

Creating a Subset

Patrick Mathias
Lesson 4
DLMP Fall 2021

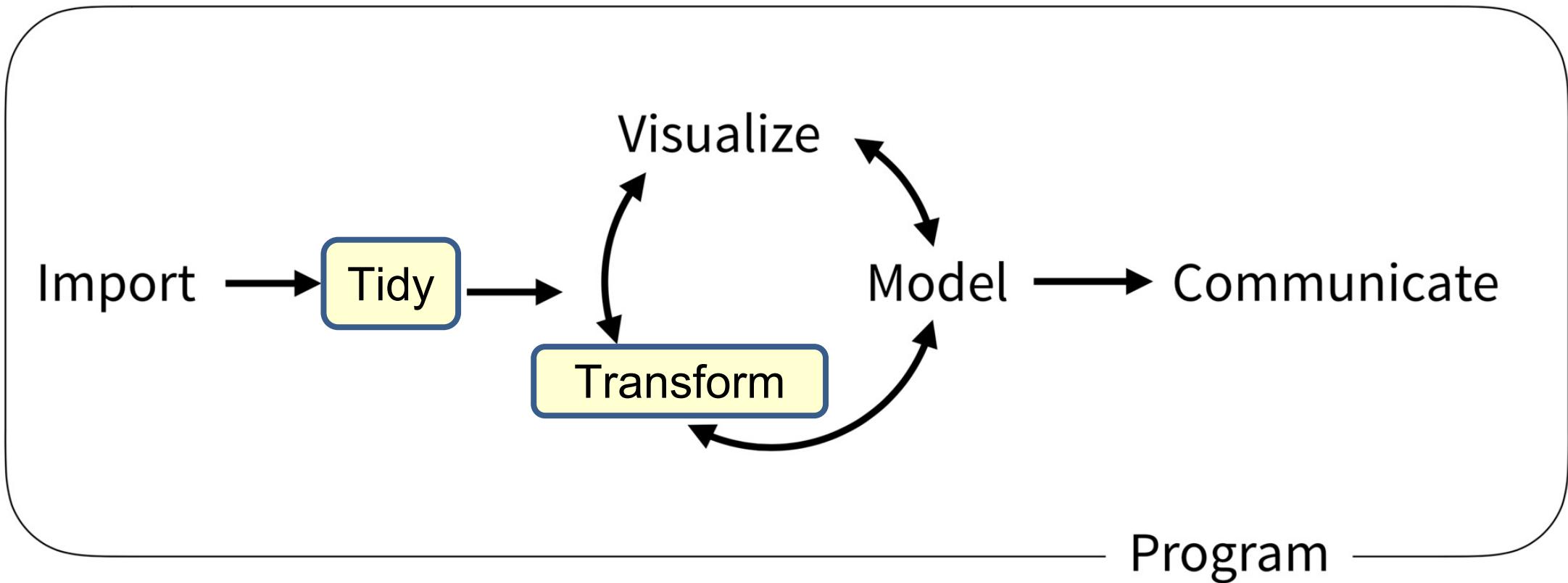
Goal

1. Learn how to use dplyr to transform data frames
2. Create a smaller subset from a data frame

Objectives

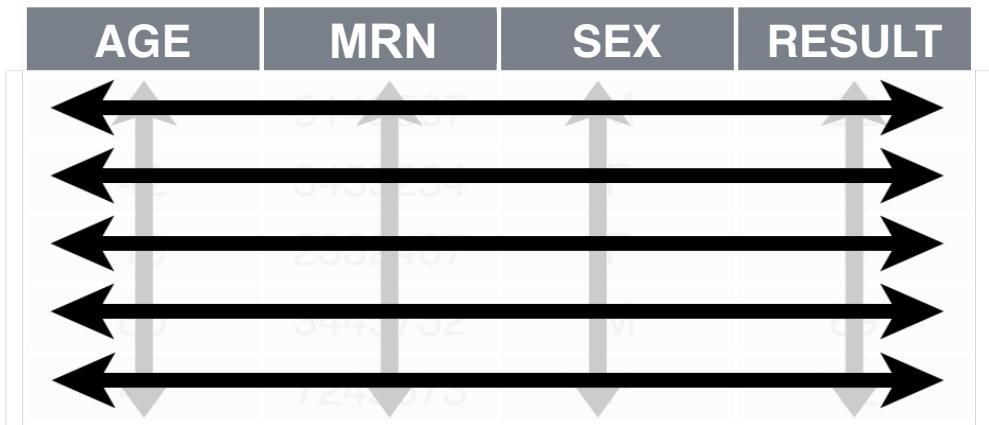
1. List the major forms of data transformation implemented in dplyr
2. Use code templates with dplyr functions to tidy a raw data set
3. Create a subset using logical conditions to extract specific rows from a data frame

Typical Data Science Pipeline



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**

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")
```
```



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

Transform Data with



dplyr



dplyr implements a *grammar* for transforming tabular data.

