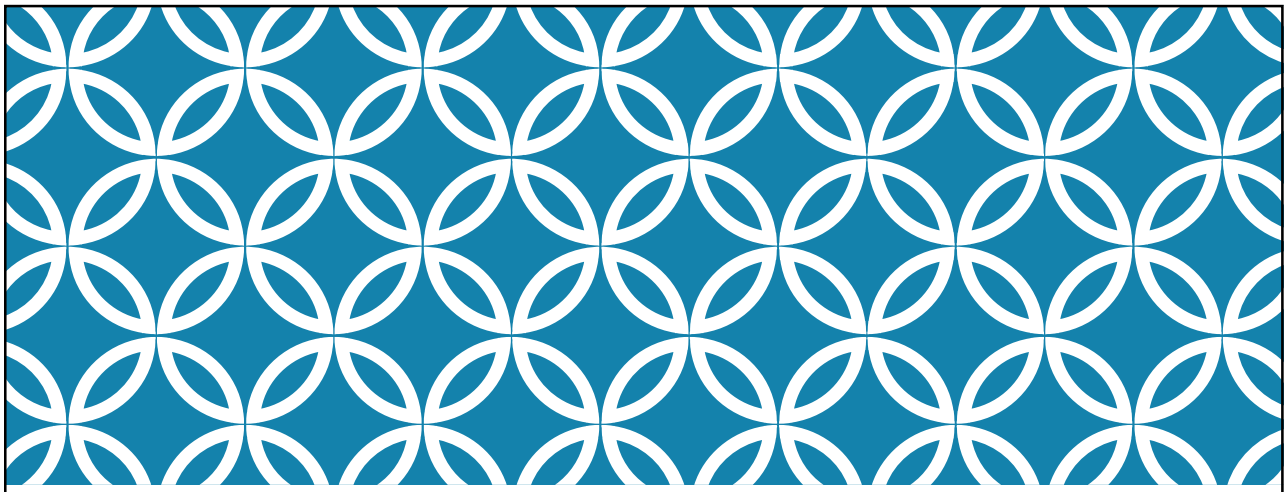


Data Transformation

Session 4

Amrom Obstfeld

1

1

Goal

1. Learn how to use dplyr to transform data frames
2. Appreciate the role of piping in facilitating data transformation

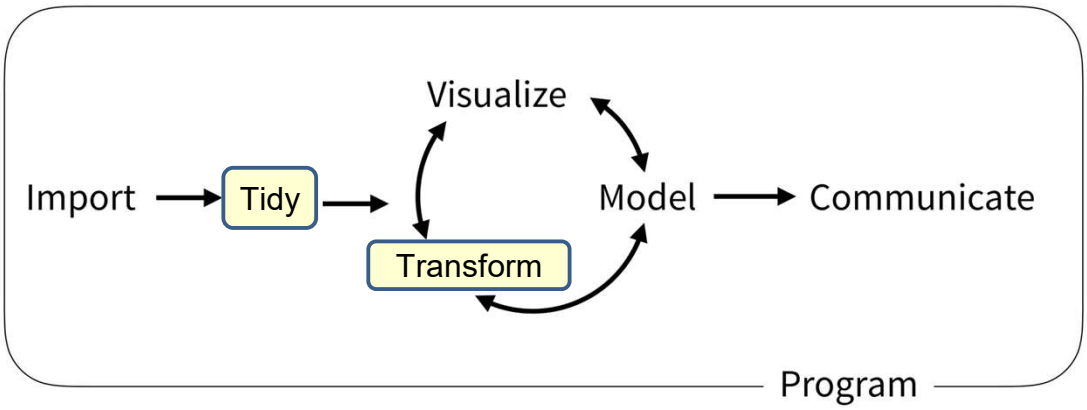
Objectives

1. List the major forms of data transformation implemented in dplyr
2. Use code templates with dplyr functions to tidy a raw data set
3. Use the pipe operator to pass the output of one function as an input to the next function
4. Create new calculated columns not found in the original data frame

2

2

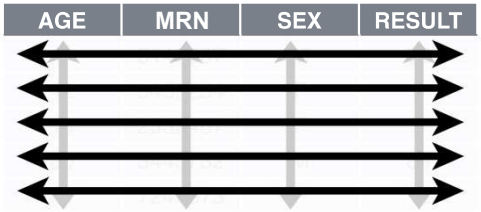
Typical Data Science Pipeline



3

3

What is a "Tidy" Data Frame



A data set is **tidy** if:

1. Each **variable** is in its own **column**
2. Each **observation** is in its own **row**
3. Each **value** is in its own **cell**

4

4

Your Turn 1

Open "**04-Transform.Rmd**"

Run the setup chunk

```
```{r setup}
library(tidyverse)  # Provides functions used throughout this session

covid_testing <- read_csv("covid_testing.csv")
```
```

5

5

Pop Quiz

How can you confirm that you have successfully loaded the data file into RStudio?

