R Programming For Natural Resource Professionals

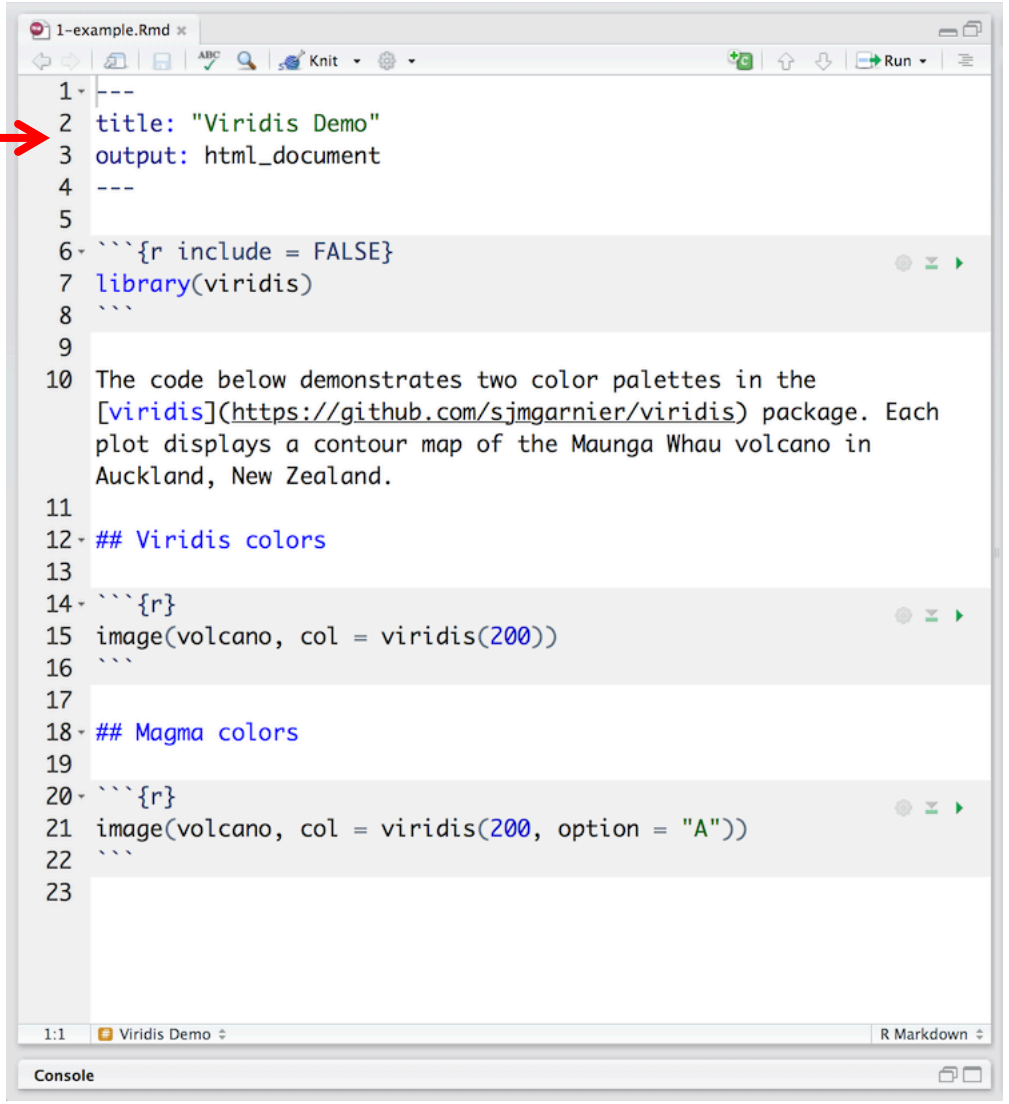
Lecture 5
Data Wrangling II:
Joining and advanced dplyr

Paper discussions

- Open this link now by typing it into a browser: https://bit.ly/3GVPwuV
- Talk with your group to identify one or two thoughts, questions, and epiphanies that resonate then record them in the Google Doc.
- Read through the list as it updates.

