut  1713      8    3501252
> 4 Florida      2915     23   18328340
> 5 Hawaii       1424      3    1288198
> # … with 8 more rows
```

Line filtering

```
h1n1 %>%
  filter(State %in% c("New York", "North Carolina"))
```

```
> # A tibble: 2 x 4
>   State          Cases Deaths Population
>   <fct>          <int>  <int>      <int>
> 1 New York        2738     63   19490297
> 2 North Carolina   483      5    9222414
```

- String manipulation using the **stringr** package (also tidyverse)

```
h1n1 %>%
  filter(str_detect(State, "A"))
```

```
> # A tibble: 4 x 4
>   State    Cases Deaths Population
>   <fct>    <int>  <int>      <int>
> 1 Alabama    477      0    4661900
> 2 Alaska     272      0     686293
> 3 Arizona    947     15    6500180
> 4 Arkansas   131      0    2855390
```

Selecting columns

- Infertility after Spontaneous and Induced Abortion (case-control)

```
library(datasets)
infert <- as_tibble(infert)
infert
```

```
> # A tibble: 248 x 8
>    education   age parity induced  case spontaneous stratum
>    <fct>     <dbl>  <dbl>   <dbl> <dbl>       <dbl>   <int>
> 1 0-5yrs       26      6       1     1           2       1
> 2 0-5yrs       42      1       1     1           0       2
> 3 0-5yrs       39      6       2     1           0       3
> 4 0-5yrs       34      4       2     1           0       4
> 5 6-11yrs      35      3       1     1           1       5
> # … with 243 more rows, and 1 more variable:
> #   pooled.stratum <dbl>
```

Selecting columns

```
infert %>% select(age, case)
```

```
> # A tibble: 248 x 2
>     age  case
>   <dbl> <dbl>
> 1    26     1
> 2    42     1
> 3    39     1
> 4    34     1
> 5    35     1
> # … with 243 more rows
```

```
infert %>% select(education:case)
```

```
> # A tibble: 248 x 5
>   education   age parity induced  case
>   <fct>     <dbl>  <dbl>   <dbl> <dbl>
> 1 0-5yrs       26      6       1     1
> 2 0-5yrs       42      1       1     1
> 3 0-5yrs       39      6       2     1
> 4 0-5yrs       34      4       2     1
> 5 6-11yrs      35      3       1     1
> # … with 243 more rows
```

Selecting columns

```
infert %>% select(contains("stratum"))
```

```
> # A tibble: 248 x 2
>   stratum pooled.stratum
>     <int>          <dbl>
> 1       1              3
> 2       2              1
> 3       3              4
> 4       4              2
> 5       5             32
> # … with 243 more rows
```

```
infert %>% select(-education, -age)
```

```
> # A tibble: 248 x 6
>   parity induced  case spontaneous stratum pooled.stratum
>    <dbl>   <dbl> <dbl>       <dbl>   <int>          <dbl>
> 1      6       1     1           2       1              3
> 2      1       1     1           0       2              1
> 3      6       2     1           0       3              4
> 4      4       2     1           0       4              2
> 5      3       1     1           1       5             32
> # … with 243 more rows
```

Sorting dataset

```
infert %>% arrange(age)
```

```
> # A tibble: 248 x 8
>    education   age parity induced  case spontaneous stratum
>    <fct>     <dbl>  <dbl>   <dbl> <dbl>       <dbl>   <int>
> 1 6-11yrs      21      1       0     1           1       9
> 2 12+ yrs      21      1       0     1           1      67
> 3 6-11yrs      21      1       0     0           1       9
> 4 12+ yrs      21      1       0     0           1      67
> 5 6-11yrs      21      1       1     0           0       9
> # … with 243 more rows, and 1 more variable:
> #   pooled.stratum <dbl>
```

```
infert %>% arrange(spontaneous, desc(age))
```

```
> # A tibble: 248 x 8
>    education   age parity induced  case spontaneous stratum
>    <fct>     <dbl>  <dbl>   <dbl> <dbl>       <dbl>   <int>
> 1 6-11yrs      44      1       0     0           0      20
> 2 6-11yrs      44      1       1     0           0      20
> 3 0-5yrs       42      1       1     1           0       2
> 4 6-11yrs      42      1       1     1           0      33
> 5 0-5yrs       42      1       0     0           0       2
> # … with 243 more rows, and 1 more variable:
```

Creating new variables

