MANIPULATING TABLES

Adelaide Code Club

16/06/2022

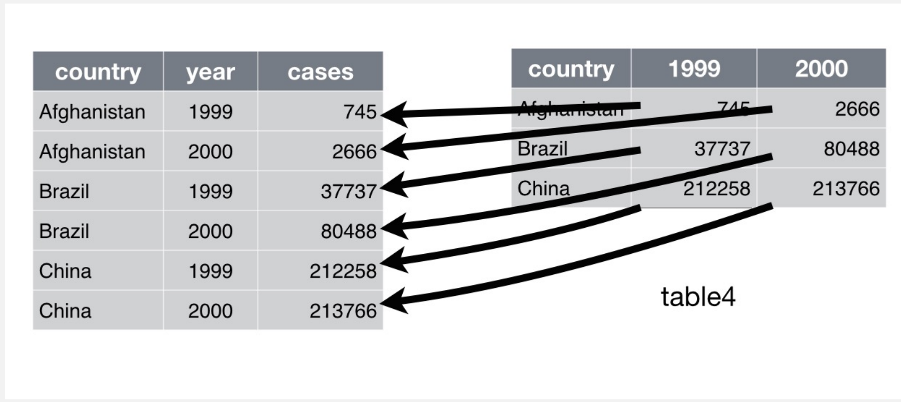
OUTLINE FOR TODAY

- Recap of last week
- A few more join functions
- Comparing similar datasets
- Extracting values

RECAP

- **pivot_longer(**
  x = dataframe,
  cols= columns to make longer,
  names_to=n ew column name,
  values_to = name of new column for values)

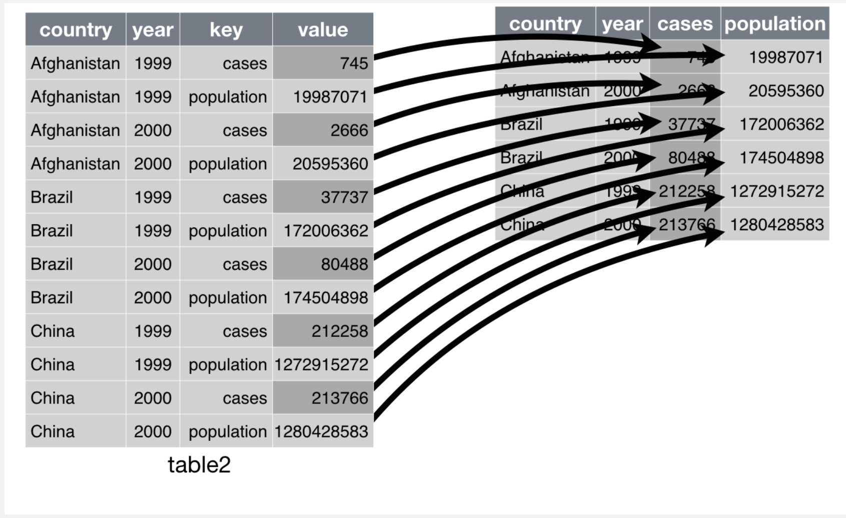
- **pivot_wider(**
  x= dataframe,
  names_from= name of column to widen,
  values_from= name of column with values)



country	year	cases
Afghanistan	1999	745
Afghanistan	2000	2666
Brazil	1999	37737
Brazil	2000	80488
China	1999	212258
China	2000	213766

country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

table4



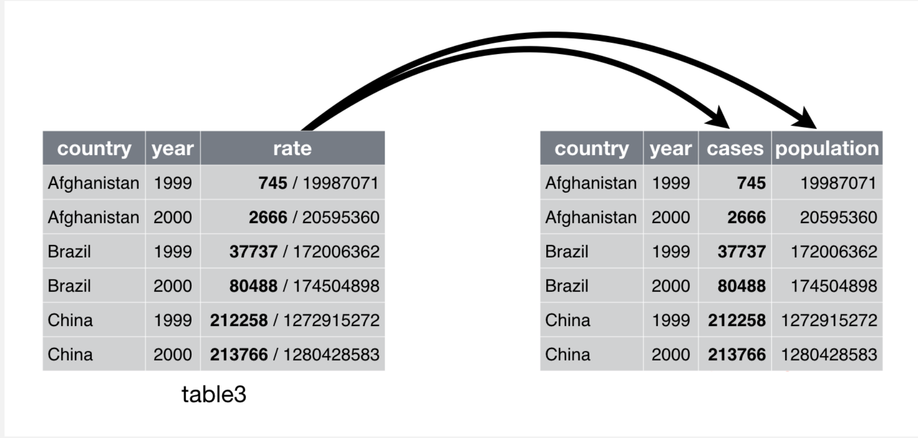
country	year	key	value
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272
China	2000	cases	213766
China	2000	population	1280428583