1. The code that imported the data did not yield an error
2. Code that references the `covid_testing` object runs without errors
3. The `covid_testing` object is present in the environment pane
4. All of the above

6

6

Transform Data with

7

7

dplyr

dplyr implements a *grammar* for transforming tabular data.

8

8

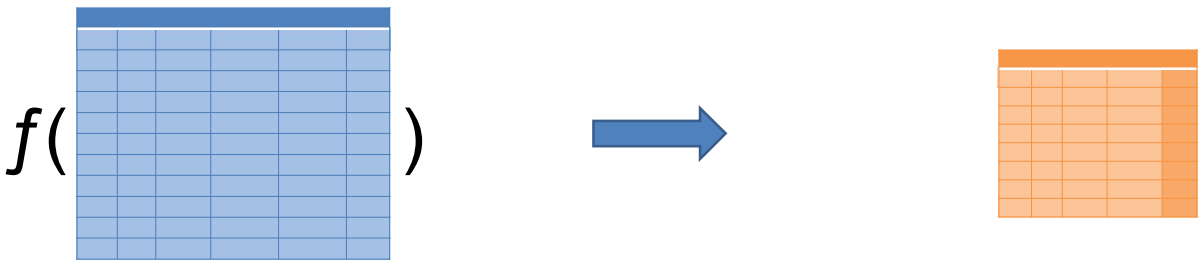
dplyr: a grammar for transforming data

1. **Choose** columns. — **select()**
2. **Extract** rows. — **filter()**
3. **Derive** new columns. — **mutate()**
4. **Change** the unit of analysis. — **summarize()**

9

9

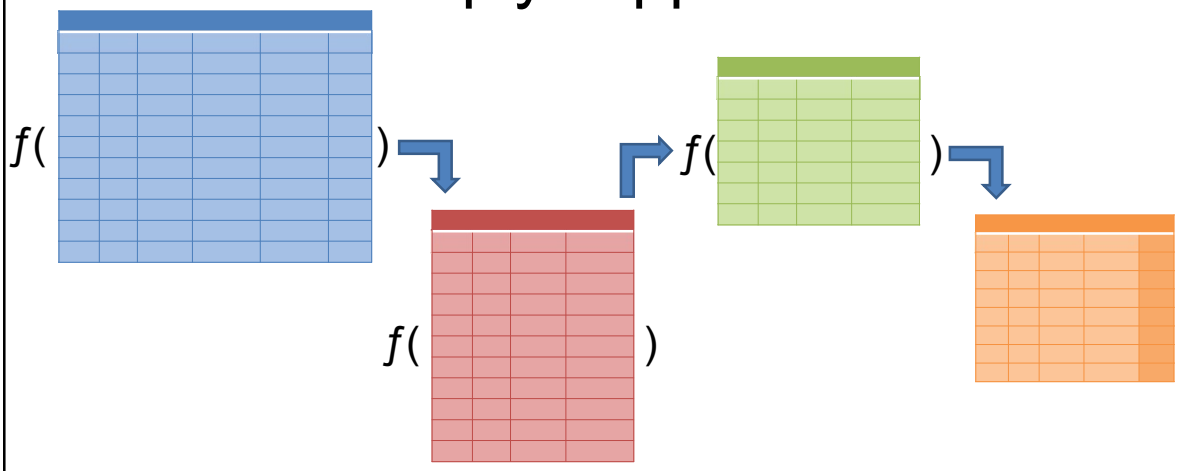
Dplyr Approach



10

10

Dplyr Approach



11

Common syntax

Each function takes a data frame as its first argument and returns a data frame as its output.

```
function(data, ...)
```

