Checking membership

CLEANING DATA IN R

Maggie Matsui
Content Developer @ DataCamp

Categorical data

- Categorical variables have a fixed and known set of possible values

Data	Example values
Marriage status	`unmarried`, `married`
Household income category	`0-20K`, `20-40K`, ...
T-shirt size	`S`, `M`, `L`, `XL`

Factors

- In a `factor`, each category is stored as a number number and has a corresponding label

Data	Labels	Numeric representation
Marriage status	`unmarried`, `married`	`1`, `2`
Household income category	`0-20K`, `20-40K`, ...	`1`, `2`, ...
T-shirt size	`S`, `M`, `L`, `XL`	`1`, `2`, `3`, `4`

Factor levels

```
tshirt_size
```

```
L  XL XL L  M  M  M  L  XL L  S  M  M  S  S  M  XL S  L  S ...
Levels: S M L XL
```

```
levels(tshirt_size)
```

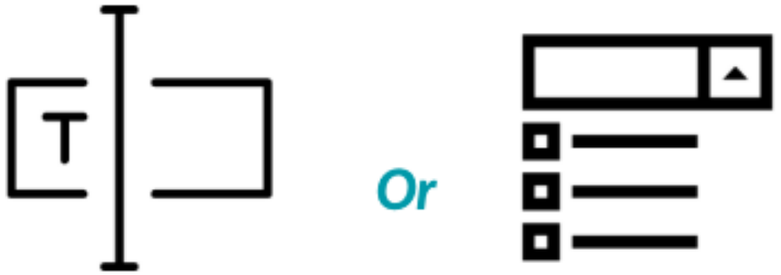
```
"S"  "M"  "L"  "XL"
```

Values that don't belong

- `factor` s cannot have values that fall outside of the predefined ones

Data	Levels	Not allowed
Marriage status	`unmarried` , `married`	`divorced`
Household income category	`0-20K` , `20-40K` , ...	`10-30K`
T-shirt size	`S` , `M` , `L` , `XL`	`S/M`

How do we end up with these values?