table2

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

RECAP

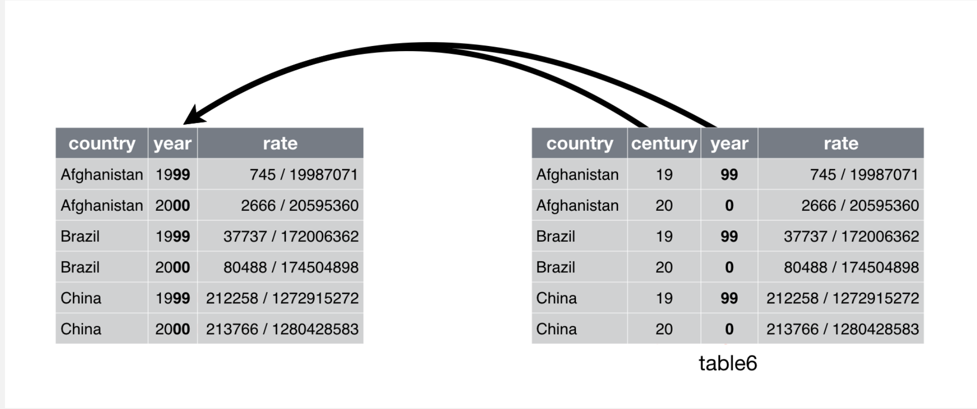
- **separate(**
  col= name of column,
  into= name of new columns,
  sep= separatpr character,
  convert= TRUE/FALSE as to whether to leave as character or change to integer)
- **unite(**
  col= name of new column,
  …= name of columns to join,
  sep= delimiter to pit between columns)



country	year	rate
Afghanistan	1999	745 / 19987071
Afghanistan	2000	2666 / 20595360
Brazil	1999	37737 / 172006362
Brazil	2000	80488 / 174504898
China	1999	212258 / 1272915272
China	2000	213766 / 1280428583

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

table3



country	year	rate
Afghanistan	1999	745 / 19987071
Afghanistan	2000	2666 / 20595360
Brazil	1999	37737 / 172006362
Brazil	2000	80488 / 174504898
China	1999	212258 / 1272915272
China	2000	213766 / 1280428583

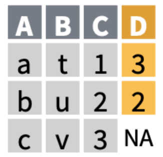
country	century	year	rate
Afghanistan	19	99	745 / 19987071
Afghanistan	20	0	2666 / 20595360
Brazil	19	99	37737 / 172006362
Brazil	20	0	80488 / 174504898
China	19	99	212258 / 1272915272
China	20	0	213766 / 1280428583

table6

RECAP

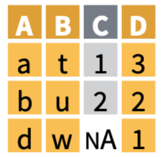
-
  ```
  *_join(
  x= df_1,
  y= df_2.
  by=col_name or c(col_1, col_2) or
  by=(x = grouping_var_x & y =
  grouping_var_y) if x and y cols have different names
  )
  ```

left_join()



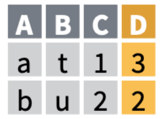
A	B	C	D
a	t	1	3
b	u	2	2
c	v	3	NA

right_join()



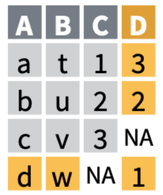
A	B	C	D
a	t	1	3
b	u	2	2
d	w	NA	1

inner_join()



A	B	C	D
a	t	1	3
b	u	2	2

full_join()



