Merging Data Sets

Data Wrangling in R

Joining in `dplyr`

- Merging/joining data sets together - usually on key variables, usually “id”
- `?join` - see different types of joining for `dplyr`
- `inner_join(x, y)` - only rows that match for `x` and `y` are kept
- `full_join(x, y)` - all rows of `x` and `y` are kept
- `left_join(x, y)` - all rows of `x` are kept even if not merged with `y`
- `right_join(x, y)` - all rows of `y` are kept even if not merged with `x`
- `anti_join(x, y)` - all rows from `x` not in `y` keeping just columns from `x`.

Merging: Simple Data

```
data_As
```

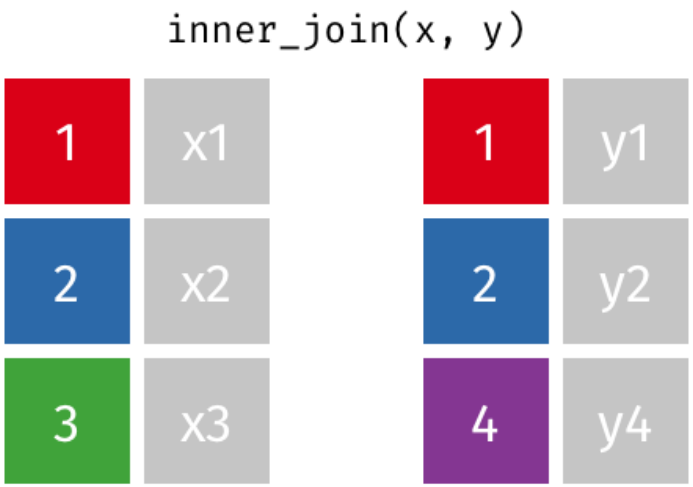
```
# A tibble: 2 × 3
  State   June_vacc_rate May_vacc_rate
  <chr>   <chr>          <chr>
1 Alabama 37.2%          36.0%
2 Alaska  47.5%          46.2%
```

```
data_cold
```

```
# A tibble: 2 × 2
  State  April_vacc_rate
  <chr>  <chr>
1 Maine  32.4%
2 Alaska 41.7%
```

Inner Join

https://github.com/gadenbuie/tidyexplain/blob/main/images/inner-join.gif



Inner Join

```
ij = inner_join(data_As, data_cold)
```

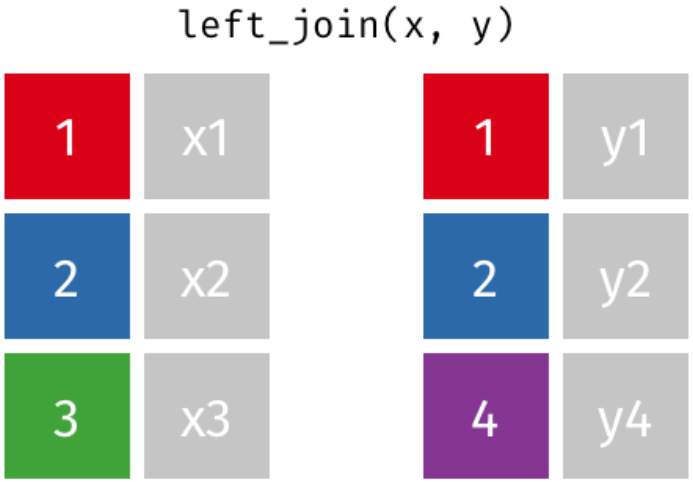
```
Joining, by = "State"
```

```
ij
```

```
# A tibble: 1 × 4
  State  June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>  <chr>          <chr>         <chr>
1 Alaska 47.5%          46.2%         41.7%
```

Left Join

https://raw.githubusercontent.com/gadenbuie/tidyexplain/main/images/left-join.gif



Left Join

```
lj = left_join(data_As, data_cold)
```

```
Joining, by = "State"
```

```
lj
```

```
# A tibble: 2 × 4
  State   June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>   <chr>          <chr>         <chr>
1 Alabama 37.2%          36.0%         <NA>
2 Alaska  47.5%          46.2%         41.7%
```