Polishing R markdowns

YAML header



YAML Header

- "Yet another markup language"
- High level formatting such as font size, figure size, title, subtitles, etc.
- Specify outputs:

```
output: pdf_document
output: word_document
output: rtf_document
output: md_document
```

Polishing R markdowns

Updated criteria for R markdown documents

1) Display resulting tibbles/tables using kableExtra

```
tib %>% kbl() %>% kable_styling()
```

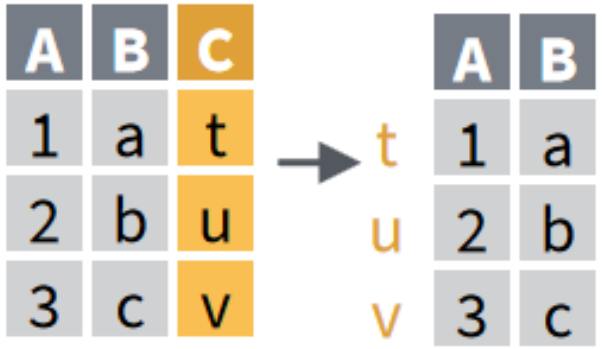
Consult excellent kableExtra vignette for options

2) Final document emphasizes code and result. No messages, errors, etc.

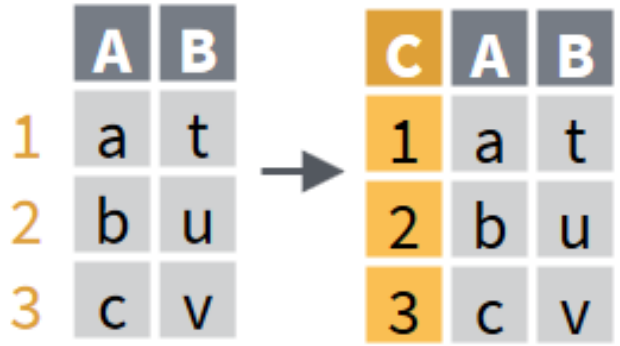
3) Goal is to generate publication quality documents

4) Get creative and have fun with it.