```
h1n1 <- h1n1 %>%
  mutate(Population = round(Population/1000, 0))
print(h1n1, n = 10)
```

```
> # A tibble: 51 x 4
>    State                Cases Deaths Population
>    <fct>                <int>  <int>      <dbl>
>  1 Alabama                477      0       4662
>  2 Alaska                 272      0        686
>  3 Arizona                947     15       6500
>  4 Arkansas               131      0       2855
>  5 California            3161     52      36757
>  6 Colorado               171      0       4939
>  7 Connecticut           1713      8       3501
>  8 Delaware               381      0        873
>  9 District of Columbia    45      0        592
> 10 Florida               2915     23      18328
> # … with 41 more rows
```

Creating new variables

```
h1n1 %>%
  mutate(Rate = Deaths/Cases) %>%
  print(n = 10)
```

```
> # A tibble: 51 x 5
>    State                Cases Deaths Population    Rate
>    <fct>                <int>  <int>      <dbl>   <dbl>
>  1 Alabama                477      0       4662 0
>  2 Alaska                 272      0        686 0
>  3 Arizona                947     15       6500 0.0158
>  4 Arkansas               131      0       2855 0
>  5 California            3161     52      36757 0.0165
>  6 Colorado               171      0       4939 0
>  7 Connecticut           1713      8       3501 0.00467
>  8 Delaware               381      0        873 0
>  9 District of Columbia    45      0        592 0
> 10 Florida               2915     23      18328 0.00789
> # … with 41 more rows
```

Creating new variables

```
h1n1 %>%
  mutate(aux = paste0(round(Deaths/Cases*100, 1), "%"),
         Rate = ifelse(Deaths == 0, "No deaths", aux)) %>%
  print(n = 10)
```

```
> # A tibble: 51 x 6
>    State                Cases Deaths Population aux   Rate
>    <fct>                <int>  <int>      <dbl> <chr> <chr>
>  1 Alabama                477      0       4662 0%    No deat…
>  2 Alaska                 272      0        686 0%    No deat…
>  3 Arizona                947     15       6500 1.6%  1.6%
>  4 Arkansas               131      0       2855 0%    No deat…
>  5 California            3161     52      36757 1.6%  1.6%
>  6 Colorado               171      0       4939 0%    No deat…
>  7 Connecticut           1713      8       3501 0.5%  0.5%
>  8 Delaware               381      0        873 0%    No deat…
>  9 District of Colum…      45      0        592 0%    No deat…
> 10 Florida               2915     23      18328 0.8%  0.8%
> # … with 41 more rows
```

Creating new variables

```
h1n1 <- h1n1 %>%
  mutate(id = row_number()) %>%
  unite(col = "state_id", id, State, sep = "_")
h1n1 %>% print(n = 10)
```

```
> # A tibble: 51 x 4
>    state_id                Cases Deaths Population
>    <chr>                   <int>  <int>      <dbl>
>  1 1_Alabama                 477      0       4662
>  2 2_Alaska                  272      0        686
>  3 3_Arizona                 947     15       6500
>  4 4_Arkansas                131      0       2855
>  5 5_California             3161     52      36757
>  6 6_Colorado                171      0       4939
>  7 7_Connecticut            1713      8       3501
>  8 8_Delaware                381      0        873
>  9 9_District of Columbia     45      0        592
> 10 10_Florida               2915     23      18328
> # … with 41 more rows
```

Creating new variables

```
h1n1 <- h1n1 %>%
  separate(col = state_id, into = c("id", "State"), sep = "_")
h1n1 %>% print(n = 10)
```

```
> # A tibble: 51 x 5
>    id    State                Cases Deaths Population
>    <chr> <chr>                <int>  <int>      <dbl>
>  1 1     Alabama                477      0       4662
>  2 2     Alaska                 272      0        686
>  3 3     Arizona                947     15       6500
>  4 4     Arkansas               131      0       2855
>  5 5     California            3161     52      36757
>  6 6     Colorado               171      0       4939
>  7 7     Connecticut           1713      8       3501
>  8 8     Delaware               381      0        873
>  9 9     District of Columbia    45      0        592
> 10 10    Florida               2915     23      18328
> # … with 41 more rows
```

Grouping and summarizing

