Week 3: Data sci tidyverse

Charlotte Hadley

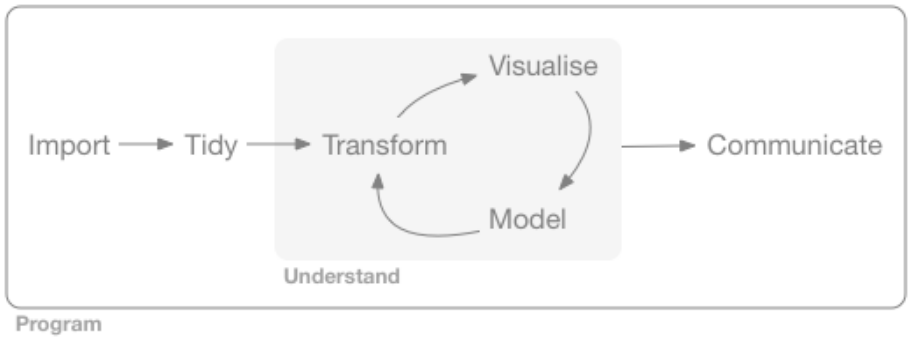
Topics for today

1. Understanding `data.frame` and `tibble`
2. Using `{readr}` for reading data
3. Using `{dplyr}` for cleaning and wrangling data
4. Using `{ggplot2}` for some basic exploratory data

What is the tidy

The tidyverse is two

- A collection of pa
  designed to work t
- A collection of pa
  the data sfo wence w



Sour Ref: or Dat[1] a S

- An opinionated framew
  work with data.

Some p[2] eowpoluld describe
tidyverse as being an
base R.

Ist ’not.

The tidyverse is a us
working with data tha
ecosystem eo’f lt ouosles.iW
quickly

> After we’ve been using the tidyverse for
> provide further context.

Task: Create a week

1. Create a new RStudio project called week-3

Installing and worki
tidyverse

To install the tidyverse collection of pack

```
install.packages("tidyverse")
```

The tidyverse packages are split into two g

- Core tidyverse packages loaded with this code

```
library(tidyverse)
```

- “Specialised” packag which are loaded explicitly {readxl} for importing data files from E

```
library(tidyverse)
library(readxl)
```

Updating the tidyver

In terms of real-world usage, keeping the
package up to date.

Thee

- When instea wpla onga e the console might promp
  of packages

```
These packages have more recent versions available.
It is recommended to update all of them.
Which would you like to update?

1: All
2: CRAN packages only
3: None
4: viridisLite (0.4.0 -> 0.4.1) [CRAN]

Enter one or more numbers, or an empty line to skip updates:
```

Updating the tidyver

In terms of real-world usage, keeping the
package up to date.

There are a few ways you might discover you s

- When installing a package the console might promp
  of packages.
- When installing a package the install fails due t
- You hear about an exciting new update to a

> There is a tidyverse_update() but it is not pr
> very rarely used.

Datasets we'll be us

We're going to be using at least 3 datasets today:

- The Global Burden of Disease study[3]. Data from the Global Health Data Exch

The Global Burden of Disease study is an ex
understanding global (and comparative) hea

There's an excellent interactive tool for download, but you
need to register for a free account to use

Datasets we'll be us

We're going to be using datasets from 3 dif

- The Global Burden of Disease Study[3].
- The msleep dataset from ggplot2 package

Lots (and lots) of R p
datasets built into th
demonstrate how to use
the package.

The msleep dataset has dat
mammalian sleep cycles
West[4]

```
1 glimpse(msleep)
```

```
Rows: 83
Columns: 11
$ name         <chr> "Cheetah", "Owl monkey", "Mountain
beaver", "Greater shor...
$ genus        <chr> "Acinonyx", "Aotus", "Aplodontia",
"Blarina", "Bos", "Bra...
$ vore         <chr> "carni", "omni", "herbi", "omni",
"herbi", "herbi", "carn...
$ order        <chr> "Carnivora", "Primates",
"Rodentia", "Soricomorpha", "Art...
$ conservation <chr> "lc", NA, "nt", "lc",
"domesticated", NA, "vu", NA, "dome...
$ sleep_total  <dbl> 12.1, 17.0, 14.4, 14.9, 4.0, 14.4,
8.7, 7.0, 10.1, 3.0, 5...
$ sleep_rem    <dbl> NA, 1.8, 2.4, 2.3, 0.7, 2.2, 1.4,
NA, 2.9, NA, 0.6, 0.8, ...
$ sleep_cycle  <dbl> NA, NA, NA, 0.1333333, 0.6666667,
0.7666667, 0.3833333, N...
$ awake        <dbl> 11.9, 7.0, 9.6, 9.1, 20.0, 9.6,
15.3, 17.0, 13.9, 21.0, 1...
$ brainwt      <dbl> NA, 0.01550, NA, 0.00029, 0.42300,
NA, NA, NA, 0.07000, 0...
$ bodywt       <dbl> 50.000, 0.480, 1.350, 0.019,
600.000, 3.850, 20.490, 0.04
```

Datasets we'll be us

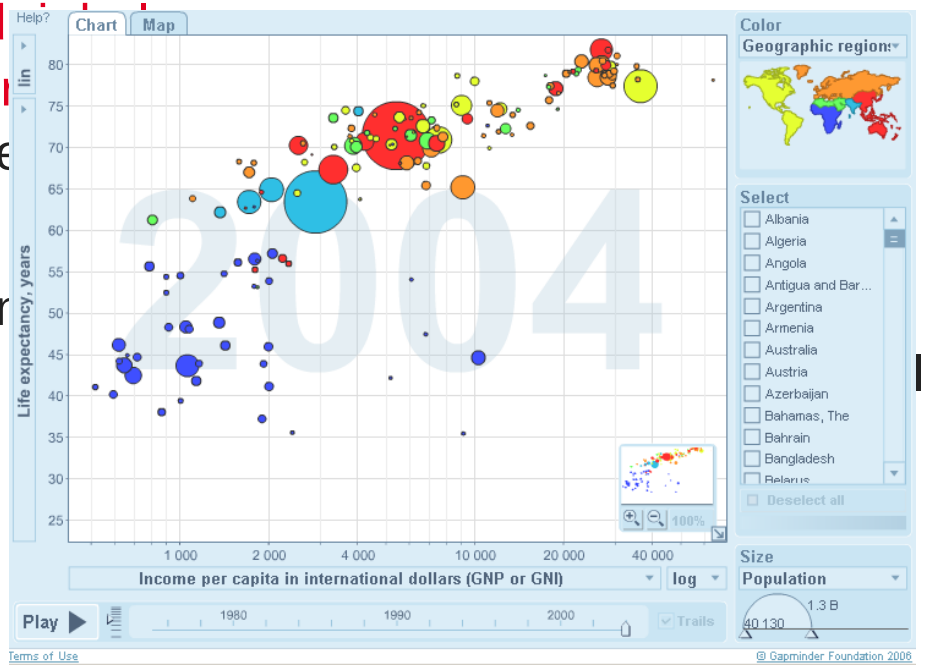
We're going to be using datasets from 3 data sources:

- The Global Burden of Disease Study[3].
- The `gapminder` dataset from the `gapminder` package[4].
- The `nasleedata` dataset from the `nasleedata` package.