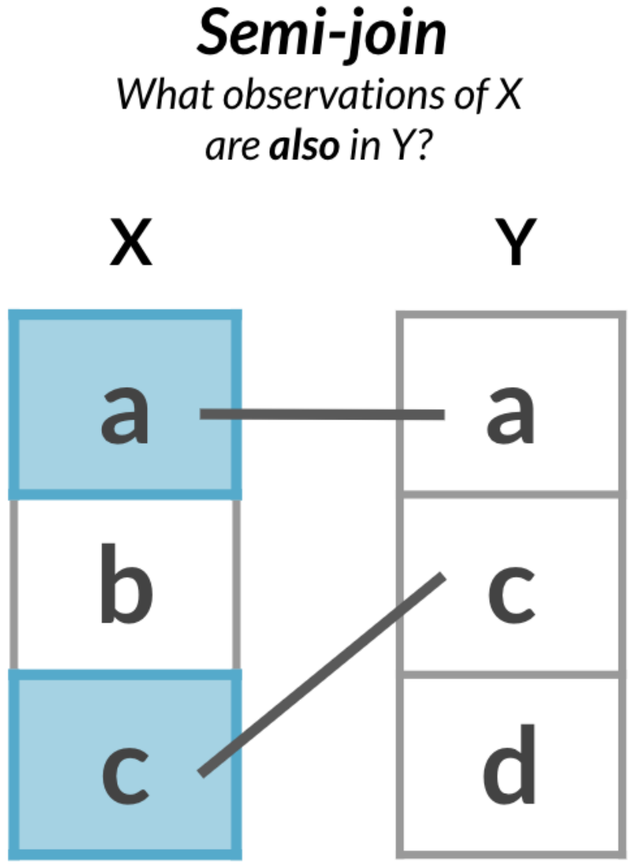


Data Entry Errors

Parsing Errors

Filtering joins: a quick review

- Keeps or removes observations from the first table without adding columns

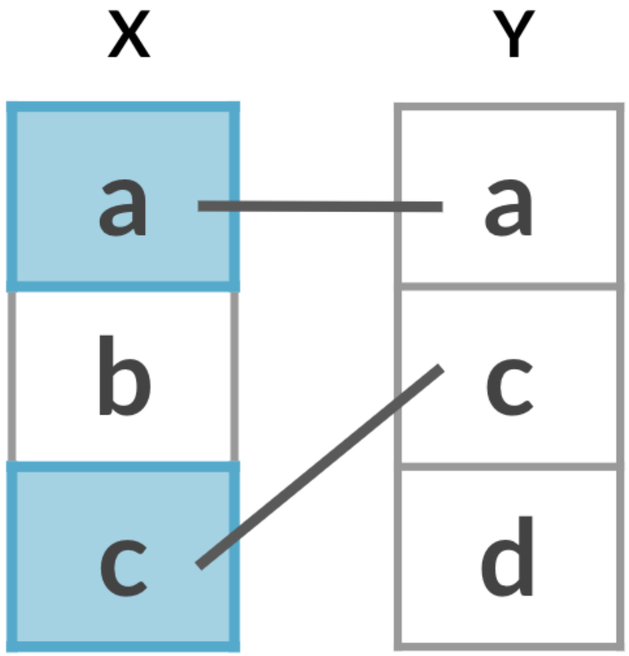


Filtering joins: a quick review

- Keeps or removes observations from the first table without adding columns

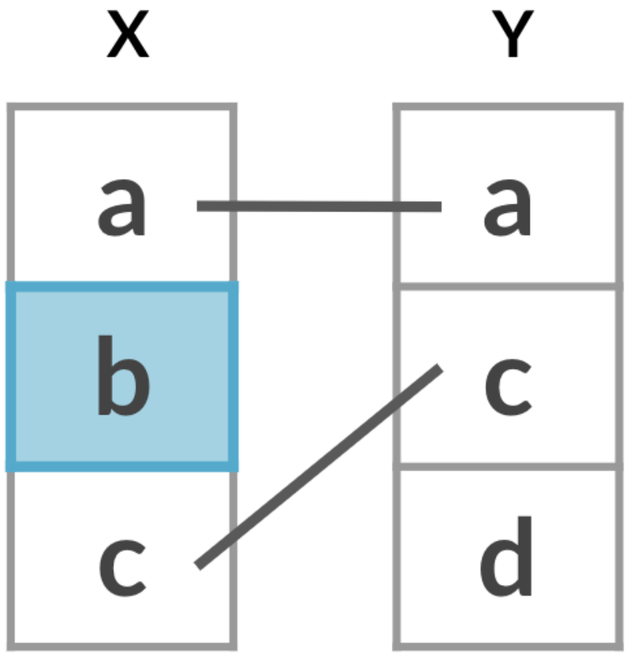
Semi-join

*What observations of X are **also** in Y?*



Anti-join

*What observations of X are **not** in Y?*



Blood type example

`study_data`

	name	birthday	blood_type
1	Beth	2019-10-20	B-
2	Ignatius	2020-07-08	A-
3	Paul	2019-08-12	O+
4	Helen	2019-03-17	O-
5	Jennifer	2019-12-17	Z+
6	Kennedy	2020-04-27	A+
7	Keith	2019-04-19	AB+

