

Scoped verbs:

A subtitle

January 2019

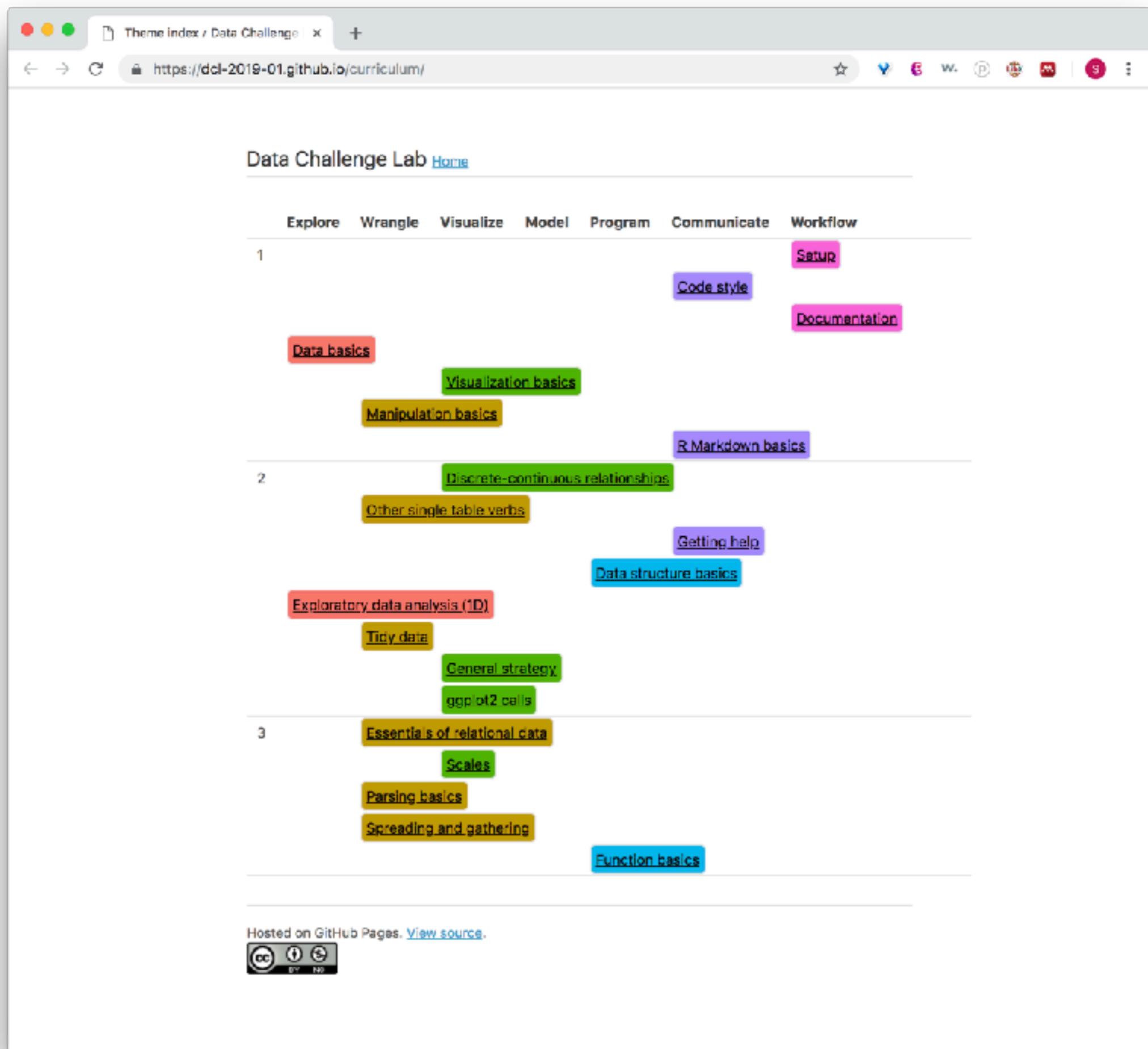