Install `tidylog` package to log outputs

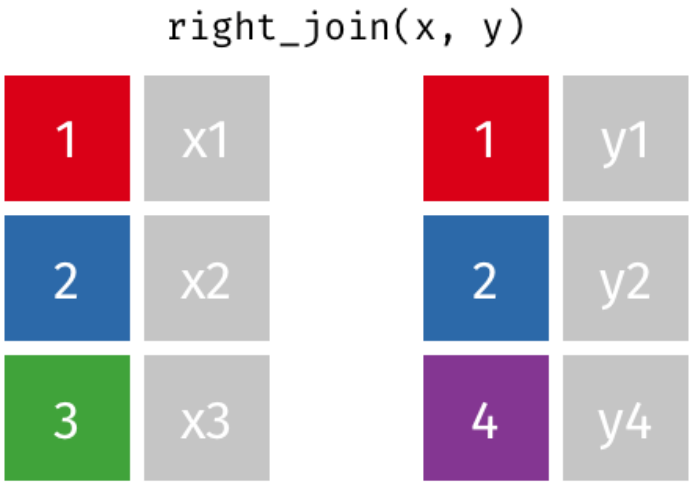
```
# install.packages("tidylog")
library(tidylog)
left_join(data_As, data_cold)
```

```
Joining, by = "State"
left_join: added one column (April_vacc_rate)
> rows only in x 1
> rows only in y (1)
> matched rows 1
> ===
> rows total 2

# A tibble: 2 × 4
  State   June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>   <chr>          <chr>         <chr>
1 Alabama 37.2%          36.0%         <NA>
2 Alaska  47.5%          46.2%         41.7%
```

Right Join

https://raw.githubusercontent.com/gadenbuie/tidyexplain/main/images/right-join.gif



Right Join

```
rj <- right_join(data_As, data_cold)
```

```
Joining, by = "State"
right_join: added one column (April_vacc_rate)
> rows only in x (1)
> rows only in y 1
> matched rows 1
> ===
> rows total 2
```

```
rj
```

```
# A tibble: 2 × 4
  State  June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>  <chr>          <chr>         <chr>
1 Alaska 47.5%          46.2%         41.7%
2 Maine  <NA>           <NA>          32.4%
```

Left Join: Switching arguments

```
lj2 <- left_join(data_cold, data_As)
```

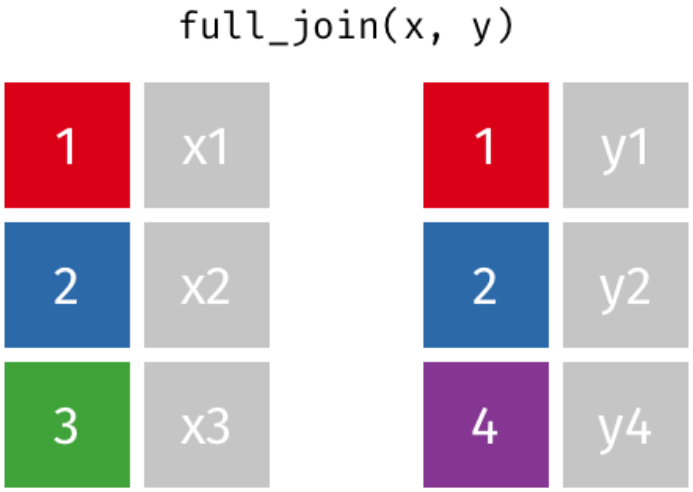
```
Joining, by = "State"
left_join: added 2 columns (June_vacc_rate, May_vacc_rate)
> rows only in x 1
> rows only in y (1)
> matched rows 1
> ===
> rows total 2
```

```
lj2
```

```
# A tibble: 2 × 4
  State  April_vacc_rate June_vacc_rate May_vacc_rate
  <chr>  <chr>           <chr>          <chr>
1 Maine  32.4%           <NA>           <NA>
2 Alaska 41.7%           47.5%          46.2%
```

Full Join

https://raw.githubusercontent.com/gadenbuie/tidyexplain/main/images/full-join.gif



Full Join

```
fj <- full_join(data_As, data_cold)
```

```
Joining, by = "State"
full_join: added one column (April_vacc_rate)
> rows only in x 1
> rows only in y 1
> matched rows 1
> ===
> rows total 3
```

```
fj
```

```
# A tibble: 3 × 4
  State   June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>   <chr>          <chr>         <chr>
1 Alabama 37.2%          36.0%         <NA>
2 Alaska  47.5%          46.2%         41.7%
3 Maine   <NA>           <NA>          32.4%
```

Watch out for "`includes duplicates`"

```
data_As
```

```
# A tibble: 2 × 2
  State   state_bird
  <chr>   <chr>
1 Alabama wild turkey
2 Alaska  willow ptarmigan
```

```
data_cold
```

```
# A tibble: 3 × 3
  State  vacc_rate month
  <chr>  <chr>     <chr>
1 Maine  32.4%     April
2 Alaska 41.7%     April
3 Alaska 46.2%     May
```

Watch out for "`includes duplicates`"

```
lj <- left_join(data_As, data_cold)
```

```
Joining, by = "State"
left_join: added 2 columns (vacc_rate, month)
> rows only in x 1
> rows only in y (1)
> matched rows 2 (includes duplicates)
> ===
> rows total 3
```

Watch out for "`includes duplicates`"

Data including the joining column ("State") has been duplicated.