Working with row names



`tibble::column_to_rownames()`



`tibble::rownames_to_column()`

Adding new data

`dplyr::mutate`

Add a new variable or change an existing variable

`mutate(newVar = [calculation])`

Adding new data

`dplyr::transmute`

Compute a new variable while dropping the others

`transmute(newVar = [calculation])`

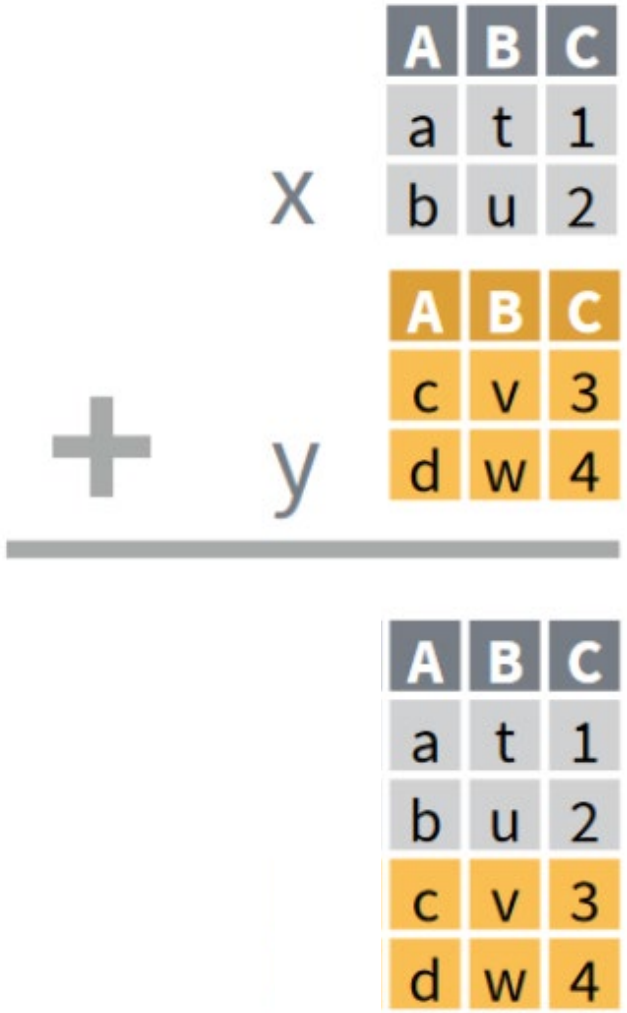
Rename variables

`dplyr::rename`

Change the name of a variable

```
rename(newName = oldName)
```

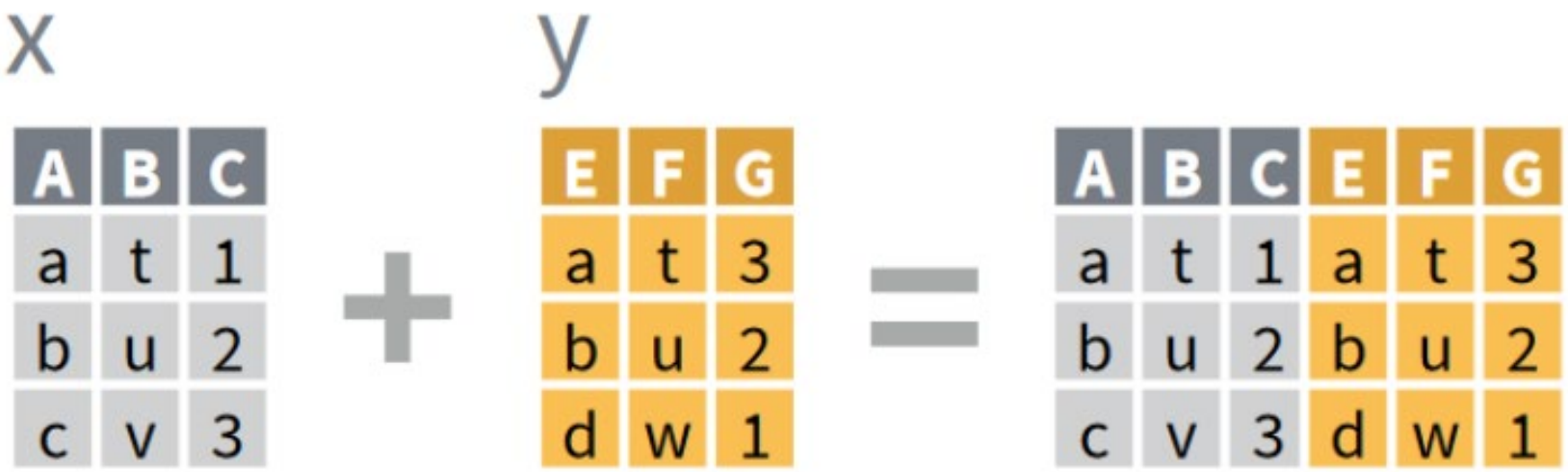
Combining data sets



`bind_rows()`

Returns one tibble pasted above the other

Combining data sets



`bind_cols()`

Returns one tibble pasted next to the other.

Important: Not for combining tibbles with the same variables!