Sara Altman

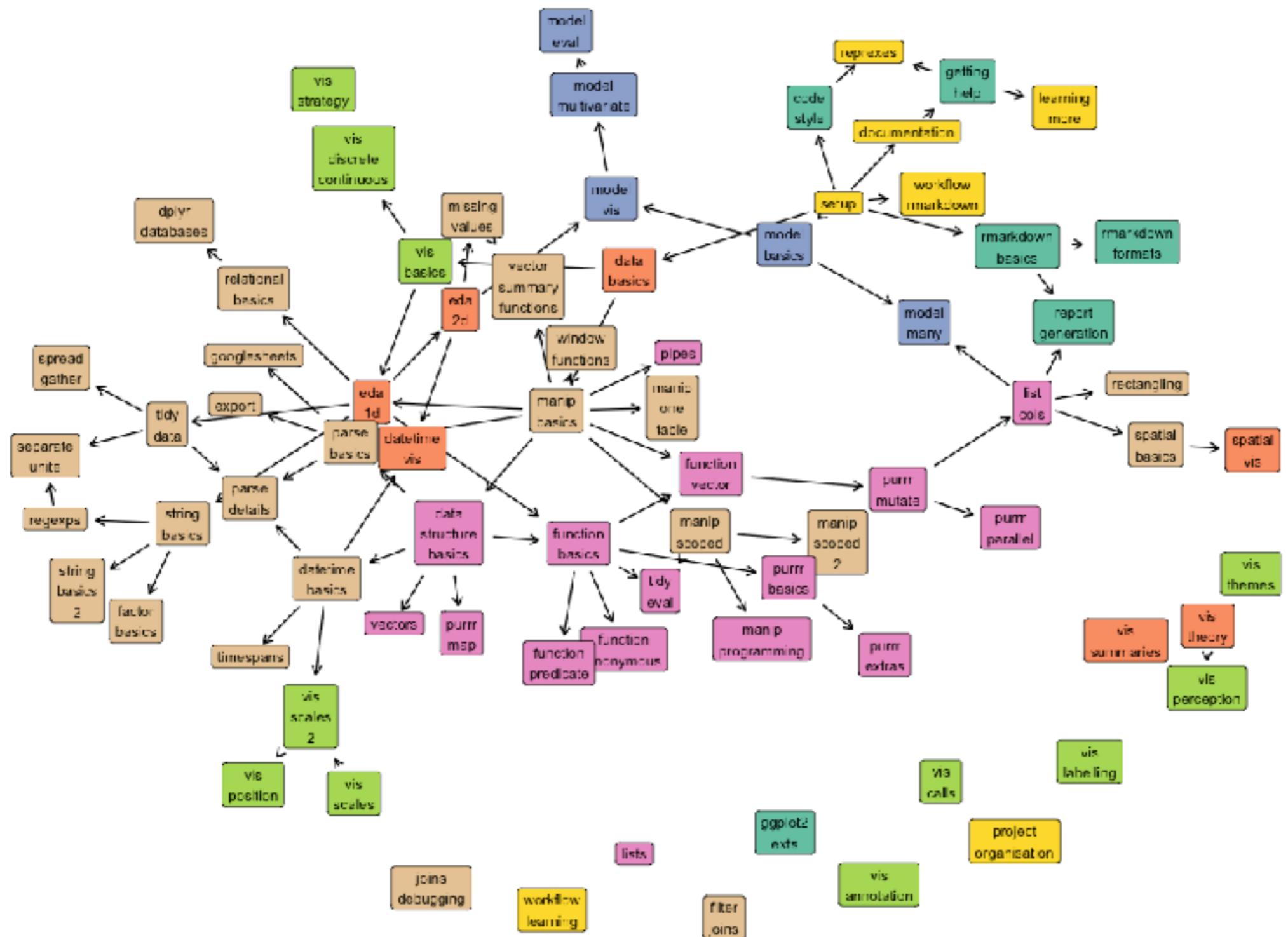
Stanford Data Lab

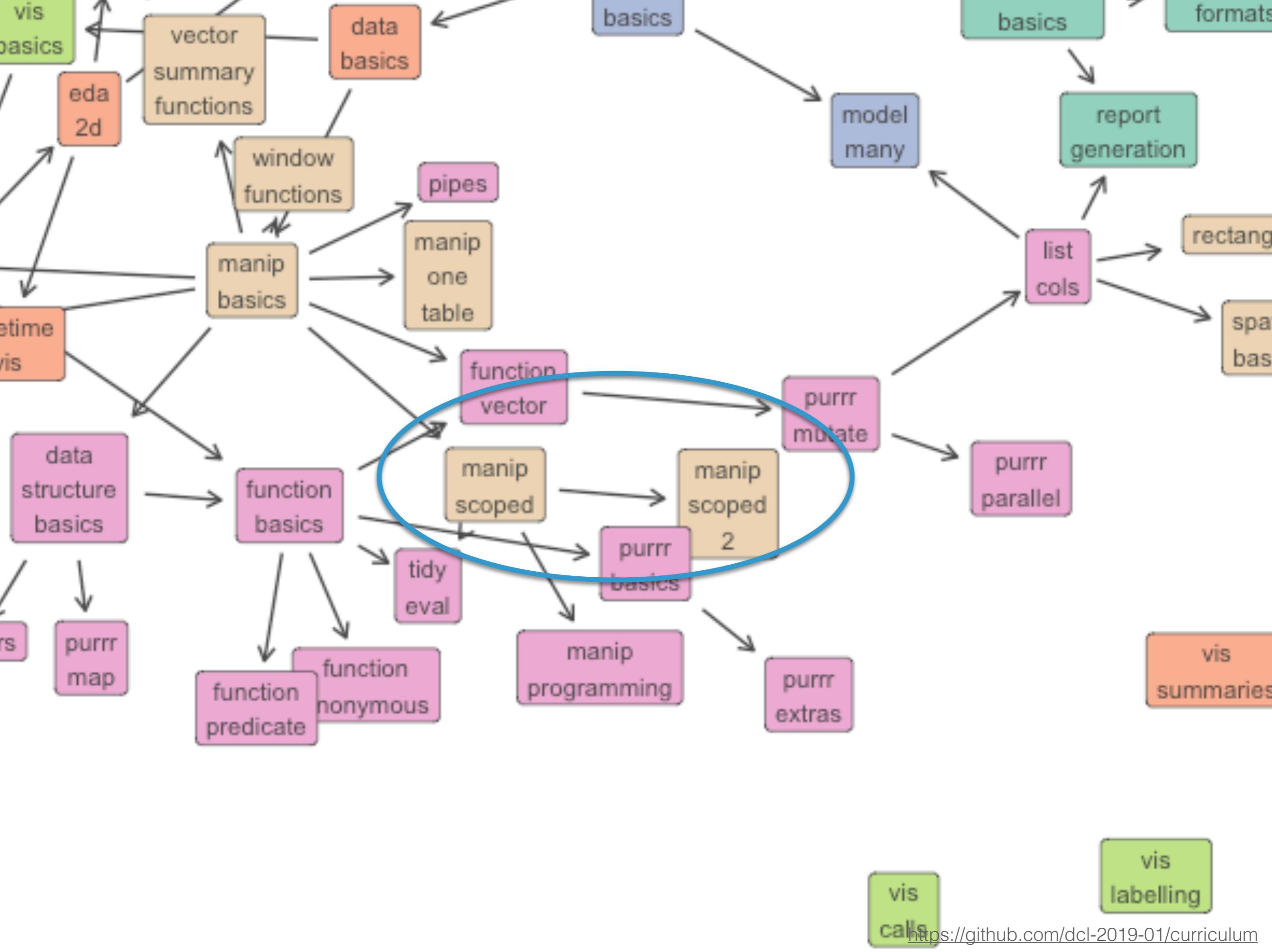
Stanford Data Lab

⋮

Data Challenge Lab







dplyr verbs

mutate()