```
infert %>%
  summarise(mean_age = mean(age, na.rm = TRUE))
```

```
> # A tibble: 1 x 1
>   mean_age
>      <dbl>
> 1     31.5
```

```
infert %>%
  summarise(mean_age = mean(age, na.rm = TRUE),
            median_age = median(age, na.rm = TRUE),
            n = n(),
            nmiss = sum(is.na(age)))
```

```
> # A tibble: 1 x 4
>   mean_age median_age     n nmiss
>      <dbl>      <dbl> <int> <int>
> 1     31.5         31   248     0
```

Grouping and summarizing

```
infert %>%
  group_by(case) %>%
  summarise(mean_age = mean(age, na.rm = TRUE),
            median_age = median(age, na.rm = TRUE),
            n = n(),
            nmiss = sum(is.na(age)))
```

```
> # A tibble: 2 x 5
>    case mean_age median_age     n nmiss
>   <dbl>    <dbl>      <dbl> <int> <int>
> 1     0     31.5         31   165     0
> 2     1     31.5         31    83     0
```

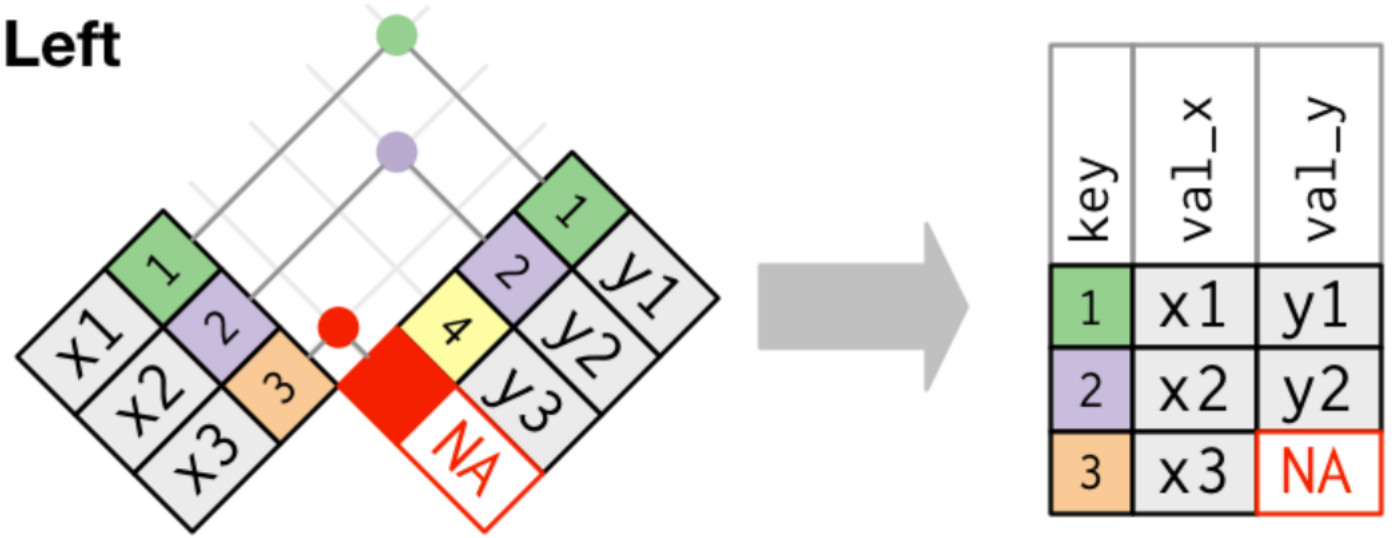
Grouping and summarizing

```
infert %>%
  group_by(case, induced) %>%
  summarise(mean_age = mean(age, na.rm = TRUE),
            median_age = median(age, na.rm = TRUE),
            n = n(),
            nmiss = sum(is.na(age))) %>%
  print(n = 10)
```