`blood_types`

	blood_type
1	O-
2	O+
3	A-
4	A+
5	B+
6	B-
7	AB+
8	AB-

Blood type example

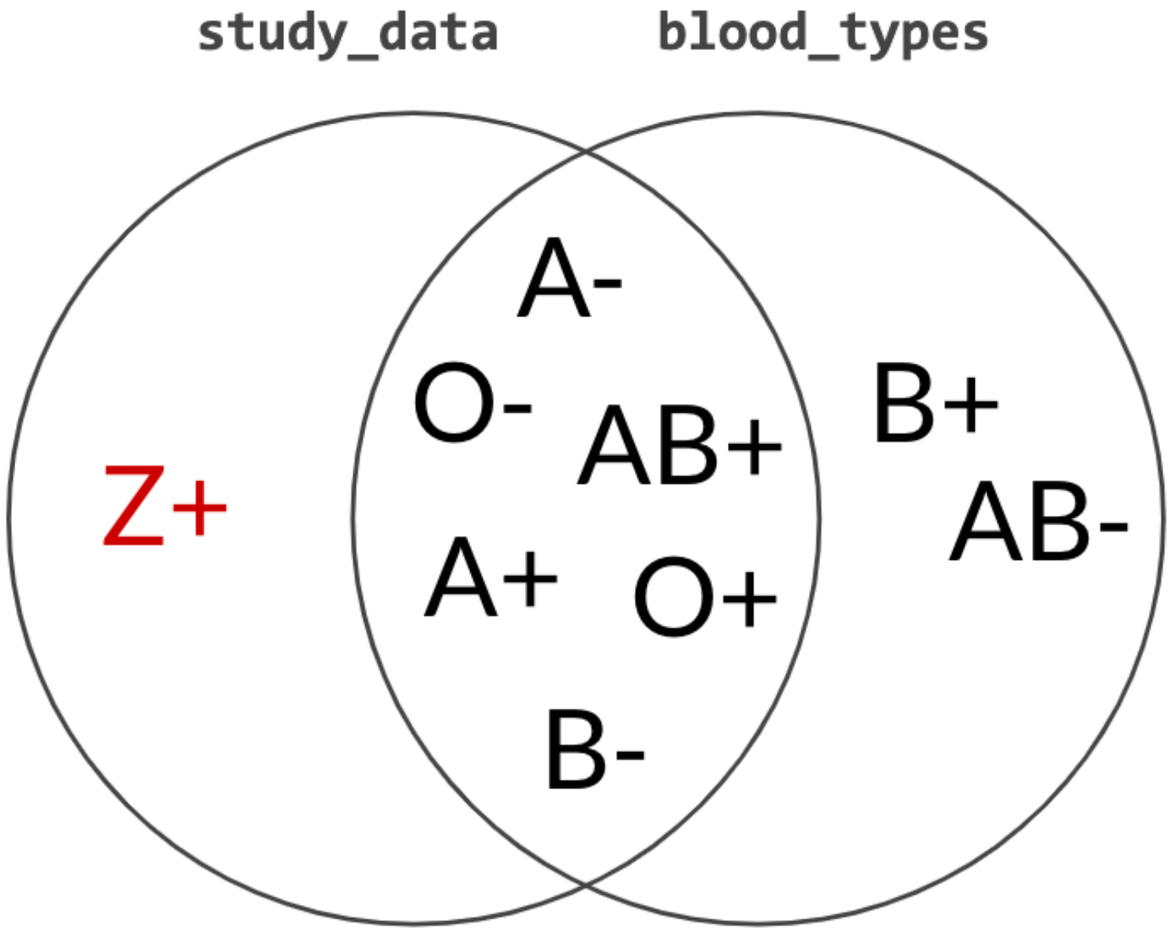
`study_data`

```
      name   birthday blood_type
1     Beth 2019-10-20          B-
2 Ignatius 2020-07-08          A-
3     Paul 2019-08-12          O+
4    Helen 2019-03-17          O-
5 Jennifer 2019-12-17          Z+  <--
6  Kennedy 2020-04-27          A+
7    Keith 2019-04-19         AB+
```

`blood_types`

```
  blood_type
1         O-
2         O+
3         A-
4         A+
5         B+
6         B-
7        AB+
8        AB-
```