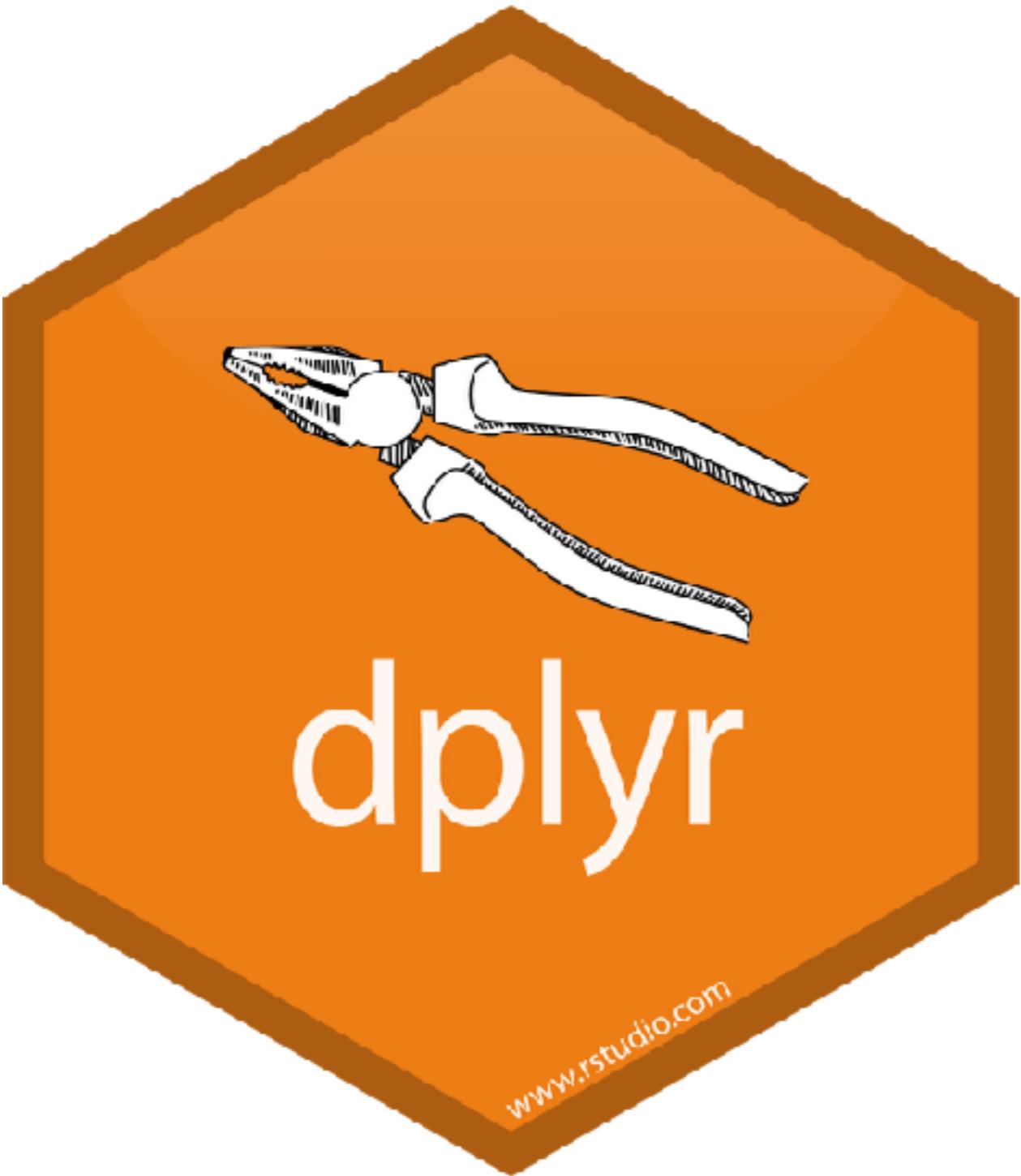
summarize()

filter()

rename()

select()

group_by()



dplyr

select helpers

predicate functions

anonymous functions

...

logic

Scoped verb basics r Data Ch x +

https://dcl-2019-01.github.io/curriculum/manip-scoped.html

Data Challenge Lab [Home](#)

Scoped verb basics [wrangle]

(Builds on: [Manipulation basics](#))
(Leads to: [Programming with dplyr](#), [Scoped verbs with predicates](#), [Tidy evaluation](#))

You'll often want to operate on multiple columns at the same time. Luckily, there are **scoped** versions of dplyr verbs that allow you to apply that verb to multiple columns at once.