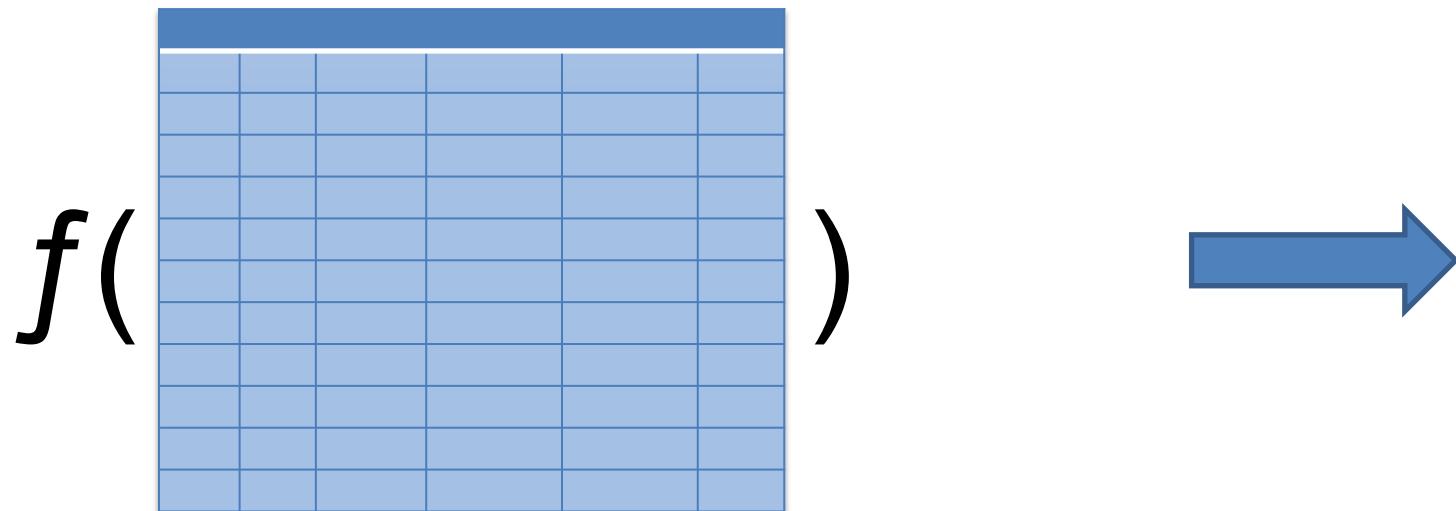


dplyr: a grammar for transforming data

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

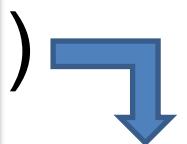
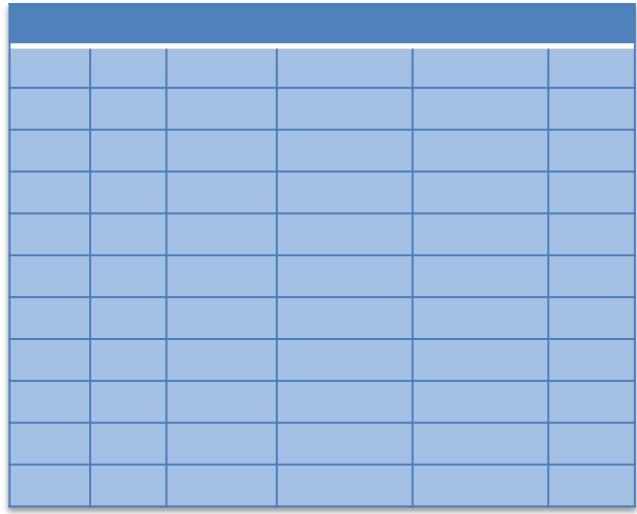