Finding non-members

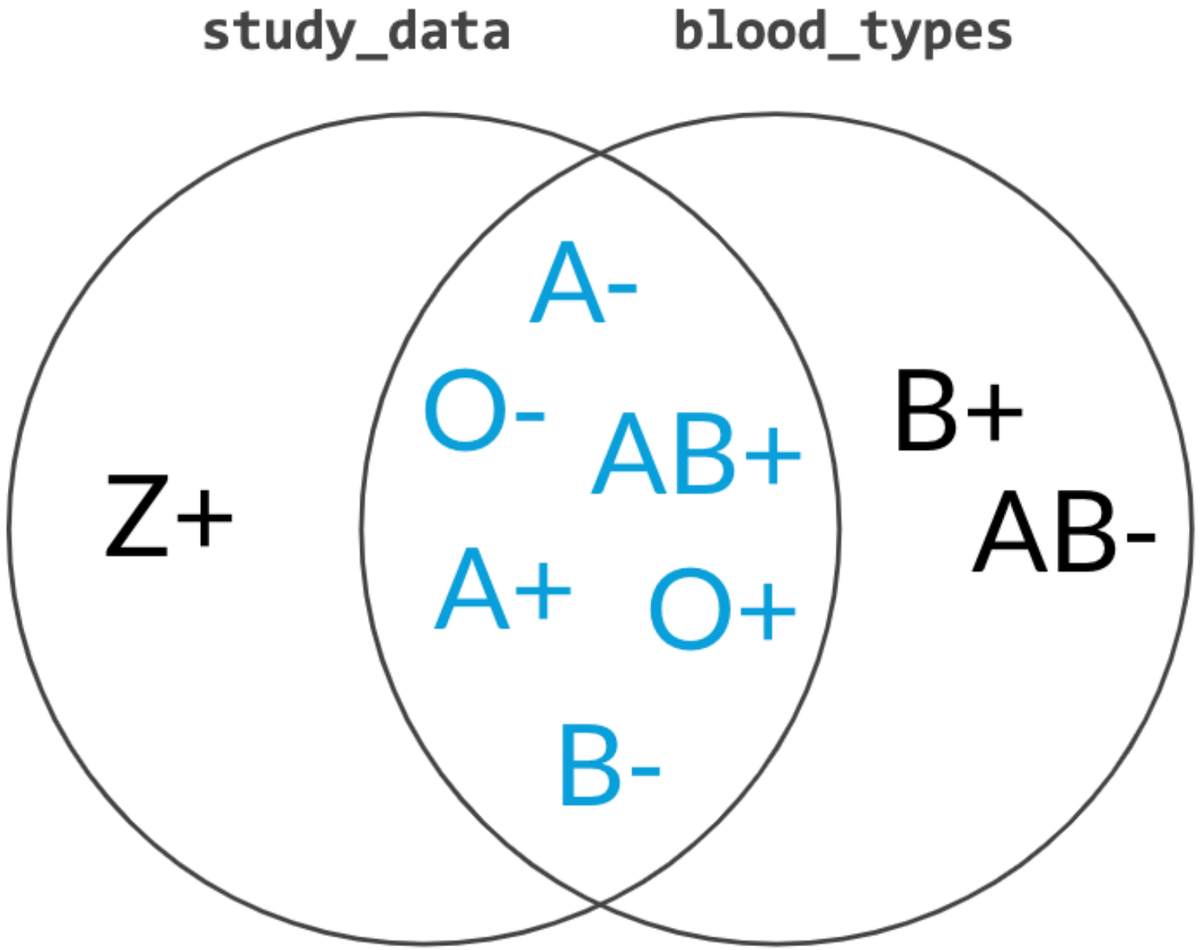


Anti-join

```
study_data %>%
  anti_join(blood_types, by = "blood_type")
```

```
      name   birthday blood_type
1 Jennifer 2019-12-17         Z+
```

Removing non-members



Semi-join

```
study_data %>%
  semi_join(blood_types, by = "blood_type")
```

```
      name   birthday blood_type
1     Beth 2019-10-20         B-
2 Ignatius 2020-07-08         A-
3     Paul 2019-08-12         O+
4    Helen 2019-03-17         O-
5  Kennedy 2020-04-27         A+
6    Keith 2019-04-19        AB+
```

Let's practice!

