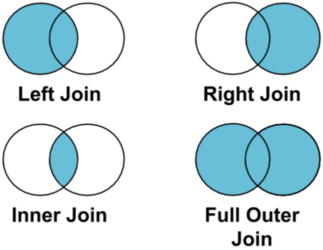
Merging and Joining Data Sets

Data Wrangling in R

Joining

"Combining datasets"



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 x 3
  State   June_vacc_rate May_vacc_rate
  <chr>            <dbl>         <dbl>
1 Alabama          0.516         0.514
2 Alaska           0.627         0.626
```

```
data_cold
```

```
# A tibble: 3 x 2
  State   April_vacc_rate
  <chr>             <dbl>
1 Maine             0.795
2 Alaska            0.623
3 Vermont           0.82
```

Inner Join

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

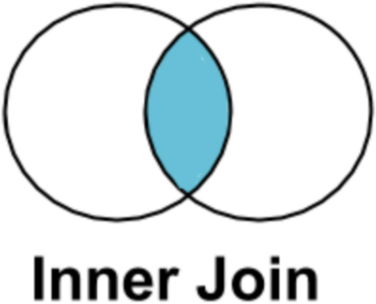
Inner Join

```
data_As
```

```
# A tibble: 2 × 3
  State   June_vacc_rate May_vacc_rate
  <chr>            <dbl>         <dbl>
1 Alabama          0.516         0.514
2 Alaska           0.627         0.626
```

```
data_cold
```

```
# A tibble: 3 × 2
  State   April_vacc_rate
  <chr>             <dbl>
1 Maine             0.795
2 Alaska            0.623
3 Vermont           0.82
```



Inner Join

```r
lj <- inner_join(data_As, data_cold)
```

```
Joining with `by = join_by(State)`
```

```r
lj
```

```
# A tibble: 1 x 4
  State  June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>           <dbl>         <dbl>           <dbl>
1 Alaska          0.627         0.626           0.623
```

Left Join

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

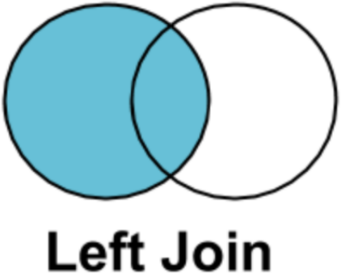
Left Join

```
data_As
```

```
# A tibble: 2 × 3
  State   June_vacc_rate May_vacc_rate
  <chr>            <dbl>         <dbl>
1 Alabama          0.516         0.514
2 Alaska           0.627         0.626
```

```
data_cold
```

```
# A tibble: 3 × 2
  State   April_vacc_rate
  <chr>             <dbl>
1 Maine             0.795
2 Alaska            0.623
3 Vermont           0.82
```



Left Join

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

```
Joining with `by = join_by(State)`
```

```
lj
```

```
# A tibble: 2 x 4
  State   June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>            <dbl>         <dbl>           <dbl>
1 Alabama          0.516         0.514          NA    
2 Alaska           0.627         0.626           0.623
```

Install tidylog package to log outputs

Numbers in parentheses indicate that these rows are not included in the result.

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

```
Joining with `by = join_by(State)`
left_join: added one column (April_vacc_rate)
> rows only in data_As 1
> rows only in data_cold (2)
> matched rows 1
> ===
> rows total 2
```

```
# A tibble: 2 x 4
  State   June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>            <dbl>         <dbl>           <dbl>
1 Alabama          0.516         0.514          NA    
2 Alaska           0.627         0.626           0.623
```

Right Join

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

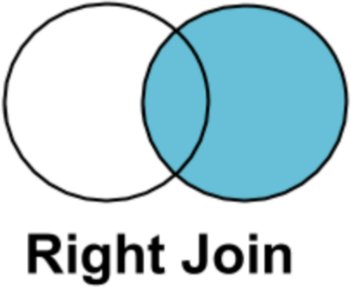
Right Join

```
data_As
```

```
# A tibble: 2 × 3
  State   June_vacc_rate May_vacc_rate
  <chr>            <dbl>         <dbl>
1 Alabama          0.516         0.514
2 Alaska           0.627         0.626
```

```
data_cold
```

```
# A tibble: 3 × 2
  State   April_vacc_rate
  <chr>             <dbl>
1 Maine             0.795
2 Alaska            0.623
3 Vermont           0.82
```



Right Join

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

```
Joining with `by = join_by(State)`
right_join: added one column (April_vacc_rate)
> rows only in data_As (1)
> rows only in data_cold 2
> matched rows 1
> ===
> rows total 3
```