```
> # A tibble: 6 x 6
> # Groups:   case [2]
>    case induced mean_age median_age     n nmiss
>   <dbl>   <dbl>    <dbl>      <dbl> <int> <int>
> 1     0       0     32.2         32    96     0
> 2     0       1     30.5         29    45     0
> 3     0       2     30.5         28    24     0
> 4     1       0     31.6         31    47     0
> 5     1       1     31.4         31    23     0
> 6     1       2     31.5         30    13     0
```

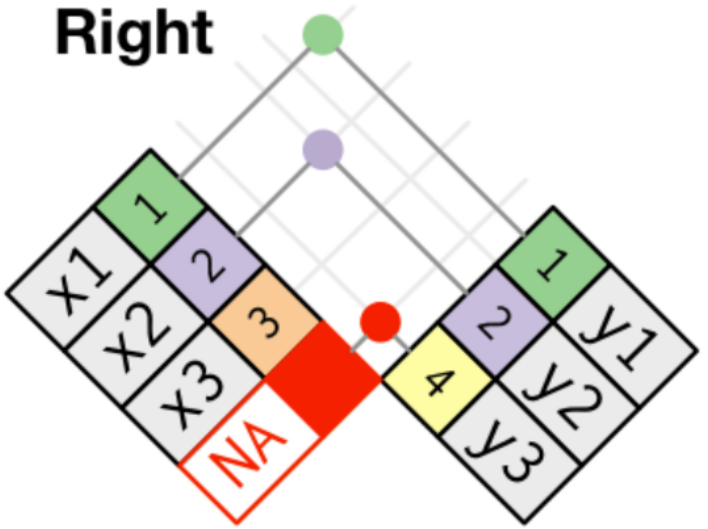
Merging datasets

- Family of `_join()` functions
  - `left_join(x, y)`: returns ALL lines from `x` and ALL columns of `x` and `y`. Lines on `x` that does not have a correspondence in `y` will receive `NA` on the new dataset



Merging datasets

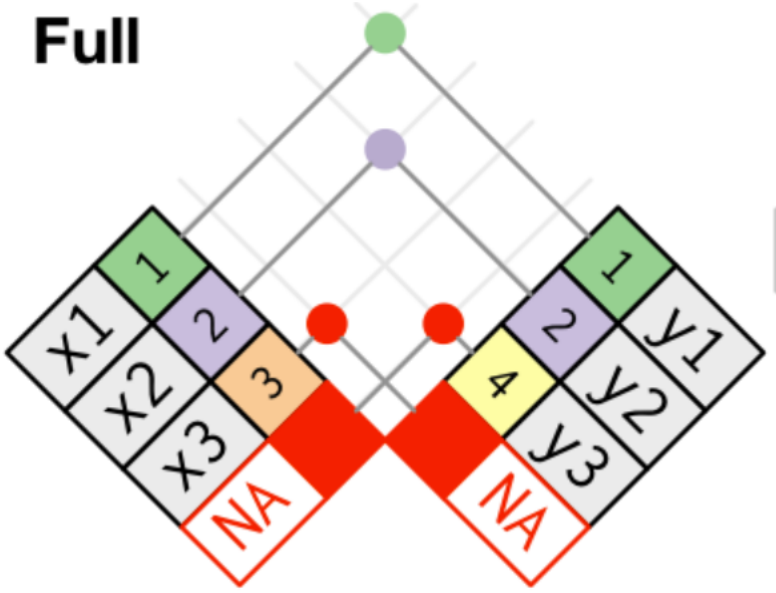
- `right_join(x, y)`: returns ALL lines from `y` and ALL columns of `y` and `x`. Lines on `y` that does not have a correspondence in `x` will receive `NA` on the new dataset



key	val_x	val_y
1	x1	y1
2	x2	y2
4	NA	y3

Merging datasets

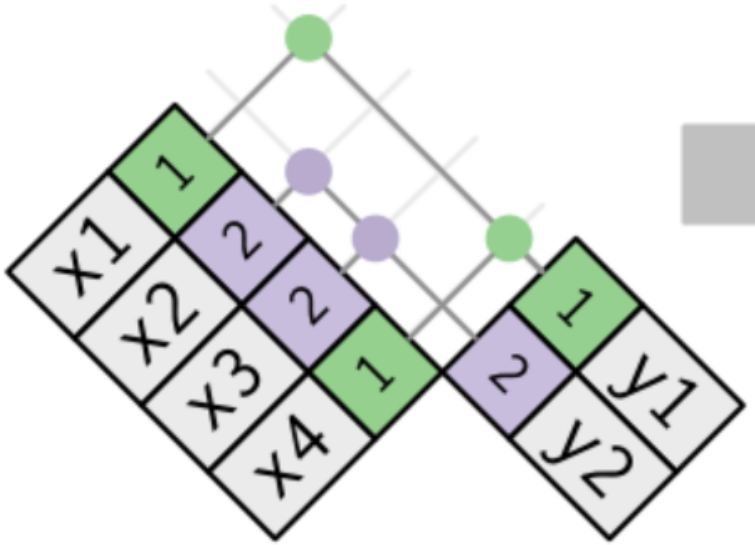
- `full_join(x, y)`: returns ALL lines and ALL columns of `y` and `x`. Lines without correspondence will receive `NA` on the new dataset



key	val_x	val_y
1	x1	y1
2	x2	y2
3	x3	NA
4	NA	y3

Merging datasets

- `by = <key_variable>`: used to match cases
- In case we have duplicated keys, this will happen



val_x	key	val_y
x1	1	y1
x2	2	y2
x3	2	y2
x4	1	y1

- Same if there are duplicated on the "look-up" table

Merging datasets