CLEANING DATA IN R

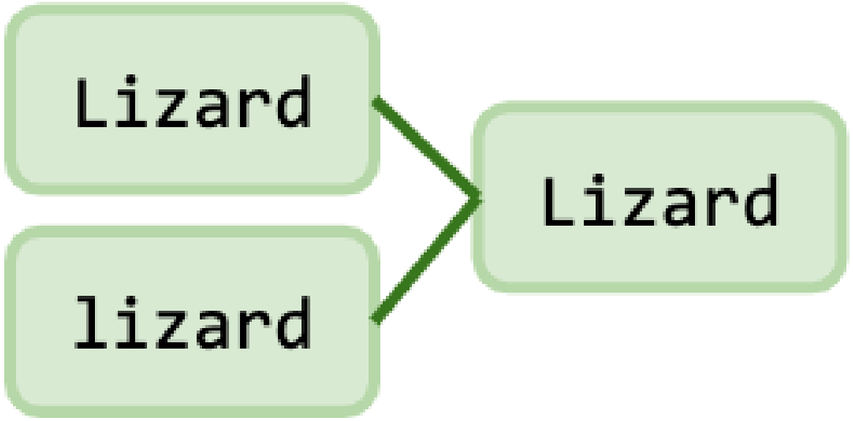
Categorical data problems

CLEANING DATA IN R

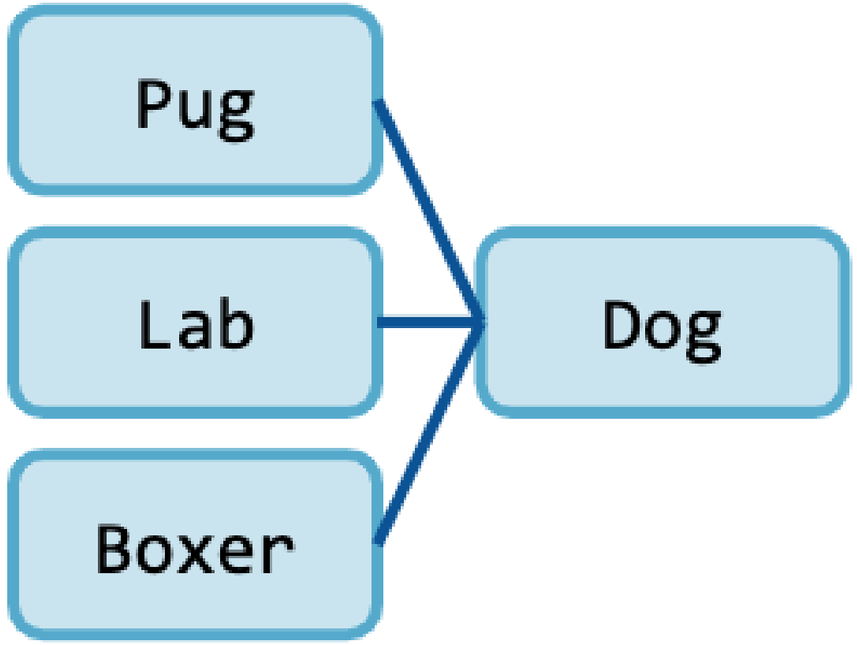
Maggie Matsui
Content Developer @ DataCamp

Categorical data problems

Inconsistency within a category



Too many categories



Example: animal classification

```
animals
```

```
# A tibble: 68 x 9
   animal_name hair  eggs  fins   legs tail  type
   <chr>       <fct> <fct> <fct> <int> <fct> <fct>
 1 mole        1     0     0         4 1     mammal
 2 chicken     0     1     0         2 1     bird
 3 capybara    1     0     0         2 1      Mammal
 4 tuna        0     1     1         0 1     fish
 5 ostrich     0     1     0         2 1     bird
# ... with 63 more rows
```

Checking categories

```
animals %>%
  count(type)
```

```
   type                n
 1 " mammal "          1
 2 "amphibian"         2
 3 "bird"             20
 4 "bug"               1
 5 "fish"              2
 6 "invertebrate"      1
 7 "mammal"           38
 8 "MAMMAL"            1
 9 "Mammal "           1
10 "reptile"           1
```

- `"mammal"`
- `" mammal "`
- `"MAMMAL"`
- `"Mammal "`

Case inconsistency

```
library(stringr)
animals %>%
  mutate(type_lower = str_to_lower(type))
```

```
   animal_name  hair  eggs  fins  legs  tail  type       type_lower
   <fct>       <int> <int>  <int> <int> <int> <fct>      <chr>
 1 mole            1     0     0     4     1  "mammal"   "mammal"
 2 chicken         0     1     0     2     1  "bird"     "bird"
 3 capybara        1     0     0     2     1  " Mammal"  " mammal"
 4 tuna            0     1     1     0     1  "fish"     "fish"
 5 ostrich         0     1     0     2     1  "bird"     "bird"
```