In 2006 Hans Rosling gave a TED talk where he introduced bubble charts as a tool to te global development.[5]

Hans Rosling also four Foundation to promote development.

The `gapminder` package cont subset of their data.



msleep

Task: Get the msleep

1. Add a heading for the msleep dataset.
2. Load the tidyverse package in the setup code chunk.
3. Add a new code chunk and print the msleep object to the screen.

```
msleep
```

Understanding our ob

The datasets embedded in {tidyverse} are tibbles and those gen
data files with {readr}, {readxl} and {haven} a

It's important to understand how this differ**e**nt data frames

data.frames (I)

The data.frame is R's general purpose rectangular data structure required to build {ggplot2} charts

The {datasets} package has a number of data.frames built-in, c

```
1 head(quakes)
```

```
     lat   long depth mag stations
1 -20.42 181.62   562 4.8       41
2 -20.62 181.03   650 4.2       15
3 -26.00 184.10    42 5.4       43
4 -17.97 181.66   626 4.1       19
5 -20.42 181.96   649 4.0       11
6 -19.68 184.31   195 4.0       12
```

The class() function is the way to find an object's 'type' or 'form' to work with:

```
1 class(quakes)
```

```
[1] "data.frame"
```

data.frames (II)

Because they're actually vectors, data.frames manage rows and columns separately

We can print the dimensions with `dim()` or `data.frame.ncol()` and `nrow()` of a data.frame:

```
dim(quakes)
```

```
[1] 1000    5
```

In Base R there are two ways to extract columns:

- The `$` operator allows us to extract columns via their name, with autocompletion:
- The `[,]` operator allows us to extract both rows and columns by their index

```
quakes$mag
```

```
quakes[, "mag"]
```

```
quakes[, 1]
```

data.frame is dead, long live tibble

The tidyverse introduces a new, improved data frame called the tibble.

Let's demonstrate the differences by loading the tidyverse

```
library("tidyverse")
```

The first thing we notice is that tibbles are much more concise when we print them to the console

```
quakes
```

```
      lat   long depth mag stations
1  -20.42 181.62   562 4.8       41
2  -20.62 181.03   650 4.2       15
3  -26.00 184.10    42 5.4       43
4  -17.97 181.66   626 4.1       19
5  -20.42 181.96   649 4.0       11
6  -19.68 184.31   195 4.0       12
7  -11.70 166.10    82 4.8       43
8  -28.11 181.93   194 4.4       15
9  -28.74 181.74   211 4.7       35
10 -17.47 179.59   622 4.3       19
11 -21.44 180.69   583 4.4       13
12 -12.26 167.00   249 4.6       16
13 -18.54 182.11   554 4.4       19
14 -21.00 181.66   600 4.4       10
15 -20.70 169.92   139 6.1       94
16 -15.94 184.95   306 4.3       11
17 -13.64 165.96    50 6.0       83
18 -17.83 181.50   590 4.5       21
19 -23.50 179.78   570 4.4       13
20 -22.63 180.31   598 4.4       18
21 -20.84 181.16   576 4.5       17
22 -10.98 166.32   211 4.2       12
23 -23.30 180.16   512 4.4       18
```

```
starwars
```

```
# A tibble: 87 × 14
   name         height  mass hair_…¹ skin_…² eye_c…³ birth…⁴ sex   gender homew…⁵
   <chr>         <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr> <chr>  <chr>
 1 Luke Skywa…    172    77 blond   fair    blue       19   male  mascu… Tatooi…
 2 C-3PO          167    75 <NA>    gold    yellow    112   none  mascu… Tatooi…
 3 R2-D2           96    32 <NA>    white,… red        33   none  mascu… Naboo
 4 Darth Vader    202   136 none    white   yellow     41.9 male  mascu… Tatooi…
 5 Leia Organa    150    49 brown   light   brown      19   fema… femin… Aldera…
 6 Owen Lars      178   120 brown,… light   blue       52   male  mascu… Tatooi…
 7 Beru White…    165    75 brown   light   blue       47   fema… femin… Tatooi…
 8 R5-D4           97    32 <NA>    white,… red        NA   none  mascu… Tatooi…
 9 Biggs Dark…    183    84 black   light   brown      24   male  mascu… Tatooi…
10 Obi-Wan Ke…    182    77 auburn  fair    blue-g
```

data.frame is dead, long

In this course we'[illegible] data.frame [illegible] tibble.

The {dplyr} provides the flexible select() function:

```r
select(quakes, mag)
```

```
    mag
1   4.8
2   4.2
3   5.4
4   4.1
5   4.0
6   4.0
7   4.8
8   4.4
9   4.7
10  4.3
11  4.4
12  4.6
13  4.4
14  4.4
15  6.1
16  4.3
17  6.0
18  4.5
19  4.4
20  4.4
21  4.5
22  4.2
23  4.4
```

```r
select(starwars, name)
```

```
# A tibble: 87 × 1
   name
   <chr>
 1 Luke Skywalker
 2 C-3PO
 3 R2-D2
 4 Darth Vader
 5 Leia Organa
 6 Owen Lars
 7 Beru Whitesun lars
 8 R5-D4
 9 Biggs Darklighter
10 Obi-Wan Kenobi
# … with 77 more rows
```

data.frame is dead, long live tibble

The `select()` function from `dplyr` returns a tibble, but sometimes we need a vector

A vector is a one-dimensional atomic object function:

```
c(1, "2", 3)
```

```
[1] "1" "2" "3"
```

If we want to extract a column as a vector, we can use `pull()`

```
pull(starwars, name)
```

```
 [1] "Luke Skywalker"        "C-3PO"                 "R2-D2"
 [4] "Darth Vader"           "Leia Organa"           "Owen Lars"
 [7] "Beru Whitesun lars"    "R5-D4"                 "Biggs Darklighter"
[10] "Obi-Wan Kenobi"        "Anakin Skywalker"      "Wilhuff Tarkin"
[13] "Chewbacca"             "Han Solo"              "Greedo"
[16] "Jabba Desilijic Tiure" "Wedge Antilles"        "Jek Tono Porkins"
[19] "Yoda"                  "Palpatine"             "Boba Fett"
[22] "IG-88"                 "Bossk"                 "Lando Calrissian"
[25] "Lobot"                 "Ackbar"                "Mon Mothma"
[28] "Arvel Crynyd"          "Wicket Systri Warrick" "Nien Nunb"
[31] "Qui-Gon Jinn"          "Nute Gunray"           "Finis Valorum"
[34] "Jar Jar Binks"         "Roos Tarpals"          "Rugor Nass"
[37] "Ric Olié"              "Watto"                 "Sebulba"
[40] "Quarsh Panaka"         "Shmi Skywalker"        "Darth Maul"
[43] "Bib Fortuna"           "Ayla Secura"           "Dud Bolt"
[46] "Gasgano"               "Ben Quadinaros"        "Mace Windu"
[49] "Ki-Adi-Mundi"          "Kit Fisto"             "Eeth Koth"
[52] "Adi Gallia"            "Saesee Tiin"           "Yarael Poof"
[55] "Plo Koon"              "Mas Amedda"            "Gregar Typho"
[58] "Cordé"                 "Cliegg Lars"           "Poggle the Lesser"
```

Exploring our datase

Lest 'get to grips with our dataset.