Joining data sets

- Merge data based on given criteria
  - “Relational merging”

Tree	Height
Maple	80
Spruce	57
Oak	121
Oak	109
Maple	92

+

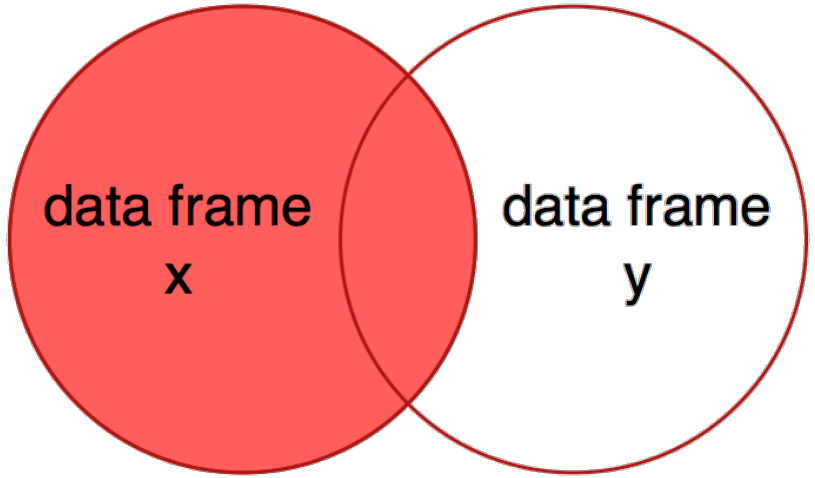
Tree	SciName
Maple	*Acer rubrum*
Spruce	*Picea pungens*
Oak	*Quercus alba*

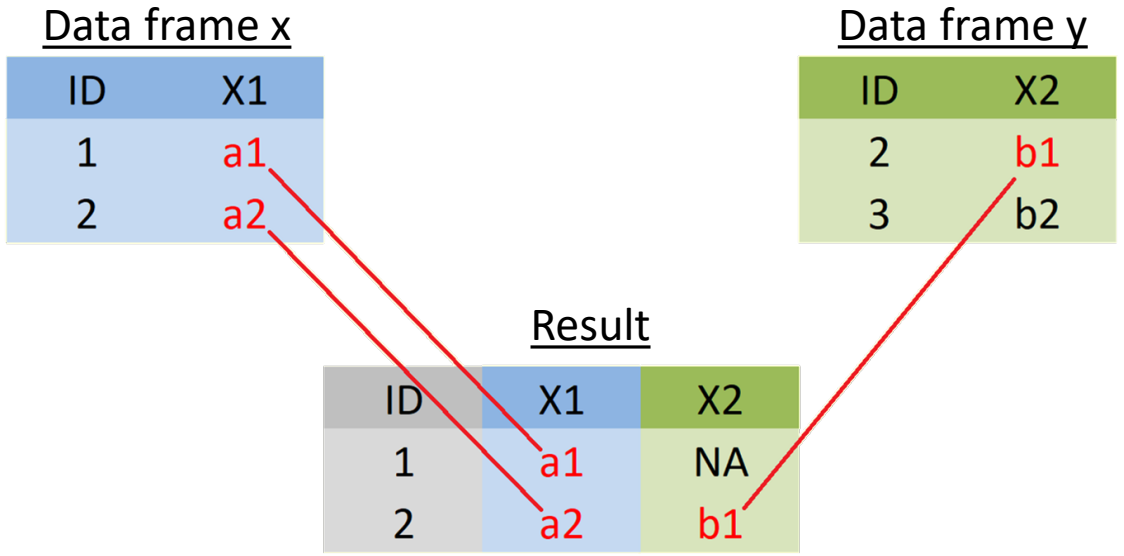
=

Tree	Height	SciName
Maple	80	*Acer rubrum*
Spruce	57	*Picea pungens*
Oak	121	*Quercus alba*
Oak	109	*Quercus alba*
Maple	92	*Acer rubrum*