Case inconsistency

```
animals %>%
  mutate(type_lower = str_to_lower(type)) %>%
  count(type_lower)
```

```
  type_lower          n                 type_lower          n
  <chr>           <int>                 <chr>           <int>
1 " mammal "          1               6 "invertebrate"      1
2 "amphibian"         2               7 "mammal"           39
3 "bird"             20               8 "mammal "           1
4 "bug"               1               9 "reptile"           1
5 "fish"              2
```

`"MAMMAL"` → `"mammal"`

Case inconsistency

```
animals %>%
  mutate(type_upper = str_to_upper(type)) %>%
  count(type_upper)
```

```
  type_upper          n
  <chr>           <int>
1 " MAMMAL "          1
2 "AMPHIBIAN"         2
3 "BIRD"             20
4 "BUG"               1
5 "FISH"              2
  type_upper          n
  <chr>           <int>
6 "INVERTEBRATE"      1
7 "MAMMAL"           39
8 "MAMMAL "           1
9 "REPTILE"           1
```

Whitespace inconsistency

```
animals %>%
  mutate(type_trimmed = str_trim(type_lower))
```

```
  animal_name  hair  eggs  fins  legs  tail  type_lower type_trimmed
  <fct>       <int> <int>  <int> <int> <int> <chr>      <chr>
1 mole            1     0     0     4     1  "mammal"   mammal
2 chicken         0     1     0     2     1  "bird"     bird
3 capybara        1     0     0     2     1  " mammal"  mammal
4 tuna            0     1     1     0     1  "fish"     fish
5 ostrich         0     1     0     2     1  "bird"     bird
```

Whitespace inconsistency

```
animals %>%
  mutate(type_trimmed = str_trim(type_lower)) %>%
  count(type_trimmed)
```

```
  type_trimmed     n            type_trimmed     n
  <chr>        <int>            <chr>        <int>
1 amphibian        2          6 mammal          41
2 bird            20          7 reptile          1
3 bug              1
4 fish             2
5 invertebrate     1
```

Too many categories

```
animals %>%
  count(type_trimmed, sort = TRUE)
```

```
  type_trimmed     n
1 mammal          41
2 bird            20
3 amphibian        2
4 fish             2
5 bug              1
6 invertebrate     1
7 reptile          1
```

Collapsing categories

```
other_categories =  c("amphibian", "fish", "bug", "invertebrate", "reptile")
```

```
library(forcats)
animals %>%
  mutate(type_collapsed = fct_collapse(type_trimmed, other = other_categories))
```

```
  animal_name  hair  eggs  fins  legs  tail  type_trimmed  type_collapsed
  <fct>       <int> <int>  <int> <int> <int> <chr>         <chr>
1 mole            1     0     0     4     1  mammal        mammal
2 chicken         0     1     0     2     1  bird          bird
3 capybara        1     0     0     2     1  mammal        mammal
4 tuna            0     1     1     0     1  fish          other
5 ostrich         0     1     0     2     1  bird          bird
```

Collapsing categories

```
animals %>%
  count(type_collapsed)
```

```
  type_collapsed     n
  <fct>          <int>
1 other              7
2 bird              20
3 mammal            41
```

```
animals %>%
  group_by(type_collapsed) %>%
  summarize(avg_legs = mean(legs))
```

```
  type_collapsed avg_legs
  <fct>              <dbl>
1 other               3.71
2 bird                2
3 mammal              3.37
```

Let's practice!

CLEANING DATA IN R

Cleaning text data

CLEANING DATA IN R

Maggie Matsui
Content Developer @ DataCamp

What is text data?

Type of data	Example values
Names	`"Veronica Hopkins"`, `"Josiah"`, ...
Phone numbers	`"6171679912"`, `"(868) 949-4489"`, ...
Emails	`"vhopkins@datacamp.com"`, `"josiah@josiah.co.uk"`, ...
Passwords	`"JZY46TVG8SM"`, `"iamjosiah21"`, ...
Comments/Reviews	`"great service!"`, `"This product broke after 2 days"`, ...