How many animals do we have for each diet

We can calculate this using the `count()` function

```
1 count(msleep, vore)
```

```
# A tibble: 5 × 2
  vore      n
  <chr>   <int>
1 carni      19
2 herbi      32
3 insecti     5
4 omni       20
5 <NA>        7
```

count documentation

If we consult the documentation count(), we for an exp... beast:

> %>%

```
starwars %>%
 count(species, homeworld, sort = TRUE)
```

The code we will use to count the column uses %>% ...

```
bechdel %>%
  count(binary)
```

Let's address what %>% does...

Little Bunny Foo Foo

To introduce pipes, we're going to borrow a Hadley Wickham

How can we convert this poem into code?

> Little bunny Foo Foo
> Went hopping through the forest
> Scooping field mice up
> And bopping them on the head

Coding up little bun

> Little bunny Foo Foo
> Went hopping through the forest
> Scooping up the field mice
> And bopping them on the head

Lest’ create an instance of a bunny called `foo_foo`

```
foo_foo <- little_bunny()
```

Now sl ewtr’i te the poem out as code:

```
bop_on(
  scoop_up(
    hop_through(foo_foo, forest),
    field_mouse
  ),
  head
)
```

Understanding our co

In order to understand what our code does,

- Find the deepest expression and see what it does (that happens)
- Work backwards (or up) the code

```
bop_on(
  scoop_up(
   hop_through(foo_foo, forest),
    field_mouse
  ),
  head
)
```

This is exactly counter to the order of ope

Piping little bunny

Lest 'instantiate a foo_bunny called

```
foo_foo <- little_bunny()
```

Now write the same code as before but using

```
foo_foo %>%
  hop_through(forest) %>%
  scoop_up(field_mouse) %>%
  bop_on(head)
```

The order we read operations is exactly the
happen!

Comparing the two

Independent of pipes, we create ourselves a

```
foo_foo <- little_bunny()
```

Now comparing the two code samples, the one

```
bop_on(
  scoop_up(
    hop_through(foo_foo, forest),
    field_mouse
  ),
  head
)
```

```
foo_foo %>%
  hop_through(forest) %>%
  scoop_up(field_mouse) %>%
  bop_on(head)
```

> Admit, edt lhyi s doesn%>t%aecxtpulaalilny wdhoaets !

Simpler %>% example

The pipe operator takes the left-hand first side argument of the right-hand side of the exp

```
"cats" %>% rep(4)
```

There's nothing special about %>% which is doing the work

%>% is an example of a tool that makes code easi

Pushing the pipe fur

In some cases you don't wirasntt atrhgeumleenftt,- hyaonud shove it somewhere else by using a period

```
1 "cats" %>% paste(., "are great", "but one can have too many", .)
```

Where does %>% come from?

magrittr is the package that first introduced %>%, in 2... has become ridiculously popular.

The pipe is now a tidyverse... made available with it.

If you want to use %>% in your own package, use this:

```
usethis::use_pipe()
```

Advice on using

The pipe isn't a hammer to be used without and read with pipes.

Try to break pipe chains into blocks of sim understand at a glance:

```
raw_data <- read_csv("data-raw/the-file.csv")

clean_data <- raw_data %>%
  clean() %>%
  clean_it() %>%
  cleaning() %>%
  cleaned()

clean_data <- clean_data %>%
  normalise() %>%
  normaler() %>%
  norm_it()
```

What if I? hate %>%

Is it perfectly acceptable to hate %>%?

That's fine.

Is it just sugar to sweeten the already lovely

However, you need a basic understanding of it
tidyverse (and beyond).

Hierarchical countin

We can count by as many attributes as we li

```
msleep %>%
  count(order, vore, sort = TRUE)
```

```
# A tibble: 32 × 3
   order          vore        n
   <chr>          <chr>   <int>
 1 Rodentia       herbi      16
 2 Carnivora      carni      12
 3 Primates       omni       10
 4 Artiodactyla   herbi       5
 5 Cetacea        carni       3
 6 Perissodactyla herbi       3
 7 Rodentia       <NA>        3
 8 Soricomorpha   omni        3
 9 Chiroptera     insecti     2
10 Hyracoidea     herbi       2
# ... with 22 more rows
```

Is there any missing

The `filter()` function allows us to query a datas

```
msleep %>%
  filter(sleep_total > 12)
```

We use `==` for equivalence tests:

```
msleep %>%
  filter(vore == "carni")
```

We can negate conditions in two ways: two dif

```
msleep %>%
  filter(vore != "carni")
```

```
msleep %>%
  filter(!vore == "carni")
```

Is there any missing

We can't use an equality test, instead we use `is.na()`

```
msleep %>%
  filter(is.na(conservation))
```

The `drop_na()` function returns only rows

```
msleep %>%
  drop_na()
```

{naniar} and {ggplot

We're going to introduce {ggplot2} today wh
scratch.