```r
rj
```

```
# A tibble: 3 x 4
  State   June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>            <dbl>         <dbl>           <dbl>
1 Alaska           0.627         0.626           0.623
2 Maine           NA            NA               0.795
3 Vermont         NA            NA               0.82
```

Left Join: Switching arguments

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

```
Joining with `by = join_by(State)`
left_join: added 2 columns (June_vacc_rate, May_vacc_rate)
> rows only in data_cold 2
> rows only in data_As (1)
> matched rows 1
> ===
> rows total 3
```

```
lj2
```

```
# A tibble: 3 x 4
  State    April_vacc_rate June_vacc_rate May_vacc_rate
  <chr>              <dbl>          <dbl>         <dbl>
1 Maine              0.795         NA            NA
2 Alaska             0.623          0.627         0.626
3 Vermont            0.82          NA            NA
```

Full Join

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

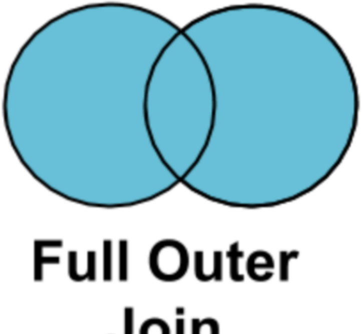
Full Join

```
data_As
```

```
# A tibble: 2 × 3
  State   June_vacc_rate May_vacc_rate
  <chr>            <dbl>         <dbl>
1 Alabama          0.516         0.514
2 Alaska           0.627         0.626
```

```
data_cold
```

```
# A tibble: 3 × 2
  State   April_vacc_rate
  <chr>             <dbl>
1 Maine             0.795
2 Alaska            0.623
3 Vermont           0.82
```



Full Join

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

```
Joining with `by = join_by(State)`
full_join: added one column (April_vacc_rate)
> rows only in data_As 1
> rows only in data_cold 2
> matched rows 1
> ===
> rows total 4
```

```r
fj
```

```
# A tibble: 4 x 4
  State    June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>             <dbl>         <dbl>           <dbl>
1 Alabama           0.516         0.514          NA
2 Alaska            0.627         0.626           0.623
3 Maine            NA            NA               0.795
4 Vermont          NA            NA               0.82
```

Watch out for “`includes duplicates`”

```r
data_As
```

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

```r
data_cold
```

```
# A tibble: 3 x 3
  State  vacc_rate month
  <chr>      <dbl> <chr>
1 Maine      0.795 April
2 Alaska     0.623 April
3 Alaska     0.626 May
```

Watch out for “includes duplicates”

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

```
Joining with `by = join_by(State)`
left_join: added 2 columns (vacc_rate, month)
> rows only in data_As 1
> rows only in data_cold (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 x 4
  State   state_bird       vacc_rate month
  <chr>   <chr>                <dbl> <chr>
1 Alabama wild turkey         NA     <NA>
2 Alaska  willow ptarmigan     0.623 April
3 Alaska  willow ptarmigan     0.626 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

Regular left join

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

Stop tidylog

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

Using the by argument

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

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

```
# A tibble: 4 x 4
  State   state_bird       vacc_rate month
  <chr>   <chr>                <dbl> <chr>
1 Alabama wild turkey         NA     <NA>
2 Alaska  willow ptarmigan     0.623 April
3 Alaska  willow ptarmigan     0.626 May
4 Maine   <NA>                 0.795 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” (base)

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

```
data_As
```

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

```
data_cold
```

```
# A tibble: 3 x 3
  State  vacc_rate month
  <chr>      <dbl> <chr>
1 Maine      0.795 April
2 Alaska     0.623 April
3 Alaska     0.626 May
```

Using “setdiff” (base)

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)

Rows are stacked on top of each other. Works like rbind() from base R, but is “smarter” and looks for matching column names.