dplyr function

data frame to transform

specific arguments

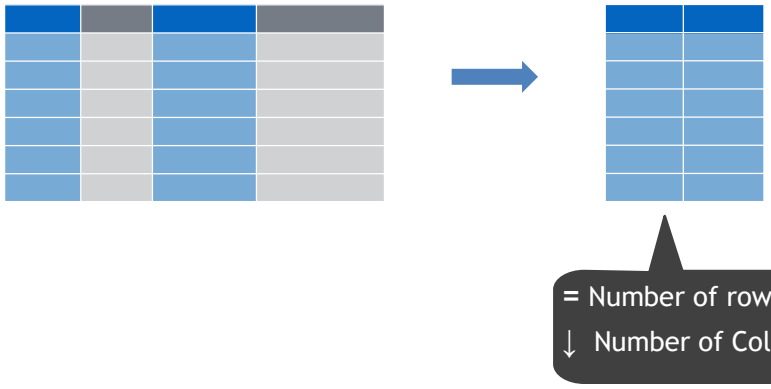
12

13

14

select()

Extract columns from a data frame



15

15

select()

Extract columns from a data frame

```
select(covid_testing, mrn, last_name)
```

dplyr function

data frame to transform

name(s) of columns to extract
(or a select helper)

16

16

select()

Extract columns from a data frame **by name**

```
select(covid_testing, mrn, last_name)
```

covid_testing

mrn	first_name	last_name	gender
5000876	sarella	stark	female
5006017	alester	stark	male
5001412	jhezane	westerling	female
5000533	penny	targaryen	female

…

→

mrn	last_name
5000876	stark
5006017	stark
5001412	westerling
5000533	targaryen

17

17

select()

Extract columns from a data frame **by name**

```
select(covid_testing, -mrn, -last_name)
```

covid_testing

mrn	first_name	last_name	gender
5000876	sarella	stark	female
5006017	alester	stark	male
5001412	jhezane	westerling	female
5000533	penny	targaryen	female

…

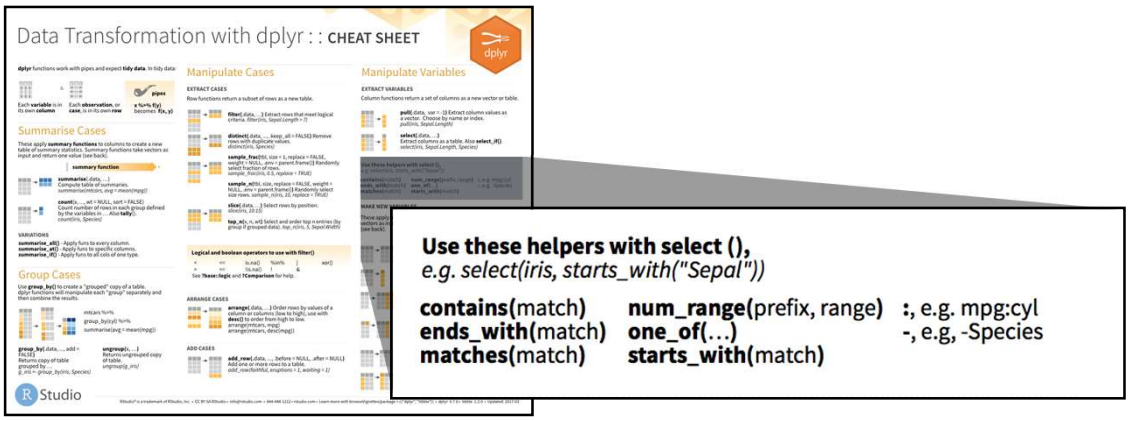
→

first_name	gender
sarella	female
alester	male
jhezane	female
penny	female

18

18

select() helpers



19

Your Turn 2

- Alter the code to select just the `first_name` column from `covid_testing`
- Use the second code chunk to see if you can remove the `first_name` column

```
covid_testing_2 <- select(covid_testing, _____)
```