```
x <- data.frame(id=c(1, 2, 3), vx=c("x1", "x2", "x3"))
y <- data.frame(id=c(1, 2), vy=c("y1", "y2"))
x; y
```

```
>   id vx
> 1  1 x1
> 2  2 x2
> 3  3 x3
```

```
>   id vy
> 1  1 y1
> 2  2 y2
```

```
left_join(x, y, by = "id")
```

```
>   id vx   vy
> 1  1 x1   y1
> 2  2 x2   y2
> 3  3 x3 <NA>
```

Merging datasets

```
x <- data.frame(id=c(1, 2, 2, 1), vx=c("x1", "x2", "x3", "x4"))
y <- data.frame(id=c(1, 2, 2), vy=c("y1", "y2", "y2"))
x; y
```

```
>   id vx
> 1  1 x1
> 2  2 x2
> 3  2 x3
> 4  1 x4
```

```
>   id vy
> 1  1 y1
> 2  2 y2
> 3  2 y2
```

```
left_join(x, y, by = "id")
```

```
>   id vx vy
> 1  1 x1 y1
> 2  2 x2 y2
> 3  2 x2 y2
> 4  2 x3 y2
> 5  2 x3 y2
> 6  1 x4 y1
```

Reshaping data wide/long

- Usually necessary when measuring individuals multiple times or over time on the same variable
- **`wide`** - one line per person

id	trt	work.T1	home.T1	work.T2	home.T2
1	treatment	0.08513597	0.6158293	0.1135090	0.05190332
2	control	0.22543662	0.4296715	0.5959253	0.26417767
3	treatment	0.27453052	0.6516557	0.3580500	0.39879073
4	control	0.27230507	0.5677378	0.4288094	0.83613414

- **`long`** - multiple lines per person

id	trt	key	time
1	treatment	work.T1	0.08513597
2	control	work.T1	0.22543662
3	treatment	work.T1	0.27453052
4	control	work.T1	0.27230507
1	treatment	home.T1	0.61582931
2	control	home.T1	0.42967153
3	treatment	home.T1	0.65165567
4	control	home.T1	0.56773775
1	treatment	work.T2	0.11350898
2	control	work.T2	0.59592531
3	treatment	work.T2	0.35804998
4	control	work.T2	0.42880942
1	treatment	home.T2	0.05190332
2	control	home.T2	0.26417767
3	treatment	home.T2	0.39879073
4	control	home.T2	0.83613414

Reshaping data wide/long

Wide to long: gather()

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