Scoped verbs are powerful. They allow you to quickly carry out complex wrangling that would otherwise be much more difficult.

Each dplyr verb comes in three scoped variants. The name of each variant consists of the dplyr verb plus one of three suffixes: `_at`, `_all`, or `_if`. In this reading, you'll learn about the `_all` and `_at` scoped verbs.

`_all` and `_at` scoped verbs

x is a simple tibble.

```
x <-  
  tibble(  
    number_1 = c(1, 1, 51),  
    number_2 = c(3, 42, NA),  
    letter = c("w", "x", "w")  
)  
  
x
```

```
## # A tibble: 3 × 3  
##   number_1 number_2 letter  
##     <dbl>     <dbl> <chr>  
## 1       1        3     w  
## 2       1       42     x  
## 3      51      NA     w
```

We can use `summarize()` to find the number of distinct values for each variable.

```
x %>%  
  summarize(  
    number_1 = n_distinct(number_1),  
    number_2 = n_distinct(number_2),
```

Scoped verbs with predicates [x](#)

[←](#) [→](#) [C](#) <https://dcl-2019-01.github.io/curriculum/manip-scoped-2.html> [☆](#) [V](#) [E](#) [W](#) [P](#) [D](#) [B](#) [M](#) [S](#) [:](#)

Data Challenge Lab [Home](#)

Scoped verbs with predicates [wrangle]

(Builds on: [Scoped verb basics](#))

In the *Scoped verb basics* reading, you learned about the `_at` and `_all` variants of `mutate()`, `transmute()`, `summarize()`, `select()`, and `rename()`.

In this reading, you'll learn about scoped verbs that use **predicate functions**. First, you'll learn about the third suffix, `_if`. Then, you'll learn about the scoped variants of `filter()`.

`_if`

Like the `_at` scoped verbs, the `_if` variants apply a dplyr verb only to specified columns. The `_at` variants specify columns based on name. The `_if` variants instead use predicate functions, applying the dplyr verb only to the columns for which the predicate function is TRUE.

`small_towns` is a tibble with information about some very small towns. However, whoever collected the data didn't do a very good job. The town and state names aren't capitalized, and there are several missing values.

```
small_towns <- tribble(
  ~town, ~state, ~population, ~sq_miles,
  "bettles", "alaska", 12, 1.74,
  "gilbert", "arkansas", NA, 0.38,
  NA, "hawaii", NA, 2,
  "ruso", "north dakota", 4, NA
)
```

We could use `mutate_at()` to capitalize the town and state names.

```
small_towns %>%
  mutate_at(vars(town, state), str_to_title)
```

```
## # A tibble: 4 x 4
##   town      state    population sq_miles
##   <chr>     <chr>        <dbl>     <dbl>
## 1 Bettles  Alaska       12       1.74
## 2 Gilbert  Arkansas     NA       0.38
## 3 <NA>      Hawaii      NA       2
## 4 Russo    North Dakota 4       NA
```

```
small_towns <-  
  tribble(  
    ~town,      ~state,                  ~population,   ~sq_miles,  
    "bettles", "alaska",                12,           1.74,  
    "gilbert", "arkansas",              NA,           0.38,  
    NA,         "hawaii",                NA,           2,  
    "ruso",     "north dakota",        4,            NA  
)  
  
small_towns
```

```
## # A tibble: 4 x 4  
##   town     state   population sq_miles  
##   <chr>    <chr>     <dbl>      <dbl>  
## 1 bettles  alaska       12      1.74  
## 2 gilbert  arkansas     NA      0.38  
## 3 <NA>     hawaii       NA       2  
## 4 ruso    north dakota     4      NA
```

Simple case

```
small_towns %>%  
  summarize(  
    town = n_distinct(town),  
    state = n_distinct(state),  
    population = n_distinct(population),  
    sq_miles = n_distinct(sq_miles)  
)
```

```
## # A tibble: 1 x 4  
##   town state population sq_miles  
##   <int> <int>       <int>     <int>  
## 1     4      4         4          3          4
```

duplication!