20

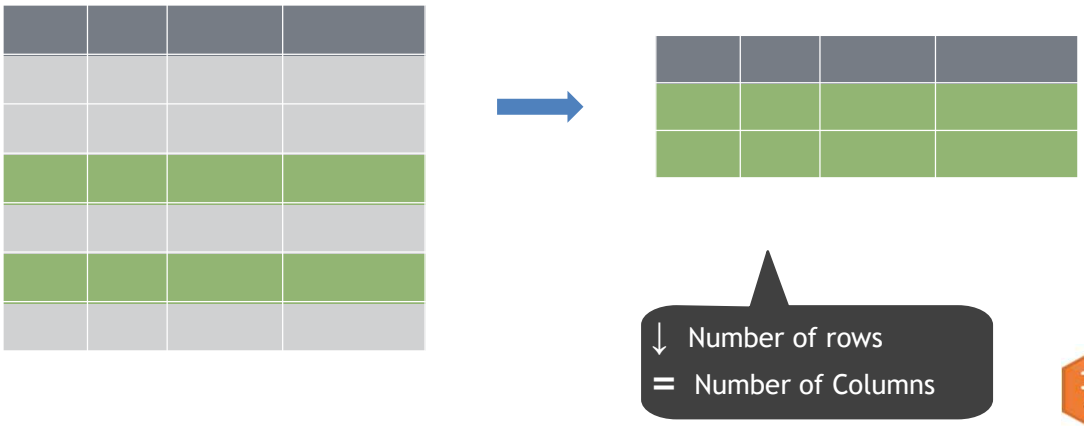
20

filter()

21

filter()

Extract rows that meet logical criteria



22

22

Common syntax

Each function takes a data frame as its first argument and returns a data frame as its output.

```
function(data, ...)
```

dplyr function

data frame to transform

specific arguments

23

23

filter()

Extract rows that meet logical criteria

```
filter(data,... )
```

data frame to transform

one or more logical tests (filter returns each row for which the test is TRUE)

24

24

filter()

Extract rows that meet logical criteria

```
filter(data, column_name == criteria )
```

one or more logical tests
(filter returns each row for which the test is TRUE)

25

25

filter()

Extract rows that meet logical criteria

```
filter(covid_testing, mrn==5000083)
```

	mrn	first_name	last_name
FALSE	5000876	sarella	stark
FALSE	5006017	alester	stark
FALSE	5001412	jhezane	westerling
TRUE	5000083	lollys	clegane

→

mrn	first_name	last_name
5000083	lollys	clegane

26

26

filter()

Extract rows that meet logical criteria

```
filter(covid_testing, mrn==5000083)
```

= sets
(returns nothing)
== tests if equal
(returns TRUE or FALSE)

mrn	first_name	last_name
5000876	sarella	stark
5006017	alester	stark
5001412	jhezane	westerling
5000083	lollys	clegane

27

27

filter()

Values coded as character strings must be surrounded by quotes

Extract rows that meet logical criteria.

```
filter(covid_testing, last_name=="stark")
```

mrn	first_name	last_name	
5000876	sarella	stark	TRUE
5006017	alester	stark	TRUE
5001412	jhezane	westerling	FALSE
5000083	lollys	clegane	FALSE

→

mrn	first_name	last_name
5000876	sarella	stark
5006017	alester	stark

28

28

filter()

Extract rows that meet logical criteria

```
filter(data, ...)
```

data frame to transform

one or more logical tests (filter returns each row for which the test is TRUE)

29

29

Logical tests

`x < y`	Less than
`x > y`	Greater than
`x == y`	Equal to
`x <= y`	Less than or equal to
`x >= y`	Greater than or equal to
`x != y`	Not equal to
`x %in% y`	Group membership
`is.na(x)`	Is NA
`!is.na(x)`	Is not NA