Combining data sets

`left_join()`

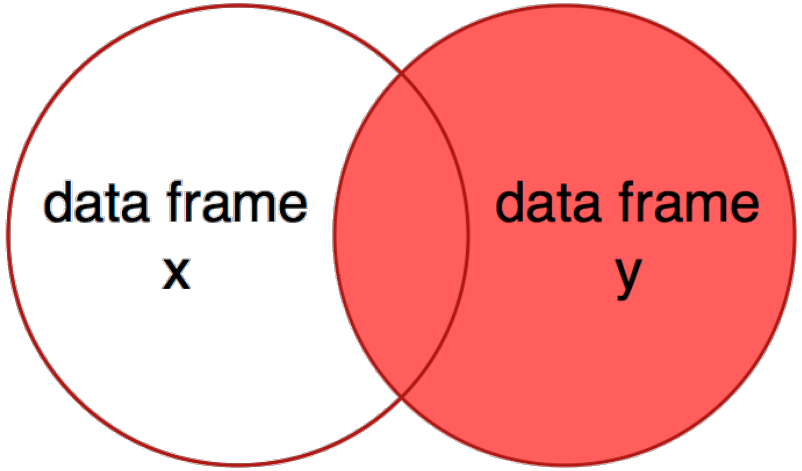


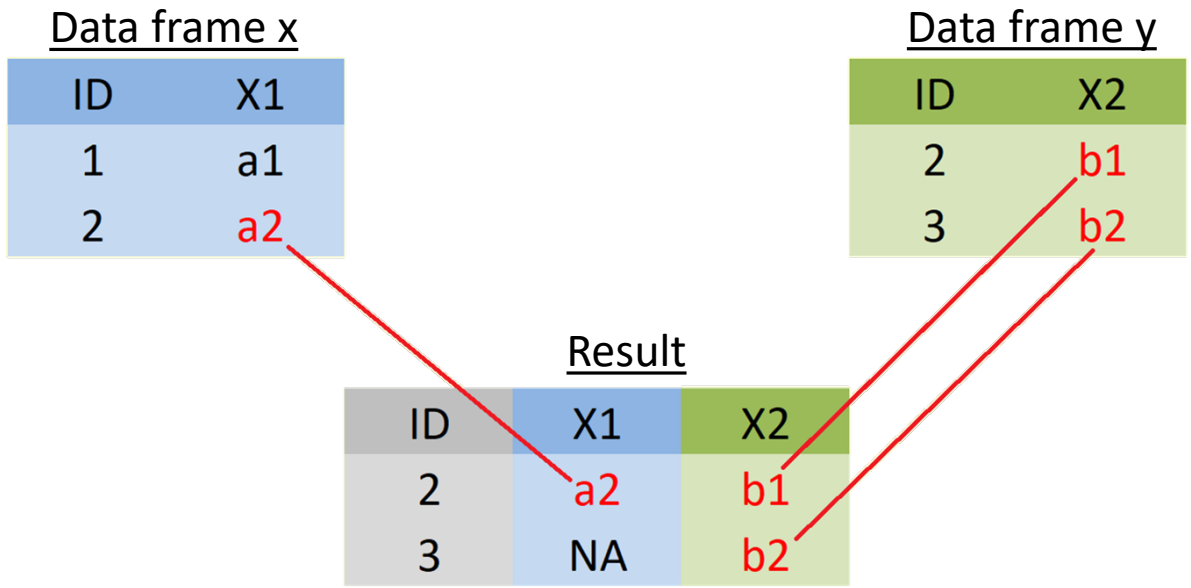


- Key variable in example is “ID”
- Returns all observations from x and all variables from x and y.
- Observations in x with no match in y will be populated with NAs