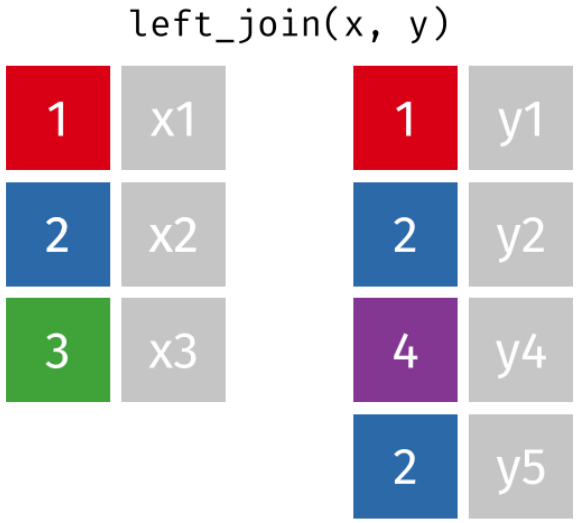
```
lj
```

```
# A tibble: 3 × 4
  State   state_bird       vacc_rate month
  <chr>   <chr>            <chr>     <chr>
1 Alabama wild turkey      <NA>      <NA>
2 Alaska  willow ptarmigan 41.7%     April
3 Alaska  willow ptarmigan 46.2%     May
```

Note that "Alaska willow ptarmigan" appears twice.

Watch out for "`includes duplicates`"

https://github.com/gadenbuie/tidyexplain/blob/main/images/left-join-extra.gif



Stop `tidylog`

```
unloadNamespace("tidylog")
```

Using the `by` argument

By default joins use the intersection of column names. If `by` is specified, it uses that.

```
full_join(data_As, data_cold, by = "State")
```

```
# A tibble: 4 × 4
  State   state_bird       vacc_rate month
  <chr>   <chr>            <chr>     <chr>
1 Alabama wild turkey      <NA>      <NA>
2 Alaska  willow ptarmigan 41.7%     April
3 Alaska  willow ptarmigan 46.2%     May
4 Maine   <NA>             32.4%     April
```

Using the `by` argument

You can join based on multiple columns by using something like `by = c(col1, col2)`.

If the datasets have two different names for the same data, use:

```
full_join(x, y, by = c("a" = "b"))
```

Using "`setdiff`"

We might want to determine what indexes ARE in the first dataset that AREN'T in the second:

```
data_As
```

```
# A tibble: 2 × 2
  State   state_bird
  <chr>   <chr>
1 Alabama wild turkey
2 Alaska  willow ptarmigan
```

```
data_cold
```

```
# A tibble: 3 × 3
  State  vacc_rate month
  <chr>  <chr>     <chr>
1 Maine  32.4%     April
2 Alaska 41.7%     April
3 Alaska 46.2%     May
```

Using "`setdiff`"

Use `setdiff` to determine what indexes ARE in the first dataset that AREN'T in the second:

```
A_states <- data_As %>% pull(State)
cold_states <- data_cold %>% pull(State)
```

```
setdiff(A_states, cold_states)
```

```
[1] "Alabama"
```

```
setdiff(cold_states, A_states)
```

```
[1] "Maine"
```

Using `bind_rows()` (`dplyr`)

```
one <- starwars[1:4, ]
two <- starwars[9:12, ]

bind_rows(one, two)
```

```
# A tibble: 8 × 14
  name      height  mass hair_color skin_color eye_color birth_year sex   genc
  <chr>      <int> <dbl> <chr>      <chr>      <chr>           <dbl> <chr> <chr
1 Luke Sky…    172    77 blond      fair       blue             19   male  masc
2 C-3PO        167    75 <NA>       gold       yellow          112   none  masc
3 R2-D2         96    32 <NA>       white, bl… red              33   none  masc
4 Darth Va…    202   136 none       white      yellow           41.9 male  masc
5 Biggs Da…    183    84 black      light      brown            24   male  masc
6 Obi-Wan …    182    77 auburn, w… fair       blue-gray        57   male  masc
7 Anakin S…    188    84 blond      fair       blue             41.9 male  masc
8 Wilhuff …    180    NA auburn, g… fair       blue             64   male  masc
# … with 5 more variables: homeworld <chr>, species <chr>, films <list>,
#   vehicles <list>, starships <list>
```

Summary

- Merging/joining data sets together - assumes all column names that overlap
  - use the `by = c("a" = "b")` if they differ
- `inner_join(x, y)` - only rows that match for `x` and `y` are kept
- `full_join(x, y)` - all rows of `x` and `y` are kept
- `left_join(x, y)` - all rows of `x` are kept even if not merged with `y`
- `right_join(x, y)` - all rows of `y` are kept even if not merged with `x`
- Use the `tidylog` package for a detailed summary
- `setdiff(x, y)` shows what in `x` is missing from `y`
- `bind_rows(x, y)` appends datasets