```
cases %>% gather(Year, n, 2:4)
```

dataframe to reshape

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

Reshaping data wide/long

Wide to long: gather()

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

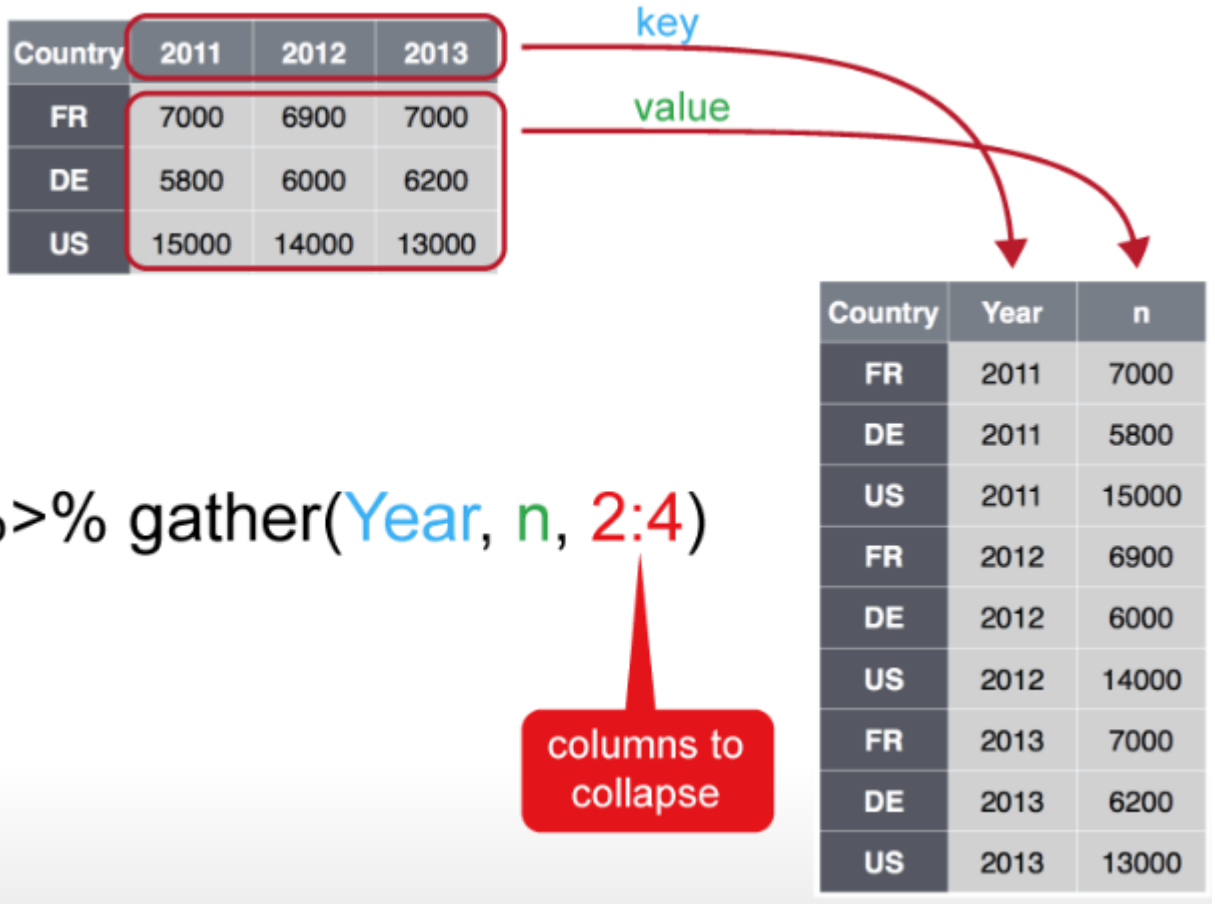
```
cases %>% gather(Year, n, 2:4)
```

name of the new "value" column

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

Reshaping data wide/long

Wide to long: gather()



Reshaping data wide/long

Wide to long: spread()

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

```
cases %>% spread(Year, n)
```

dataframe to reshape

Reshaping data wide/long

Wide to long: spread()

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

```
cases %>% spread(Year, n)
```

column to use as keys
(new column names)

Reshaping data wide/long

Wide to long: spread()