There are lots of extension packages that ... abilities of {ggp

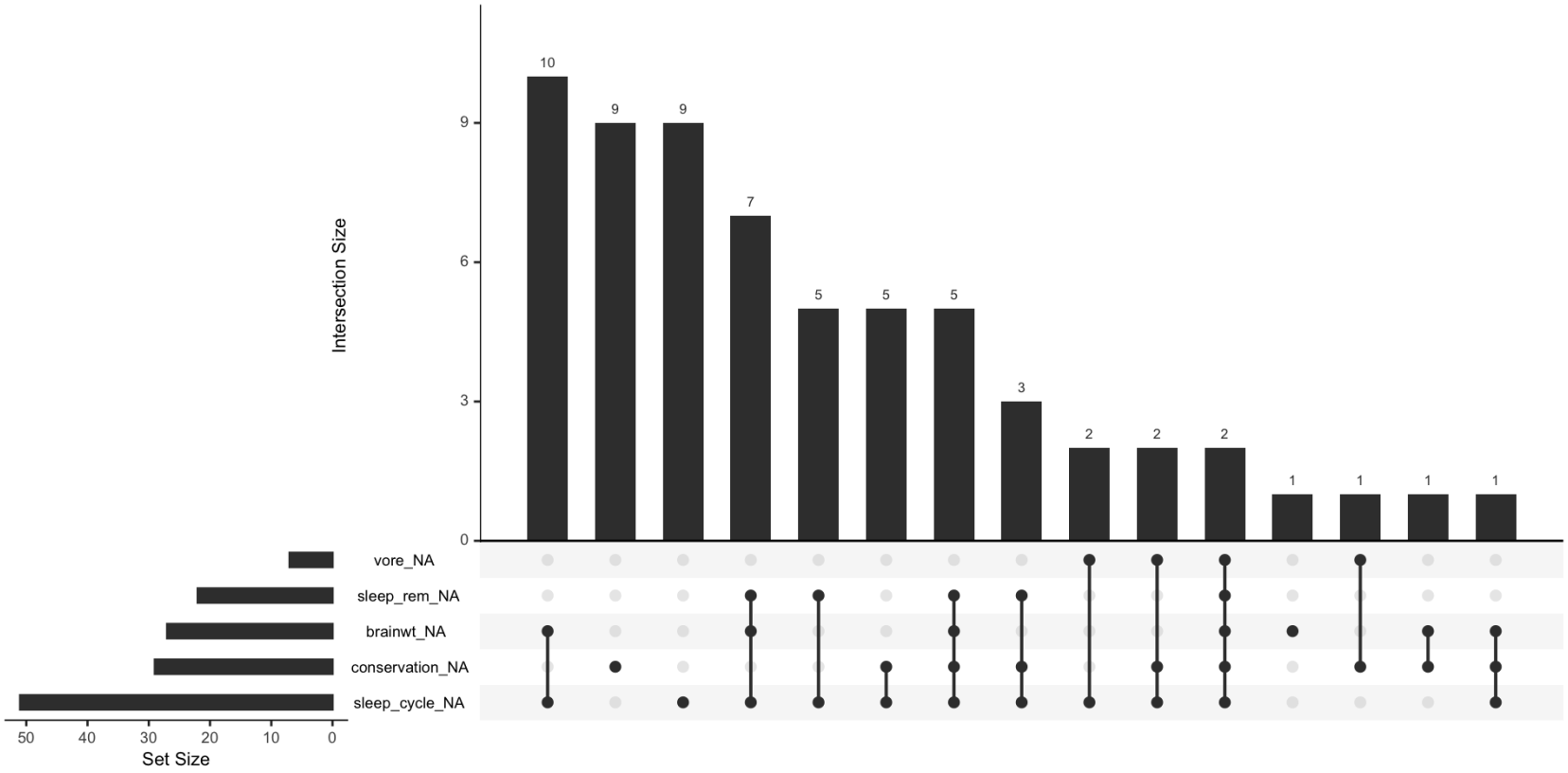
Some of these packages ... Some of these packages
additional geoms to help you build data visualisati
charts.

{naniar} is an exampl

{naniar} and {ggplot

1. Install and load the {naniar} package
2. Run this code:

```
1 msleep %>%
2   gg_miss_upset()
```



Making our own data

We're going to make two quick dataviz with the

- Scatter plot of sleep_rem vs sleep_total
- Bar chart of mean sleep_total per vore

sleep_rem scatter pl

We start ggplot2 charts by providing a data

```
msleep %>%
  ggplot()
```

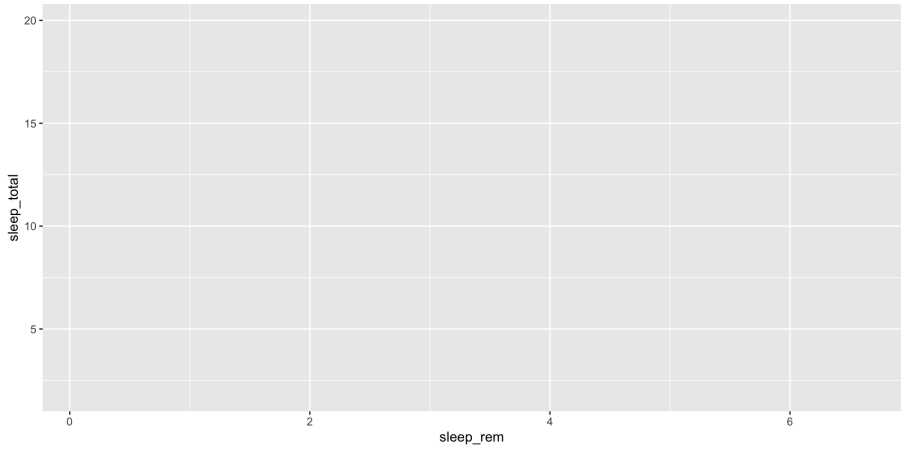

This creates an infrastructure by providing the data so it can create a meaningful dataviz.

sleep_rem scatter pl

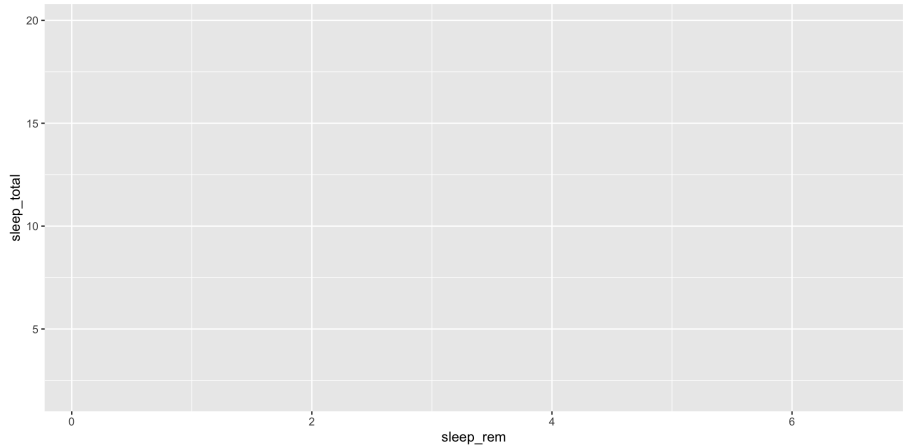
We need to tell {ggplot2} how to map columr
chart.

We do this with the aes() function. We can do this in two different ways:

```
1 msleep %>%
2   ggplot(aes(x = sleep_rem,
3              y = sleep_total))
```




```
1 msleep %>%
2   ggplot() +
3   aes(x = sleep_rem,
4       y = sleep_total)
```