Unstructured data problems

- Formatting inconsistency
  - `"6171679912"` vs. `"(868) 949-4489"`
  - `"9239 5849 3712 0039"` vs. `"4490459957881031"`
- Information inconsistency
  - `+1 617-167-9912` vs. `617-167-9912`
  - `"Veronica Hopkins"` vs. `"Josiah"`
- Invalid data
  - Phone number `"0492"` is too short
  - Zip code `"19888"` doesn't exist

Customer data

```
customers
```

```
# A tibble: 99 x 3
   name            company                    credit_card
   <chr>           <chr>                      <chr>
 1 Galena          In Magna Associates        5171 5854 8986 1916
 2 MacKenzie       Iaculis Ltd                5128-5078-8008-5824
 3 Megan Acosta    Semper LLC                 5502 4529 0732 1744
 4 Phoebe Delacruz Sit Amet Nulla Limited     5419-7308-7424-0944
 5 Jessica         Pellentesque Sed Ltd       5419 2949 5508 9530
# ... with 95 more rows
```

Detecting hyphenated credit card numbers

```
str_detect(customers$credit_card, "-")
```

```
FALSE TRUE FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE ...
```

```
customers %>%
  filter(str_detect(credit_card, "-"))
```

```
   name            company                  credit_card
 1 MacKenzie       Iaculis Ltd              5128-5078-8008-5824
 2 Phoebe Delacruz Sit Amet Nulla Limited   5419-7308-7424-0944
 3 Abel            Lorem PC                 5211-6023-0805-0217
 ...
```

Replacing hyphens

```
customers %>%
  mutate(credit_card_spaces = str_replace_all(credit_card, "-", " "))
```

```
   name             company                      credit_card_spaces
 1 Galena           In Magna Associates          5171 5854 8986 1916
 2 MacKenzie        Iaculis Ltd                  5128 5078 8008 5824
 3 Megan Acosta     Semper LLC                   5502 4529 0732 1744
 4 Phoebe Delacruz Sit Amet Nulla Limited        5419 7308 7424 0944
 5 Jessica          Pellentesque Sed Ltd         5419 2949 5508 9530
 ...
```

Removing hyphens and spaces

```
credit_card_clean <- customers$credit_card %>%
  str_remove_all("-") %>%
  str_remove_all(" ")
customers %>%
  mutate(credit_card = credit_card_clean)
```

```
   name            company                credit_card
 1 Galena          In Magna Associates    5171585489861916
 2 MacKenzie       Iaculis Ltd            5128507880085824
 3 Megan Acosta    Semper LLC             5502452907321744
 ...
```

Finding invalid credit cards

```
str_length(customers$credit_card)
```

```
16 16 16 16 16 16 16 16 16 16 16 16 12 16 16 16 16 16 16 16 16 16 16 16 ...
```

```
customers %>%
  filter(str_length(credit_card) != 16)
```

```
  name            company                      credit_card
1 Jerry Russell   Sed Eu Company               516294099537
2 Ivor Christian  Ut Tincidunt Incorporated    544571330015
3 Francesca Drake Etiam Consulting             517394144089
```

Removing invalid credit cards

```
customers %>%
  filter(str_length(credit_card) == 16)
```

```
   name             company                    credit_card
 1 Galena           In Magna Associates        5171585489861916
 2 MacKenzie        Iaculis Ltd                5128507880085824
 3 Megan Acosta     Semper LLC                 5502452907321744
 4 Phoebe Delacruz Sit Amet Nulla Limited      5419730874240944
 5 Jessica          Pellentesque Sed Ltd       5419294955089530
...
```

More complex text problems

- A ***regular expression*** is a sequence of characters that allows for robust searching within a string.
- Certain characters are treated differently in a regular expression:
  - `(`, `)`, `[`, `]`, `$`, `.`, `+`, `*`, and others
- `stringr` functions use regular expressions
- Searching for these characters requires using **`fixed()`**:
  - `str_detect(column, fixed("$"))`

Learn more in String Manipulation with stringr in R & Intermediate Regular Expressions in R

Let's practice!

CLEANING DATA IN R