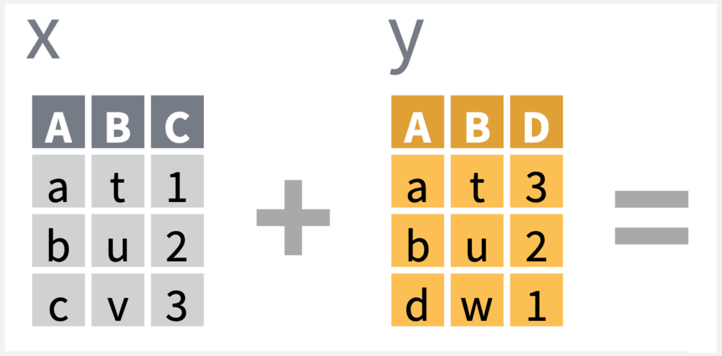
A	B	C	D
a	t	1	3
b	u	2	2
c	v	3	NA
d	w	NA	1

MANIPULATING DATASETS

- You may have two of the same dataset from different sources
- These might have small differences that need to be reconciled to assimilate into one data frame

A FEW MORE JOIN FUNCTIONS

- There are 2 more join functions that can be useful
  - to check a join is going to go how you want.
  - Syntax is the same

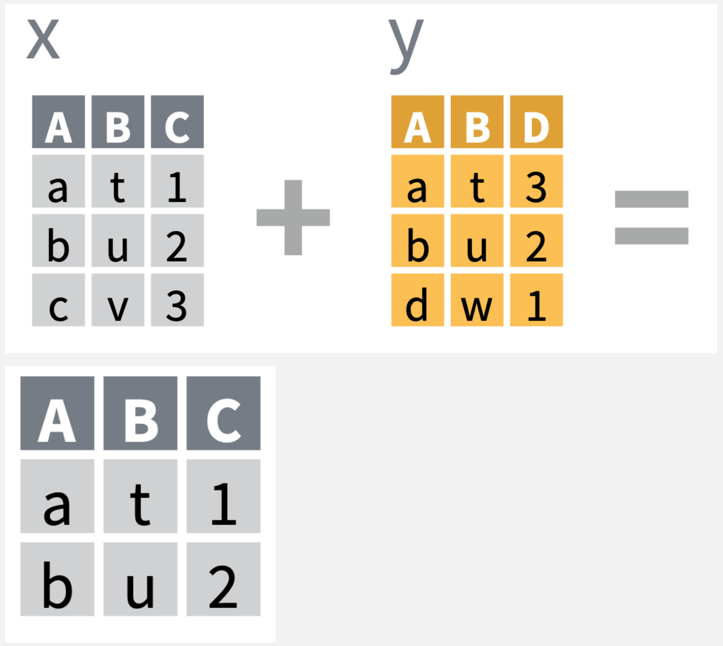


A	B	C
a	t	1
b	u	2
c	v	3

A	B	D
a	t	3
b	u	2
d	w	1

A FEW MORE JOIN FUNCTIONS

- There are 2 more join functions that can be useful
  - to check a join is going to go how you want.
  - Syntax is the same
- **`semi_join (x, y, by=)`** – rows of x that match in y (if you perform a left join will show what will be kept)



A	B	C
a	t	1
b	u	2
c	v	3

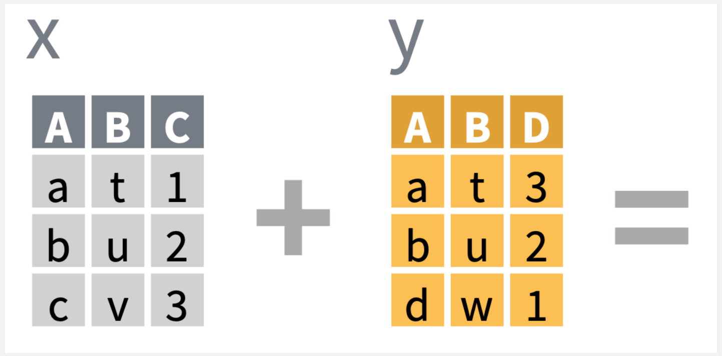
A	B	D
a	t	3
b	u	2
d	w	1

A	B	C
a	t	1
b	u	2

A FEW MORE JOIN FUNCTIONS

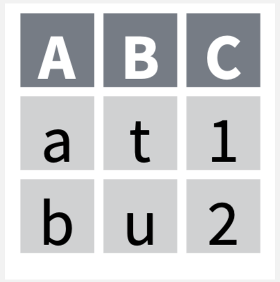
- There are 2 more join functions that can be useful
  - to check a join is going to go how you want.
  - Syntax is the same
- **`semi_join (x, y, by=)`** – rows of x that match in y (if you perform a left join will show what will be kept)