```
rbind(data_As, data_cold)
```

```
Error in rbind(deparse.level, ...): numbers of columns of a
```

```
bind_rows(data_As, data_cold)
```

```
# A tibble: 5 x 4
  State   state_bird       vacc_rate month
  <chr>   <chr>                <dbl> <chr>
1 Alabama wild turkey         NA     <NA>
2 Alaska  willow ptarmigan    NA     <NA>
3 Maine   <NA>                 0.795 April
4 Alaska  <NA>                 0.623 April
5 Alaska  <NA>                 0.626 May
```

bind_rows vs full join

```r
full_join(data_As, data_cold)
```

```
Joining with `by = join_by(State)`
```

```
# A tibble: 4 x 4
  State   state_bird       vacc_rate month
  <chr>   <chr>                <dbl> <chr>
1 Alabama wild turkey         NA     <NA>
2 Alaska  willow ptarmigan     0.623 April
3 Alaska  willow ptarmigan     0.626 May
4 Maine   <NA>                 0.795 April
```

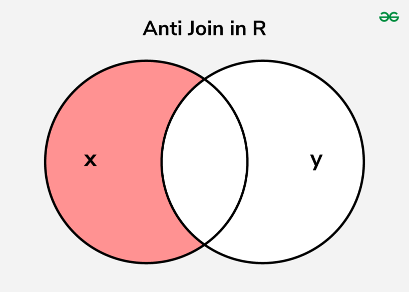
Other stuff: anti_join (dplyr)

```
data_As
```

```
# A tibble: 2 × 3
  State   June_vacc_rate May_vacc_rate
  <chr>            <dbl>         <dbl>
1 Alabama          0.516         0.514
2 Alaska           0.627         0.626
```

```
data_cold
```

```
# A tibble: 3 × 2
  State   April_vacc_rate
  <chr>             <dbl>
1 Maine             0.795
2 Alaska            0.623
3 Vermont           0.82
```



source

anti_join()

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

Other stuff: anti_join (dplyr)

```r
anti_join(data_As, data_cold)
```

```
Joining with `by = join_by(State)`

# A tibble: 1 x 2
  State   state_bird
  <chr>   <chr>
1 Alabama wild turkey
```

Other stuff: anti_join (dplyr)

```
anti_join(data_cold, data_As)
```

```
Joining with `by = join_by(State)`
```

```
# A tibble: 1 x 3
  State vacc_rate month
  <chr>     <dbl> <chr>
1 Maine     0.795 April
```

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

Extra slides

Other stuff: cross_join (dplyr)

Cross joins match each row in x to every row in y, resulting in a data frame with nrow(x) * nrow(y) rows.

```
cross_join(data_As, data_cold)
```

```
# A tibble: 6 x 5
  State.x state_bird        State.y vacc_rate month
  <chr>   <chr>             <chr>       <dbl> <chr>
1 Alabama wild turkey       Maine       0.795 April
2 Alabama wild turkey       Alaska      0.623 April
3 Alabama wild turkey       Alaska      0.626 May
4 Alaska  willow ptarmigan Maine       0.795 April
5 Alaska  willow ptarmigan Alaska      0.623 April
6 Alaska  willow ptarmigan Alaska      0.626 May
```

Other stuff: nest_join (dplyr)

A nest join leaves x almost unchanged, except that it adds a new column for the y dataset. Matched values are stored inside the "cell" as a tibble.

```
nj <- nest_join(data_As, data_cold)
```

```
Joining with `by = join_by(State)`
```

```
nj
```

```
# A tibble: 2 x 3
  State   state_bird       data_cold
  <chr>   <chr>            <list>
1 Alabama wild turkey      <tibble [0 x 2]>
2 Alaska  willow ptarmigan <tibble [2 x 2]>
```

Other stuff: nest_join (dplyr)

```r
nj %>% pull(data_cold)
```

```
[[1]]
# A tibble: 0 x 2
# i 2 variables: vacc_rate <dbl>, month <chr>

[[2]]
# A tibble: 2 x 2
  vacc_rate month
      <dbl> <chr>
1     0.623 April
2     0.626 May
```

“Includes duplicates” with both datasets duplicated:

```
data_As <- tibble(State = c("Alabama", "Alaska", "Alaska"),
                 state_bird = c("wild turkey", "willow ptar
data_cold <- tibble(State = c("Maine", "Alaska", "Alaska"),
                    vacc_rate = c("32.4%", "41.7%", "46.2%"
                    month = c("April", "April", "May"))
```

“Includes duplicates” with both datasets duplicated:

```r
full_join(data_As, data_cold)
```

```
Joining with `by = join_by(State)`

Warning in full_join(data_As, data_cold): Detected an unexp
i Row 2 of `x` matches multiple rows in `y`.
i Row 2 of `y` matches multiple rows in `x`.
i If a many-to-many relationship is expected, set `relatio
  "many-to-many"` to silence this warning.

# A tibble: 6 x 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 Alaska  puffin           41.7%     April
5 Alaska  puffin           46.2%     May
6 Maine   <NA>             32.4%     April
```