30

30

Pop Quiz

What is the result?

```
1 == 1
```

31

31

Pop Quiz

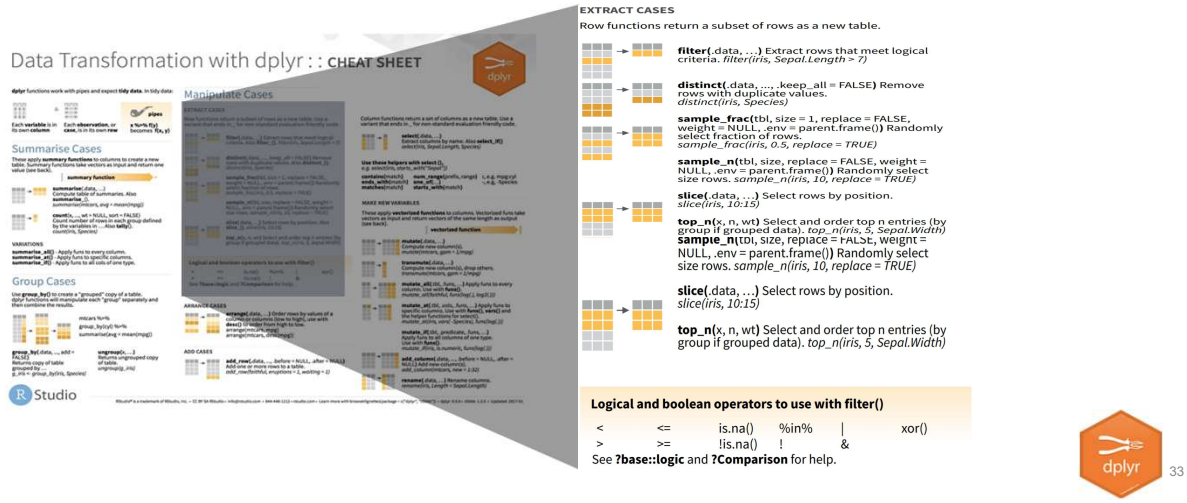
What is the result?

```
3 != 1
```

32

32

filter() variants



33

Your Turn 3

Use filter() with the logical operators to find:

- Every test for patients **over age 80**
- All of the covid testing where the demographic group (demo_group) is **equal to "client"**

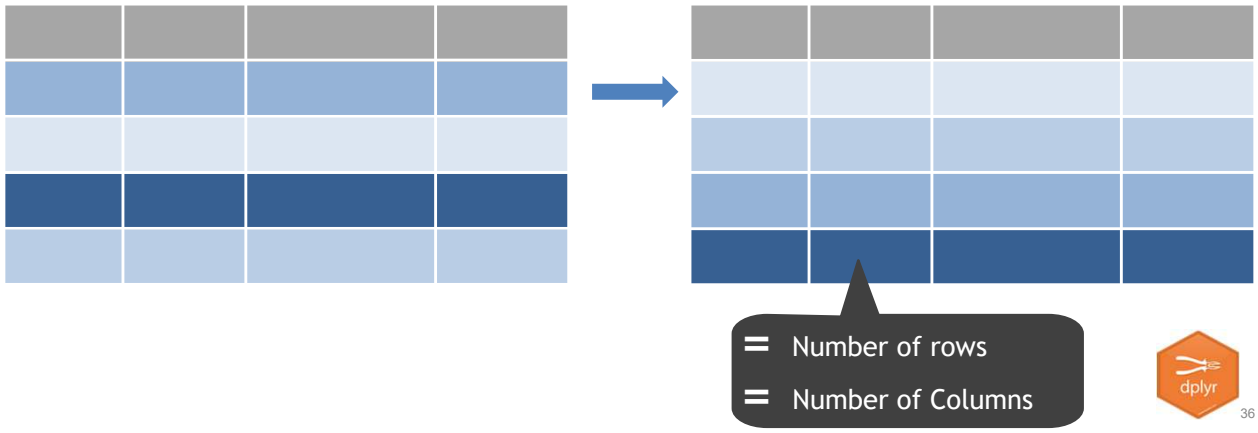
34

arrange()

35

arrange()

Order rows by values in a column



36

arrange()

Order rows by values in a column

```
arrange(data,... )
```

data frame to transform

name(s) of columns to arrange by

37

37

arrange()

Order rows by values in a column

```
arrange(covid_testing, first_name)
```

mrn	first_name	last_name
5000876	sarella	stark
5006017	alester	stark
5001412	jhezane	targaryen
5000533	penny	targaryen

→

mrn	first_name	last_name
5006017	alester	stark
5001412	jhezane	targaryen
5000533	penny	targaryen
5000876	sarella	stark

38

38

arrange()

Order rows by values in a column

```
arrange(covid_testing, desc(mrn))
```

mrn	first_name	last_name
5000876	sarella	stark
5006017	alester	stark
5001412	jhezane	targaryen
5000533	penny	targaryen

→

mrn	first_name	last_name
5006017	alester	stark
5001412	jhezane	targaryen
5000876	sarella	stark
5000533	penny	targaryen

39

39

Your Turn 4

The column `ct_value` contains the cycle threshold (Ct) for the real-time PCR that generated the final result.

How might you use `arrange()` to determine the highest and lowest Ct result in the dataset?

40

Pop Quiz

The default behavior of arrange() is to order from lower to higher values.

When might arrange() place "1000" before "50"?