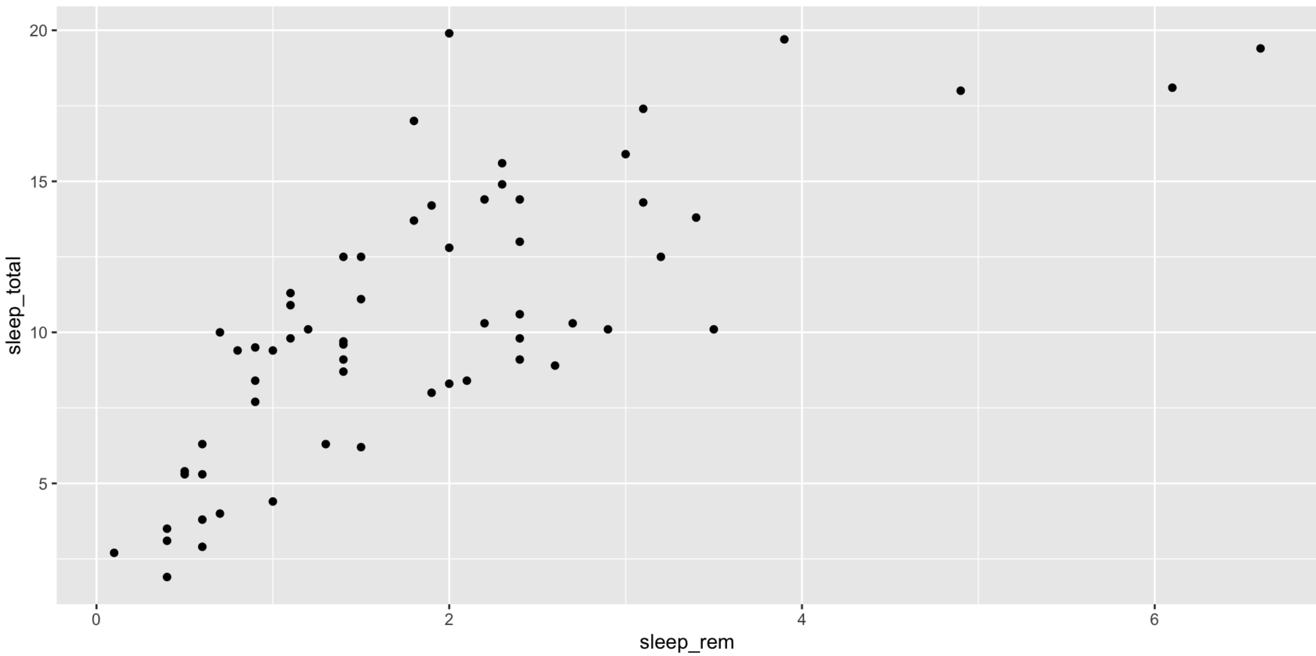
- {ggplot2} uses the columns to create the scale. And when ... numeric we get an x and y continuous scale
- The {tidyverse} functions are written spec... column names thanks to tidy evaluation

sleep_rem scatter pl

We now add geoms to our charts. These use t
shapes to our chart.

- {ggplot2} wbæsf oirtmhæve mptiepde exi+s tedadsdo wæyess

```r
msleep %>%
  ggplot() +
  aes(x = sleep_rem,
      y = sleep_total) +
  geom_point()
```



sleep_rem scatter pl

This is a pretty useless chart. It doesn't

> What can we do to improve the chart?

sleep_rem bar chart:

Before we can create a bar chart of the mea
these values!

This means introducing the function for calculatin

1.Add groups to da

2. Calculate i

13 Ungroup the
finished.

```
msleep %>%
  group_by(vore)
```

```
# A tibble: 83 × 11
# Groups:   vore [5]
   name          genus vore  order conse...¹ sleep...² sleep...³ sleep...⁴
awake  brainwt
   <chr>         <chr> <chr> <chr> <chr>     <dbl>   <dbl>   <dbl>
<dbl>    <dbl>
 1 Cheetah       Acin... carni Carn... lc         12.1    NA    NA
11.9 NA
 2 Owl monkey    Aotus omni  Prim... <NA>       17       1.8   NA       7
0.0155
 3 Mountain be... Aplo... herbi Rode... nt         14.4     2.4   NA
9.6 NA
 4 Greater sho... Blar... omni  Sori... lc         14.9     2.3   0.133
9.1  0.00029
 5 Cow           Bos   herbi Arti... domest...     4       0.7   0.667  20
0.423
 6 Three-toed ... Brad... herbi Pilo... <NA>       14.4     2.2   0.767
9.6 NA
 7 Northern fu... Call... carni Carn... vu         8.7      1.4   0.383
15.3 NA
 8 Vesper mouse Calo... <NA>  Rode... <NA>        7       NA    NA      17
NA
 9 Dog           Canis carni Carn... domest...    10.1     2.9   0.333
13.9  0.07
```

sleep_rem bar chart:

Before we can create a bar chart of the mea
these values!

This means introducing the function for calculatin

1. Add group

2. Calculate in g

- mutate() leaves

3. Ungroup the disc

```
msleep %>%
  group_by(vore) %>%
  mutate(mean_sleep_total = mean(sleep_total))
```

```
# A tibble: 83 × 12
# Groups:   vore [5]
   name          genus vore  order conse...¹ sleep...² sleep...³ sleep...⁴
awake  brainwt
   <chr>         <chr> <chr> <chr> <chr>       <dbl>   <dbl>   <dbl>
<dbl>    <dbl>
 1 Cheetah       Acin... carni Carn... lc           12.1    NA    NA
11.9 NA
 2 Owl monkey    Aotus omni  Prim... <NA>         17       1.8   NA       7
0.0155
 3 Mountain be... Aplo... herbi Rode... nt           14.4     2.4   NA
9.6 NA
 4 Greater sho... Blar... omni  Sori... lc           14.9     2.3    0.133
9.1   0.00029
 5 Cow           Bos   herbi Arti... domest...      4       0.7    0.667  20
0.423
 6 Three-toed ... Brad... herbi Pilo... <NA>         14.4     2.2    0.767
9.6 NA
 7 Northern fu... Call... carni Carn... vu           8.7      1.4    0.383
15.3 NA
 8 Vesper mouse Calo... <NA>  Rode... <NA>          7       NA    NA       17
NA
 9 Dog           Canis carni Carn... domest...     10.1     2.9    0.333
13.9   0.07
```

sleep_rem bar chart:

Before we can create a bar chart of the mea
these values!

This means introducing the group_by() function for calculatin

1. Add group

2. Calculate in g

- mutate() leaves columns
- group_by() throws away columns

3. Ungroup the sc

```
msleep %>%
  group_by(vore) %>%
  summarise(mean_sleep_total = mean(sleep_total))
```

```
# A tibble: 5 × 2
  vore    mean_sleep_total
  <chr>              <dbl>
1 carni               10.4
2 herbi                9.51
3 insecti             14.9
4 omni                10.9
5 <NA>                10.2
```

sleep_rem bar chart:

Before we can create a bar chart of the mea
these values!

This means introducing the function for calculatin

1. Add groups to

2. Calculate i

- mutate() leav
  columns
- group_by() th
  columns

3. Ungroup the data when

```
msleep %>%
  group_by(vore) %>%
  summarise(mean_sleep_total = mean(sleep_total)) %>%
  ungroup()
```

```
# A tibble: 5 × 2
  vore    mean_sleep_total
  <chr>              <dbl>
1 carni               10.4
2 herbi                9.51
3 insecti             14.9
4 omni                10.9
5 <NA>                10.2
```

sleep_rem bar chart:

Before we can create a bar chart of the mea
these values!