Dplyr Approach

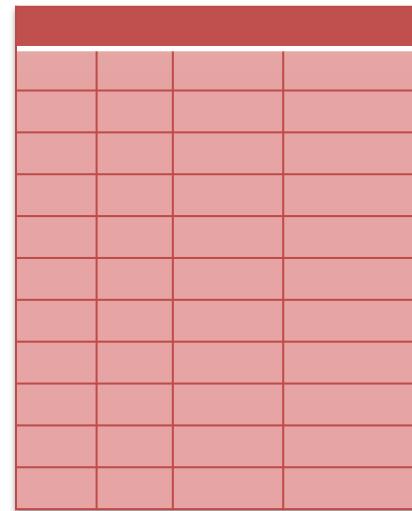


Dplyr Approach

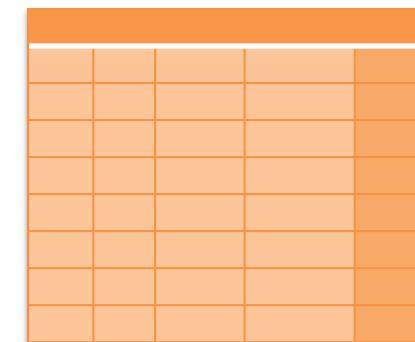
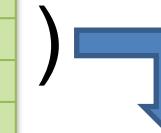
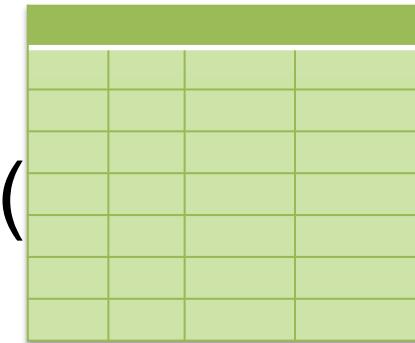
$f($



$f($



$f($



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



Subset Specific Columns

select()

Extract columns from a data frame

| | | | |
|------|------|------|------|
| Blue | Grey | Blue | Grey |
| Blue | Grey | Blue | Grey |
| Blue | Grey | Blue | Grey |
| Blue | Grey | Blue | Grey |
| Blue | Grey | Blue | Grey |



| | |
|------|--|
| Blue | |

= Number of rows
↓ Number of Columns



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)



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 |

...



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 |

...



select() helpers

Data Transformation with dplyr :: CHEAT SHEET

dplyr functions work with pipes and expect **tidy data**. In tidy data:

- Each variable is in its own column
- Each observation, or case, is in its own row
- x %>% f(y) becomes f(x, y)

Manipulate Cases

EXTRACT CASES
Row functions return a subset of rows as a new table.

- filter(data, ...)
- distinct(..., keep_all = FALSE)
- sample_frac(tbl, size = 1, replace = FALSE, weight = NULL, prob = parent.frame())
- sample_n(tbl, size, replace = FALSE, weight = NULL, env = parent.frame())
- slice(...)
- top_n(..., n, wt)

VARIATIONS

- summarise_all()
- summarise_att()
- summarise_if()

Group Cases

Use group_by() to create a "grouped" copy of a table. dplyr functions will manipulate each "group" separately and then combine the results.

- mtcars %>% group_by(cyl) %>% summarise(avg = mean(mpg))
- ungroup(x, ...)
- group_by(data, ..., add = FALSE)
- group_by_copy(...)
- g_iris <- group_by(iris, Species)

Manipulate Variables

EXTRACT VARIABLES
Column functions return a set of columns as a new vector or table.

- pull(data, var = -1)
- select(...)
- select_if(...)

MAKE NEW VARIABLES
These apply vectors as inputs to functions (see back).

- contains(match)
- ends_with(match)
- matches(match)

Logical and boolean operators to use with filter()

- <
- <=
- is.na()
- %in%
- |
- xor()
- &

See ?base::logical and ?Comparison for help.

ARRANGE CASES

- arrange(data, ...)
- arrange(mtcars, mpg)
- arrange(mtcars, desc(mpg))

ADD CASES

- add_row(data, ..., before = NULL, after = NULL)
- add_rows(faithful, eruptions = 1, waiting = 1)

Use these helpers with select (), e.g. select(iris, starts_with("Sepal"))

contains(match) **num_range(prefix, range)** **:**, e.g. mpg:cyl
ends_with(match) **one_of(...)** **-**, e.g. -Species
matches(match) **starts_with(match)**

R Studio

RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more with browseVignettes(package = c("dplyr", "tidyverse")) • dplyr 0.7.0 • tibble 1.2.0 • Updated 2017-03