41

%>%

42

Data Analysis Steps

```
day_10 <- filter(covid_testing, pan_day <= 10)

day_10 <- select(day_10, clinic_name)

day_10 <- arrange(day_10 , clinic_name)
```

1. Filter tests to those on pandemic day less than 10
2. Select the column that contains ordering location
3. Arrange those columns by location

43

43

Data Analysis Steps

```
day_10 <- arrange(
            select(
              filter(
                covid_testing,
                pan_day <= 10
              ),
              clinic_name
            ),
            clinic_name
          )
```

44

44

The Pipe Operator %>%

Passes result on left into first argument of function on right.

```
covid_testing %>% filter(____, pan_day <= 10)
```

```
filter(covid_tesing, pan_day <= 10)
covid_tesing %>% filter(pan_day <= 10)
```

45

45

Data Analysis Steps

```
day_10 <- arrange(
            select(
              filter(
                covid_testing,
                pan_day <= 10
              ),
              clinic_name
            ),
            clinic_name
          )
```

46

46

Data Analysis Steps

```
covid_testing %>%
   filter(pan_day <= 10) %>%
   select(clinic_name) %>%
   arrange(clinic_name)
```

47

47

Shortcut to type %>%

Cmd + Shift + M (Mac)

Ctrl + Shift + M (Windows)

48

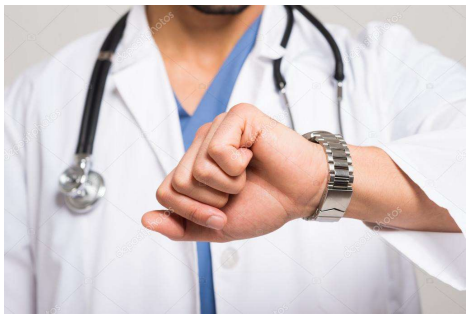
48

Scene

The PICU would like a word with you because of a recent incident involving a delay in results for a patient who required a AGP

They had to wait over 10 hours before the procedure could begin

You decide to investigate… WITH DATA



49

49

Your Turn 5

Use %>% to write a sequence of three functions that:

1. Filters to tests from the clinic (**`clinic_name`**) of "picu"
2. Selects the column with the receive to verify turnaround time (**`rec_ver_tat`**) as well as the day from start of the pandemic (**`pan_day`**)
3. Arrange the `` **`pan_day`** `` from highest to lowest

Using <-, assign the result to a new variable, call it whatever you want.

50

Isolating data

Extract variables with **select()**

Extract rows with **filter()**

Arrange rows, with **arrange()**.

51

51

Deriving Data

52

What is the mean and median collect to verify turnaround time by clinic?

53

53

Breaking down the analytical question

1. Total TAT for each test
2. Group tests by clinic
3. Calculate mean and median for each clinic

54

54

Deriving data

Make new variables with **mutate()**

Make summaries of data with **summarize()**

55

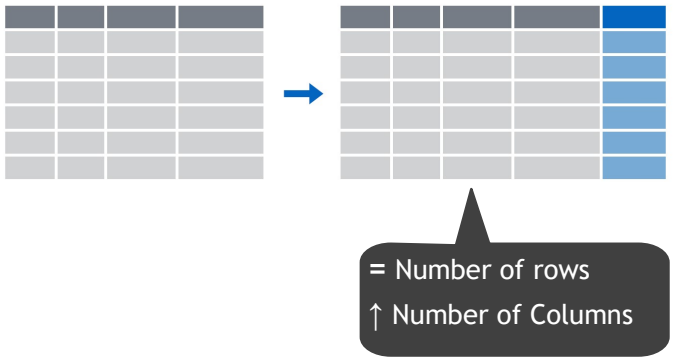
55

mutate()

56

mutate()

Creating new calculated columns



57

57

mutate()

Creating new calculated columns

```
Covid_testing %>%
    mutate(new_column = calculation)
```

name for new column

equals

function whose results will populate columns

58

58

mutate()

Creating new calculated columns

```
covid_testing %>%
    mutate(c_r_tat_mins = col_rec_tat * 60)
```

mrn	col_rec_tat	rec_ver_tat
5000876	29.5	11.5
5006017	3.6	5
5001412	1.4	5.2
5000533	2.3	5.8

→

mrn	col_rec_tat	rec_ver_tat	c_r_tat_mins
5000876	29.5	11.5	1770
5006017	3.6	5	216
5001412	1.4	5.2	84
5000533	2.3	5.8	138