This means introducing the function for calculatin

1. Add groups to

2. Calculate i

- mutate() leav
  columns
- group_by() th
  columns

3. Ungroup the dataset when

```
msleep %>%
  group_by(vore) %>%
  summarise(mean_sleep_total = mean(sleep_total)) %>%
  ungroup()
```

```
# A tibble: 5 × 2
  vore    mean_sleep_total
  <chr>              <dbl>
1 carni               10.4
2 herbi                9.51
3 insecti             14.9
4 omni                10.9
5 <NA>                10.2
```

Don't forget to m
assignment for th
subset/analysis of the dataset

```
mean_sleep_by_vore <- msleep %>%
  group_by(vore) %>%
  summarise(mean_sleep_total = mean(sleep_total)) %>%
  ungroup()
```

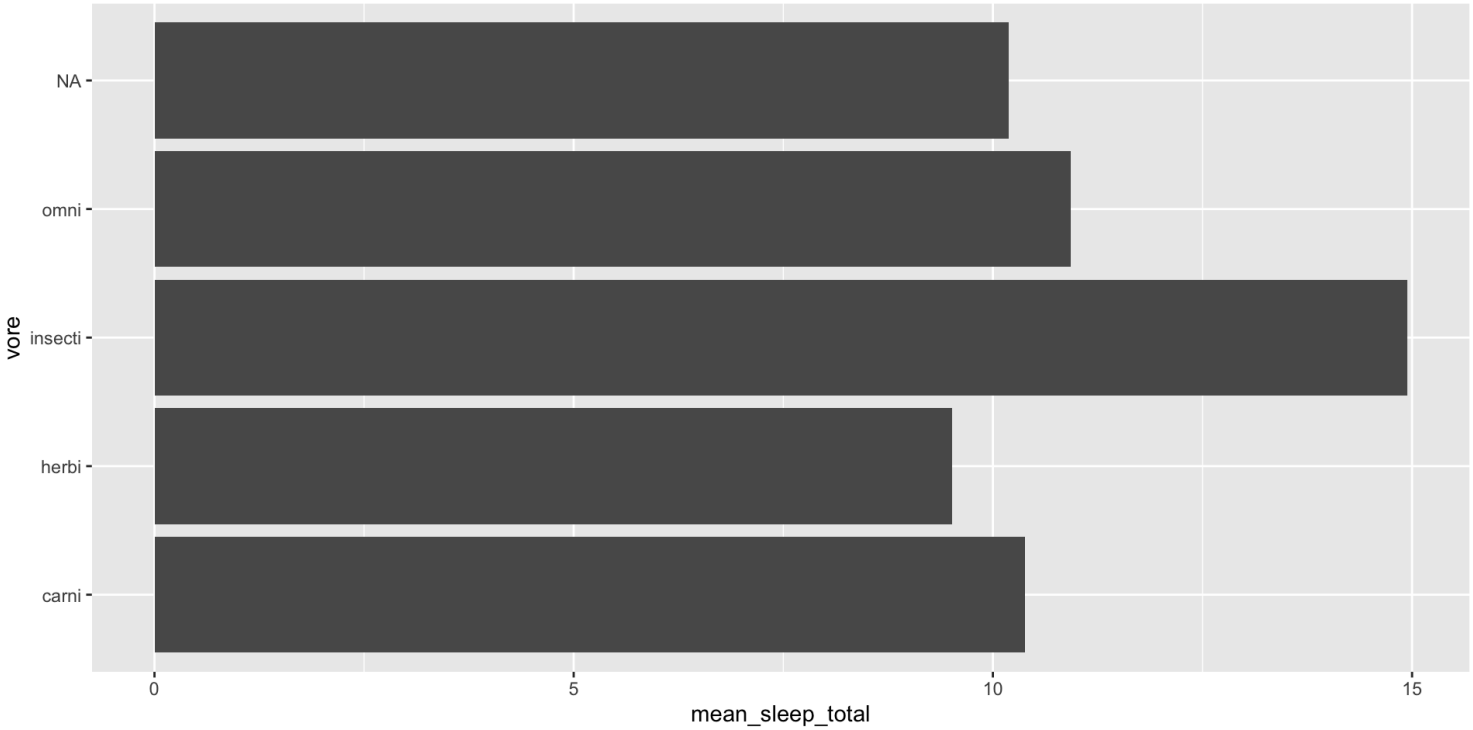
sleep_rem bar chart:

Please setup a {ggplot2} with the following

- x axis should be the “mean_sleep_total” co
- y axis should be the “vore” column

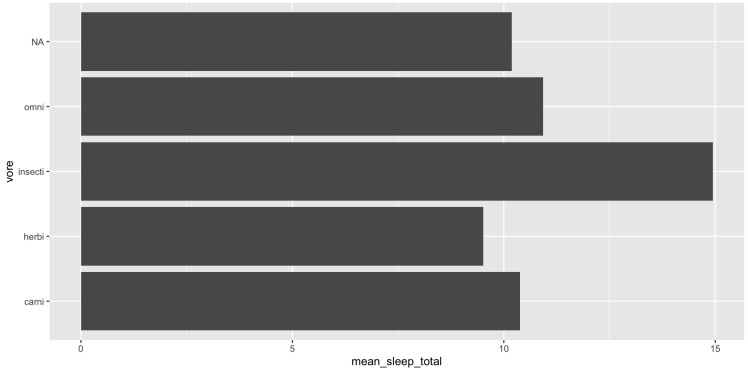
sleep_rem bar chart:

```
mean_sleep_by_vore %>%
  ggplot() +
  aes(x = mean_sleep_total,
      y = vore) +
  geom_col()
```



How can we improve this chart?!

sleep_rem bar chart:



This chart is confusing to
bars are shown.

{ggplot2} uses information
the order of scales.

- In the character (chr) colu
  ordering is used
- In the factor (fct) columns
  the levels. in the factor

> We're going to come back
> visualisation week.

Mutating multiple co

Often we need to target multiple columns a
convert* columns in the dataset to fractio

This is achieved with the

across()

The across() function tak

```
msleep %>%
  mutate(across(argument_1, argument_2))
```

Mutating multiple co

Often we need to target multiple columns a
convert (`sleep*`) columns in the dataset to fractio

This is achieved with `across()` within the

The first argument is for which columns we

This is achieved with helper functions.

```
msleep %>%
  mutate(across(argument_1, argument_2))
```

```
msleep %>%
  mutate(across(starts_with("sleep"),
                argument_2))
```

Mutating multiple co

Often we need to target multiple columns a

convert all sleep* columns in the dataset to fractio

This is achieved with the across() function

The 2nd argument is where we specify what we want to do to each column selected by the 1st argument.

```
msleep %>%
  mutate(across(argument_1, argument_2))
```

We need to write a function here.