Combining data sets

`right_join()`

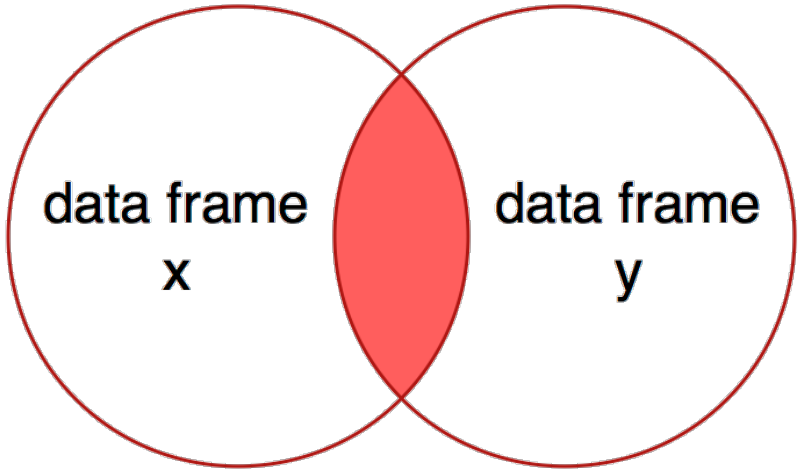


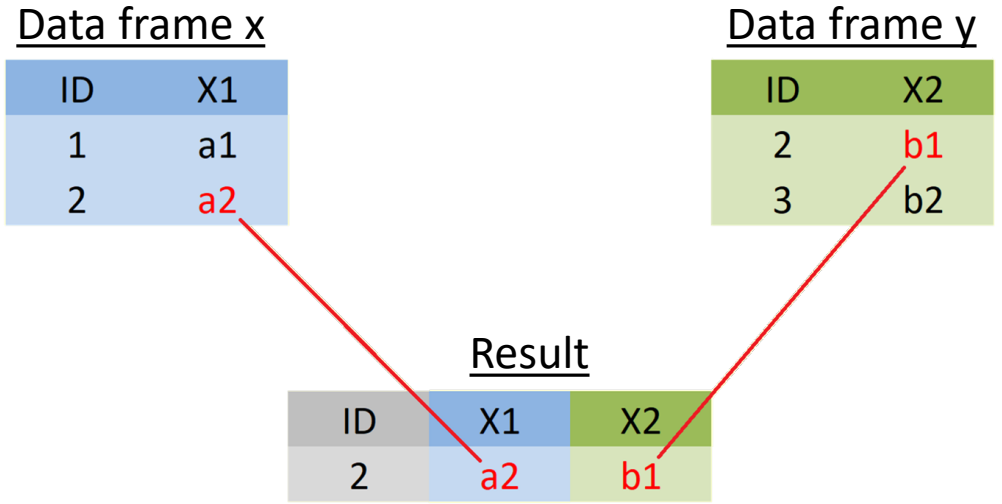


- Key variable in example is “ID”
- Returns all observations from y and all variables from x and y.
- Observations in y with no match in x will be populated with NAs

Combining data sets

`inner_join()`





Key variable in example is “ID”

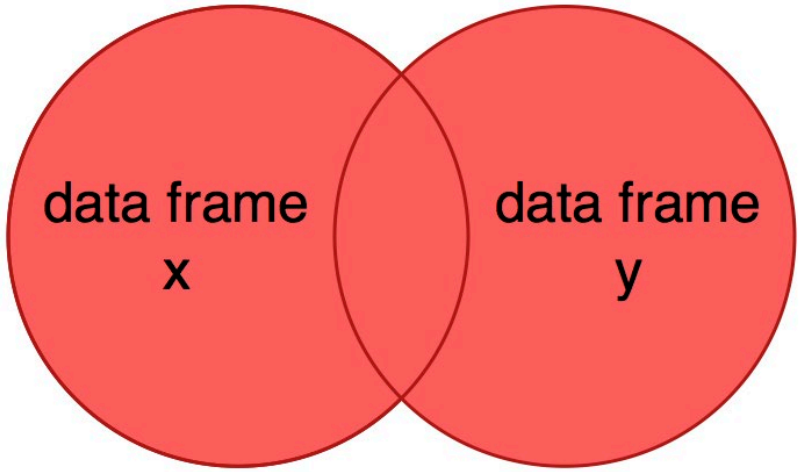
Returns all observations from x with matching observations in y