59

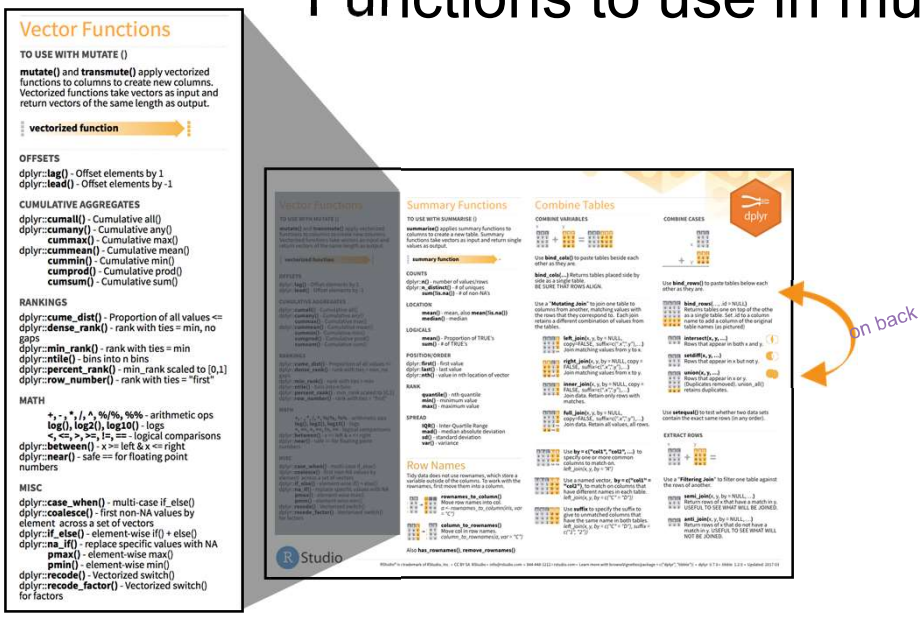
59

Your Turn 6

Create a new column using the mutate() function that contains the total TAT (sum of **`col_rec_tat`** and **`rec_ver_tat`**)

60

Functions to use in mutate()



61

61

Goal

1. Learn how to use dplyr to transform data frames
2. Appreciate the role of piping in facilitating data transformation

Objectives

1. List the major forms of data transformation implemented in dplyr
2. Use code templates containing dplyr functions to tidy a raw data set
3. Use the pipe operator to pass the output of one function as an input to the next function
4. Create new calculated columns not found in the original data frame

62

62

Dplyr tips and tricks

63

select()

Renaming columns

```
covid_testing %>%
    select(MRN = mrn, first_name, last_name)
```

mrn <dbl>	first_name <chr>	last_name <chr>
5001412	jhezane	westerling
5000533	penny	targaryen
5009134	grunt	rivers
5008518	melisandre	swyft

MRN <dbl>	first_name <chr>	last_name <chr>
5001412	jhezane	westerling
5000533	penny	targaryen
5009134	grunt	rivers
5008518	melisandre	swyft

64

64

filter()

Filter to multiple matches

```
covid_testing %>%
    filter(first_name %in% c("jon","daenerys"))
```

mrn <dbl>	first_name <chr>
5001412	jhezane
5000533	penny
5009134	grunt
5008518	melisandre
5008967	rolley

mrn <dbl>	first_name <chr>
5002427	daenerys
5011120	jon
5001092	jon
5004082	jon
5005197	daenerys

65

65

mutate()

Replacing columns

Function to "coerce" one type of data into another type of data

```
covid_testing %>%
    mutate(mrn = as.character(mrn))
```

mrn <dbl>	first_name <chr>	last_name <chr>
5000876	sarella	stark
5006017	alester	stark
5001412	jhezane	westerling
5000533	penny	targaryen

mrn <chr>	first_name <chr>	last_name <chr>
5000876	sarella	stark
5006017	alester	stark
5001412	jhezane	westerling
5000533	penny	targaryen

66

66

mutate()

Conditionally replacing values

```
covid_testing %>%
     mutate(last_name = if_else(last_name=="targaryen",
                                "TARGARYEN",last_name))
```

mrn <dbl>	first_name <chr>	last_name <chr>
5001412	jhezane	westerling
5000533	penny	targaryen
5009134	grunt	rivers
5008518	melisandre	swyft

mrn <dbl>	first_name <chr>	last_name <chr>
5001412	jhezane	westerling
5000533	penny	TARGARYEN
5009134	grunt	rivers
5008518	melisandre	swyft

67

67