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, _____)
```



Subset Specific Rows Based on Logical Conditions

filter()

Extract rows that meet logical criteria

| | | | |
|--|--|--|--|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |



| | | | |
|--|--|--|--|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

↓ Number of rows
= Number of Columns



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



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)

| | | | | |
|--|--|--|--|-------|
| | | | | FALSE |
| | | | | FALSE |
| | | | | TRUE |
| | | | | FALSE |
| | | | | TRUE |
| | | | | FALSE |



| | | | |
|--|--|--|--|
| | | | |
| | | | |
| | | | |
| | | | |



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)

| | | | | |
|--|--|--|--|-------|
| | | | | FALSE |
| | | | | FALSE |
| | | | | TRUE |
| | | | | FALSE |
| | | | | TRUE |
| | | | | FALSE |



| | | | |
|--|--|--|--|
| | | | |
| | | | |
| | | | |
| | | | |



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 |



filter()

Extract rows that meet logical criteria

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

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

= sets

(returns nothing)

== tests if equal

(returns TRUE or FALSE)



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 |



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)



Logical tests

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



Pop Quiz

What is the result?

`1 == 1`

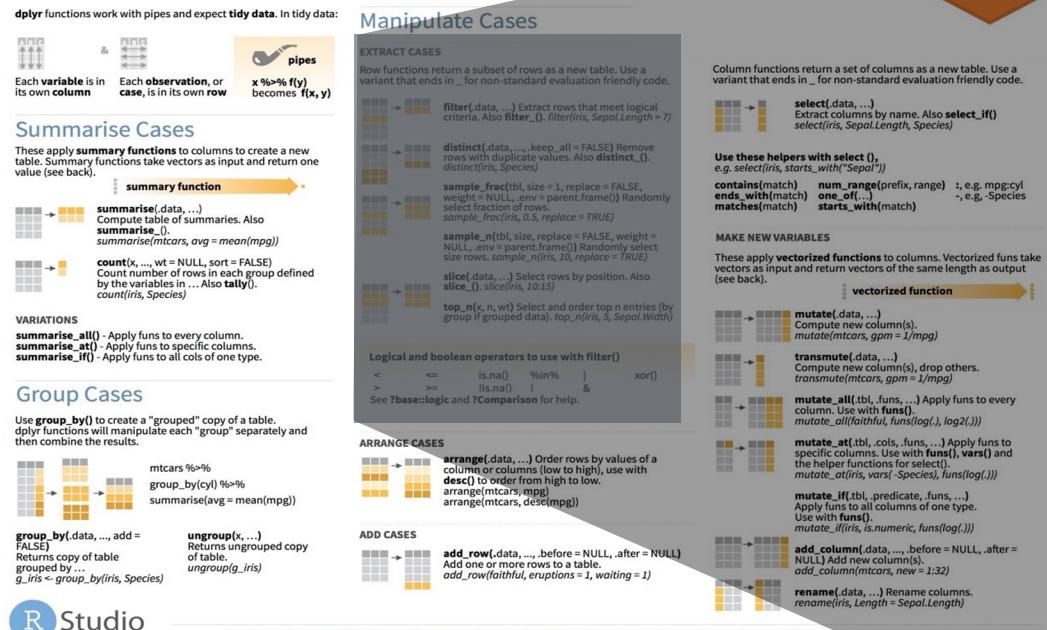
Pop Quiz

What is the result?

$$3 != 1$$

filter() variants

Data Transformation with dplyr :: CHEAT SHEET



EXTRACT CASES

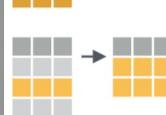
Row functions return a subset of rows as a new table.



filter(.data, ...) Extract rows that meet logical criteria. `filter(iris, Sepal.Length > 7)`



distinct(.data, ..., .keep_all = FALSE) Remove rows with duplicate values. `distinct(iris, Species)`



sample_frac(tbl, size = 1, replace = FALSE, weight = NULL, env = parent.frame()) Randomly select fraction of rows.
`sample_frac(iris, 0.5, replace = TRUE)`



slice(.data, ...) Select rows by position.
`slice(iris, 10:15)`



top_n(x, n, wt) Select and order top n entries (by group if grouped data). `top_n(iris, 5, Sepal.Width)`

Logical and boolean operators to use with filter()

| | | | | | |
|---|----|----------|------|---|-------|
| < | <= | is.na() | %in% | | xor() |
| > | >= | !is.na() | ! | & | |

See `?base::logic` and `?Comparison` for help.



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"**

Recap

dplyr is a package that contains a variety of data transformation packages intended to be used sequentially



select() will extract specific columns from a data frame



filter() extracts specific rows from a data frame based on logical conditions, and evaluates each row to determine if it is retained or removed



What else?

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 |



filter()

Filter to multiple matches

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

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



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



Goal

1. Learn how to use dplyr to transform data frames
2. Create a smaller subset from a data frame

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. Create a subset using logical conditions to extract specific rows from a data frame