Returns all columns of x and y

If there are multiple matches between x and y, all combinations are returned

Combining data sets

`full_join()`



Data frame x

ID	X1
1	a1
2	a2

Data frame y

ID	X2
2	b1
3	b2

Result

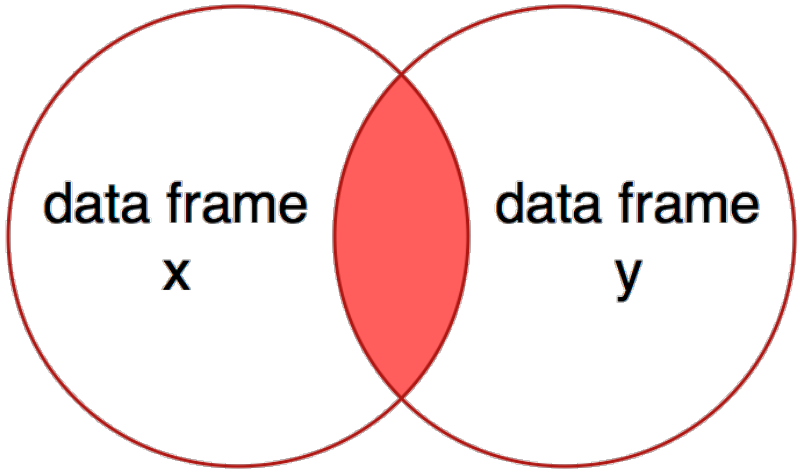
ID	X1	X2
1	a1	NA
2	a2	b1
3	NA	b2

Returns all observations and variables of both x and y

When not matching, returns NA

Combining data sets

`semi_join()`



Data frame x

ID	X1
1	a1
2	a2

Data frame y

ID	X2
2	b1
3	b2

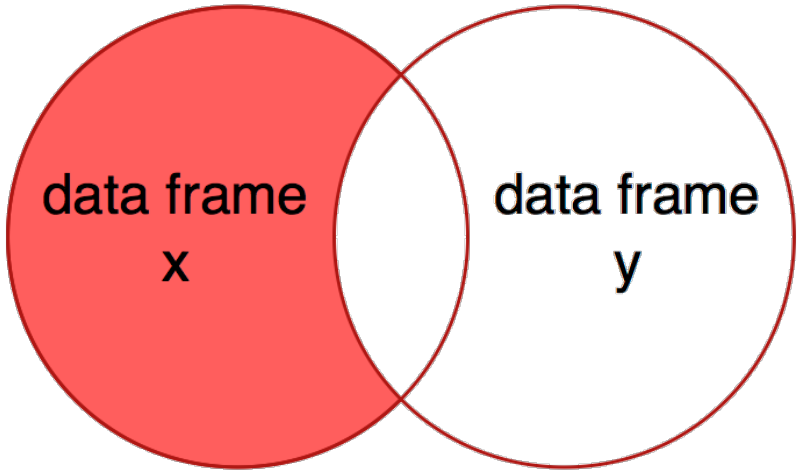
Result

ID	X1
2	a2

- A type of “filtering join”
- Returns observations from x that **do** also occur in y.
- Differs from inner_join() because it doesn’t retain variables in y

Combining data sets

anti_join()



Data frame x

ID	X1
1	a1
2	a2

Data frame y

ID	X2
2	b1
3	b2

Result

ID	X1
1	a1

- A type of “filtering join”
- Returns observations from x that **do not** also occur in y.

Combining data sets

- What if the key variable isn't the same in x and y?

```
tib1 %>%
   left_join(tib2, by = c("fish" = "species"))
```

Tib1

fish	length
Brook trout	12
Walleye	15
Walleye	16

Tib2

species	genus_species
Brook trout	*Salvelinus fontinalis*
Walleye	*Sander vitreus*

Create new variables using conditional statements