Format

dplyr verb

suffix

mutate

_all

summarize

_at

filter

+

rename

_if

...

Simple case

```
small_towns %>%  
  summarize_all( )
```

```
## # A tibble: 1 x 4  
##   town state population sq_miles  
##   <int> <int>       <int>     <int>  
## 1     4     4         3         4
```

```
## # A tibble: 4 x 4
##   town      state population sq_miles
##   <chr>    <chr>     <dbl>      <dbl>
## 1 bettles  alaska       12      1.74
## 2 gilbert arkansas     NA      0.38
## 3 <NA>     hawaii      NA        2
## 4 ruso     north dakota     4      NA
```

_at

```
small_towns %>%  
  mutate_at(
```

```
## # A tibble: 4 x 4  
##   town      state    population  sq_miles  
##   <chr>     <chr>      <dbl>        <dbl>  
## 1 Bettles  Alaska       12        1.74  
## 2 Gilbert  Arkansas     NA        0.38  
## 3 <NA>      Hawaii      NA         2  
## 4 Russo    North Dakota     4        NA
```

...
...

```
mutate_all(.tbl, .fun, ...)
```

```
small_towns %>%  
  summarize_at(vars(population, sq_miles), median,  
  )
```

Anonymous functions

```
small_towns %>%  
  summarize_all(~ ))
```

Anonymous functions

```
ugly_names <-  
  tibble(  
    Var.1 = c(1, 2),  
    Var.2 = c(3, 4)  
)
```

```
small_towns %>%  
  mutate_at(vars(town, state), str_to_title)
```

```
## # A tibble: 4 x 4  
##   town      state    population  sq_miles  
##   <chr>     <chr>      <dbl>        <dbl>  
## 1 Bettles  Alaska       12        1.74  
## 2 Gilbert  Arkansas     NA        0.38  
## 3 <NA>      Hawaii      NA         2  
## 4 Russo    North Dakota     4        NA
```

Predicate functions

```
small_towns %>%  
  mutate_if(is.character, str_to_title)
```

```
## # A tibble: 4 x 4  
##   town      state    population  sq_miles  
##   <chr>     <chr>      <dbl>        <dbl>  
## 1 Bettles  Alaska       12        1.74  
## 2 Gilbert  Arkansas     NA        0.38  
## 3 <NA>      Hawaii      NA         2  
## 4 Russo    North Dakota 4        NA
```

filter()

```
## # A tibble: 4 x 4
##   town      state population sq_miles
##   <chr>     <chr>      <dbl>     <dbl>
## 1 bettles   alaska       12      1.74
## 2 gilbert   arkansas    NA      0.38
## 3 <NA>      hawaii      NA        2
## 4 ruso      north dakota  4      NA
```

any_vars(), all_vars()

```
## # A tibble: 4 x 4
##   town      state population sq_miles
##   <chr>     <chr>      <dbl>     <dbl>
## 1 bettles   alaska       12      1.74
## 2 gilbert   arkansas    NA      0.38
## 3 <NA>      hawaii      NA        2
## 4 ruso      north dakota  4       NA
```

all_vars()

]
any_vars()

any_vars(), all_vars()

```
small_towns %>%  
  filter_at(vars(town, population, sq_miles), all_vars(!is.na(.)))
```

filter_if()

```
small_towns %>%  
  filter_if(is.numeric, all_vars(!is.na(.)))
```

Readings

<https://dcl-2019-01.github.io/curriculum/manip-scoped.html>

<https://dcl-2019-01.github.io/curriculum/manip-scoped.html>

<https://dcl-2019-01.github.io/curriculum/function-predicate.html>

<https://dcl-2019-01.github.io/curriculum/function-anonymous.html>

<https://github.com/skaltman/slides/blob/master/manip-scoped-reference.md>

Sara Altman

skaltman@stanford.edu

[GitHub: skaltman](https://github.com/skaltman)

Stanford Data Lab

datalab.stanford.edu

<https://datalab.stanford.edu/sign-up>

<https://github.com/dcl-2019-01/curriculum>