That's usually achieved ~ with the anonymous

```
~ . / 24
```

```
msleep %>%
  mutate(across(starts_with("sleep"),
                ~ . / 24))
```

```
# A tibble: 83 × 11
   name         genus vore  order conse...¹ sleep...² sleep...³ sleep_...⁴ awake  brainwt
   <chr>        <chr> <chr> <chr> <chr>       <dbl>    <dbl>     <dbl> <dbl>    <dbl>
 1 Cheetah      Acin... carni Carn... lc        0.504 NA       NA         11.9 NA
 2 Owl monkey   Aotus omni  Prim... <NA>        0.708  0.075   NA          7    0.0155
 3 Mountain b... Aplo... herbi Rode... nt          0.6   0.1     NA          9.6 NA
 4 Greater sh... Blar... omni  Sori... lc        0.621  0.0958  0.00556    9.1  0.00029
 5 Cow          Bos   herbi Arti... domest...   0.167  0.0292  0.0278    20    0.423
 6 Three-toed... Brad... herbi Pilo... <NA>        0.6   0.0917  0.0319     9.6 NA
 7 Northern f... Call... carni Carn... vu        0.362  0.0583  0.0160    15.3 NA
 8 Vesper mou... Calo... <NA>  Rode... <NA>      0.292 NA       NA         17   NA
 9 Dog          Canis carni Carn... domest...   0.421  0.121   0.0139    13.9  0.07
10 Roe deer     Capr  herbi Arti  lc
```

Mutating multiple co

Often we need to target multiple columns a
convert all columns in the dataset to fractio

This is achieved with the

The 2nd argument is where
to each column that is selected.

We need to write a function here.

```
msleep %>%
  mutate(across(argument_1, argument_2))
```

That is usually achieved with

```
msleep %>%
  mutate(across(starts_with("sleep"),
                ~ . / 24))
```

```
~ . / 24
```

All tidyverse wrangling is done "column-wis
operations with . We'll see some examples of t

g a p m i n d e r

Task: Get the gapmi

SLIDE 1 OF 3

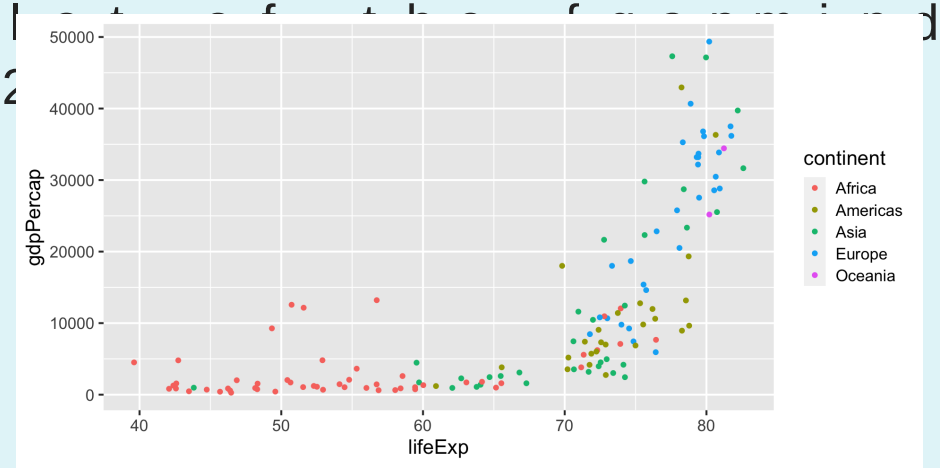
1. Add a heading for the gapminder dataset.
2. Load the gapminder package in the setup code chun
3. Add a new code chunk and print out the gapminder object

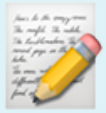
```
gapminder
```

Task: gapminder sca

SLIDE 2 OF 3

Create this scatter p dataset for the year 2

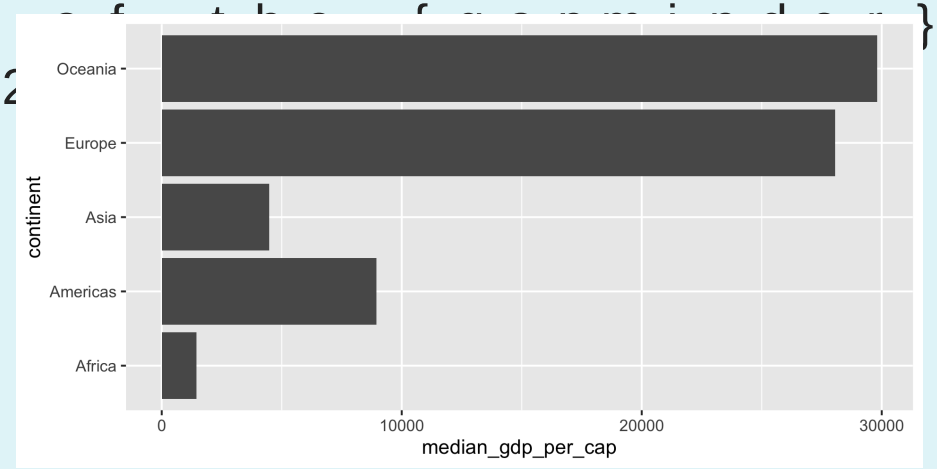




Task: gapminder bar

SLIDE 3 OF 3

Create this bar chart of the gapminder dataset for the year 2



GBD Dataset

Task: Get the GBD d

SLIDE 1 OF 2

1. Add a sub-folder to your project called data
2. Inside data folder add a bash script called
3. Add this code

```
download.file("https://raw.githubusercontent.com/charliejhadley/eng7218_data-science-for-healthcare-applications_bcu-
              destfile = "data/global-burden-of-disease-data.csv")
```

5. Run the code

📝 Task: Get the GBD d

SLIDE 2 OF 2

1.Add a new heading for the GBD Dataset to y

Reading data into R

The {readr} package provides excellent tools for reading .csv and .tsv files.

We need to think about file paths. The procedure is as follows:

1. Add a code chunk to your .Rmd
2. Choose a name for the dataset, commonly we use raw or something similar to denote this is your original data

```
disease_burden_raw <-
```

3. Call the appropriate {readr} function …

```
disease_burden_raw <- read_csv("")
```

4. Press TAB with your cursor inside the quotation marks and select your file.

```
disease_burden_raw <- read_csv("data/global-burden-of-disease-data.csv")
```

Matching text in R (

A lot of data wrangling comes down to working with "strings" or in computer science we usually refer to "strings" as "character".

The GBD dataset gives example of this.

```
disease_burden_raw %>%
  count(location_name)
```

```
# A tibble: 10 × 2
   location_name                    n
   <chr>                        <int>
 1 African Region                 720
 2 Eastern Mediterranean Region   720
 3 European Region                720
 4 Region of the Americas         720
 5 South-East Asia Region         720
 6 Western Pacific Region         720
 7 World Bank High Income         720
 8 World Bank Low Income          720
 9 World Bank Lower Middle Income 720
10 World Bank Upper Middle Income 720
```

There are three not dtiyfpes o the dataset

- World Bank regions
- Geographic regions

The tidyverse gives us { stringr } to help us work with strings.

Matching text in R (

We can search the beginning of strings with `str_starts()`

```
disease_burden_raw %>%
  filter(str_starts(location_name, "World Bank"))
```

```
# A tibble: 2,880 × 16
   measu…¹ measu…² locat…³ locat…⁴ sex_id sex_n…⁵ age_id age_n…⁶ cause…⁷ cause…⁸
     <dbl> <chr>     <dbl> <chr>    <dbl> <chr>    <dbl> <chr>     <dbl> <chr>
 1       1 Deaths    44575 World …      3 Both        22 All ag…     409 Non-co…
 2       1 Deaths    44575 World …      3 Both        22 All ag…     409 Non-co…
 3       1 Deaths    44575 World …      3 Both        22 All ag…     409 Non-co…
 4       1 Deaths    44575 World …      3 Both        22 All ag…     294 All ca…
 5       1 Deaths    44575 World …      3 Both        22 All ag…     294 All ca…
 6       1 Deaths    44575 World …      3 Both        22 All ag…     294 All ca…
 7       1 Deaths    44575 World …      3 Both        22 All ag…     295 Commun…
 8       1 Deaths    44575 World …      3 Both        22 All ag…     295 Commun…
 9       1 Deaths    44575 World …      3 Both        22 All ag…     295 Commun…
10       1 Deaths    44576 World        3 Both        22
```

Matching text in R (

For more complex strin
have to make use of RE

REGEX stands for regul
and is an approach to
that is implemented ir
languages.

We can use regular exp
pattern arguments to stringr
functions.

regex101.com is really useful tool for
building up complex regular expressions.

```
1 disease_burden_raw %>%
2   filter(str_detect(location_name, "^World Bank"))
```

```
# A tibble: 2,880 × 16
   measu...¹ measu...² locat...³ locat...⁴ sex_id sex_n...⁵ age_id
age_n...⁶ cause...⁷ cause...⁸
     <dbl> <chr>     <dbl> <chr>    <dbl> <chr>    <dbl>
<chr>     <dbl> <chr>
 1       1 Deaths    44575 World ...      3 Both        22
All ag...     409 Non-co...
 2       1 Deaths    44575 World ...      3 Both        22
All ag...     409 Non-co...
 3       1 Deaths    44575 World ...      3 Both        22
All ag...     409 Non-co...
 4       1 Deaths    44575 World ...      3 Both        22
All ag...     294 All ca...
 5       1 Deaths    44575 World ...      3 Both        22
All ag...     294 All ca...
 6       1 Deaths    44575 World ...      3 Both        22
All ag...     294 All ca...
 7       1 Deaths    44575 World ...      3 Both        22
All ag...     295 Commun...
 8       1 Deaths    44575 World ...      3 Both        22
All ag...     295 Commun...
 9       1 Deaths    44575 World ...      3 Both        22
All ag...     295 Commun...
10       1 Deaths    44576 World ...      3 Both        22
```

GBD data visualisati

I'd like to filter out the dataset so that all of thes

- We can only see data for the most recent y
- We can select to see the number of deaths for each cause_name
- There are 16 filtered rows with this dataset

GBD data visualisati

Once we've filtered down the data, select only the interest

```r
disease_burden_percent_deaths %>%
  select(location_name, cause_name, val)
```

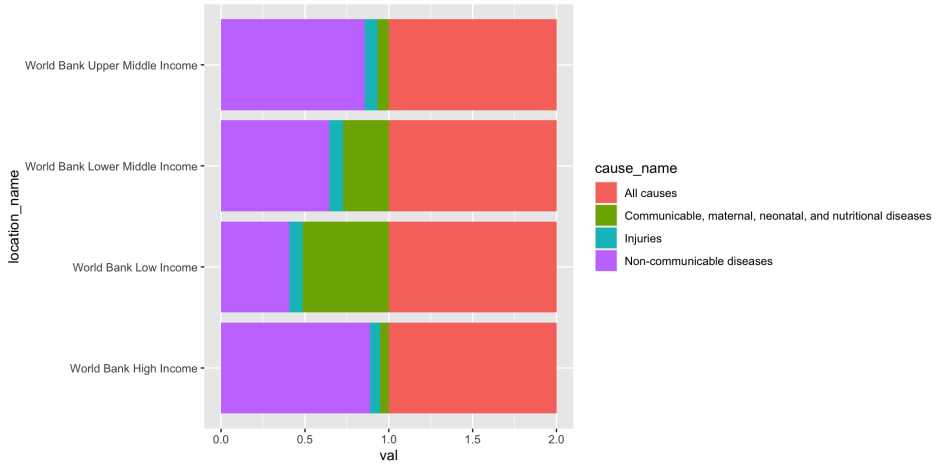
```
# A tibble: 16 × 3
   location_name                 cause_name                                 val
   <chr>                         <chr>                                    <dbl>
 1 World Bank High Income        All causes                              1     
 2 World Bank High Income        Communicable, maternal, neonatal, and … 0.0528
 3 World Bank Upper Middle Income Non-communicable diseases              0.856 
 4 World Bank Upper Middle Income All causes                             1     
 5 World Bank Upper Middle Income Communicable, maternal, neonatal, and … 0.0662
 6 World Bank Low Income         All causes                              1     
 7 World Bank Upper Middle Income Injuries                               0.0780
 8 World Bank Low Income         Communicable, maternal, neonatal, and … 0.513 
 9 World Bank High Income        Injuries                                0.0579
10 World Bank Lower Middle Income Non-communicable diseases              0.645 
11 World Bank High Income        Non-communicable diseases               0.889 
12 World Bank Low Income         Injuries                                0.0812
13 World Bank Lower Middle Income All causes                             1     
14 World Bank Lower Middle Income Communicable, maternal, neonatal, and … 0.273 
15 World Bank Lower Middle Income Injuries                               0.0824
16 World Bank Low Income         Non-communicable diseases               0.406 
```

Lest 'visualise this as a barchart

GBD data visualisati

> How can we make this chart more meaningfu

```
disease_burden_percent_deaths %>%
  select(location_name, cause_name, val) %>%
  ggplot() +
  aes(x = val,
      y = location_name,
      fill = cause_name) +
  geom_col()
```



Going further

We've only barely scratched the surface of wrangling with the tidyverse. The topics we have covered:

- Read in files with read_*() functions
- Understand the difference between tibble and data.frame
- Filter data with filter()
- Use {stringr} to search/modify strings
- Use mutate() to modify existing columns and add new columns
- Use group_by() to calculate in-group values
- Use {ggplot2} for exploratory data analysis.

- The R for Data Science is a great free resource. These reinforce and extend to more advanced …
- RStudio has a lot of … https://www.rstudio.com/re…
- We'll get into the te… {ggplot2} next lecture.
- In the "survey data" … principles of tidy da… from {tidyr} for wrangling …

References

1. Wickham, H. & Grolemund, G. R for data science: import, tidy, transform, visualize, and model data. (O'Reilly, 2016).
2. Matloff, N. Teaching R in a Kinder, Gentler, More Effective Manner: (2022).
3. Global Health Data Exchange. Global Burden of Disease Dataset Explorer. Institute for Health Metrics and Evaluation (2022).
4. Savage, V. M. & West, G. B. A quantitative, theoretical framework for understanding mammalian sleep. Proceedings of the National Academy of Sciences 104, 1051–1056 (2007).
5. Hans Rosling. The best stats you've ever seen [Video]. The best stats you've ever seen (2006).