- **`anti_join (x, y, by=)`** – shows what in x is not in y (and will be excluded from join)

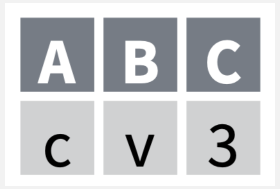


A	B	C
a	t	1
b	u	2
c	v	3

A	B	D
a	t	3
b	u	2
d	w	1

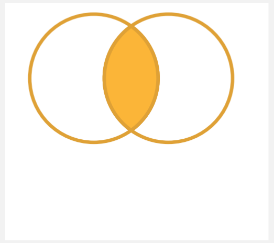


A	B	C
a	t	1
b	u	2



A	B	C
c	v	3

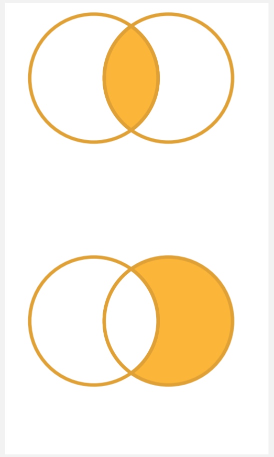
COMPARING SIMILAR DATASETS



- **`intersect(x, y)`**– rows that are in both x and y

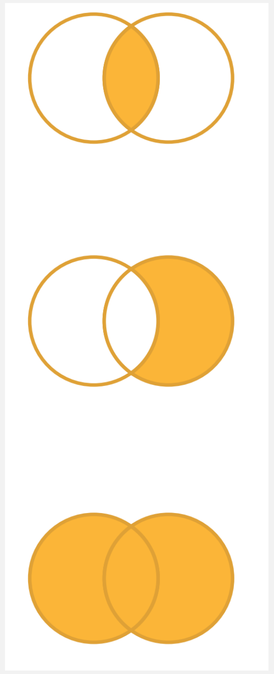
COMPARING SIMILAR DATASETS

- **`intersect(x, y)`**– rows that are in both x and y
- **`setdiff(x, y)`**– rows that are in x but not y

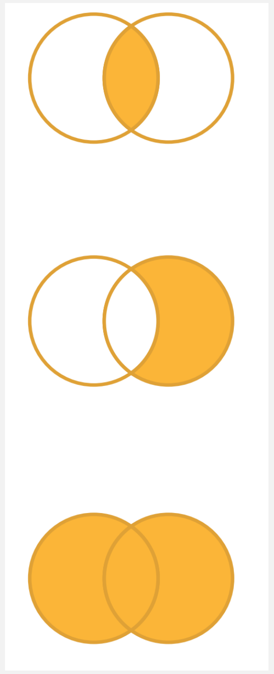


COMPARING SIMILAR DATASETS

- **`intersect(x, y)`**– rows that are in both x and y
- **`setdiff(x, y)`**– rows that are in x but not y
- **`union(x, y)`**– rows that’s are in x or y but without duplicates.



COMPARING SIMILAR DATASETS

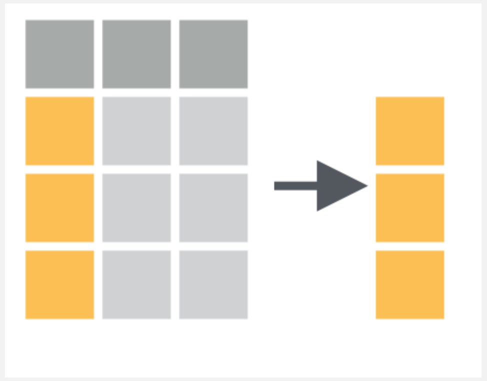


- **`intersect(x, y)`**– rows that are in both x and y
- **`setdiff(x, y)`**– rows that are in x but not y
- **`union(x, y)`**– rows that's are in x or y but without duplicates.
- **`setequal(x, y)`**– to test if two data set have the same rows in

  any order.

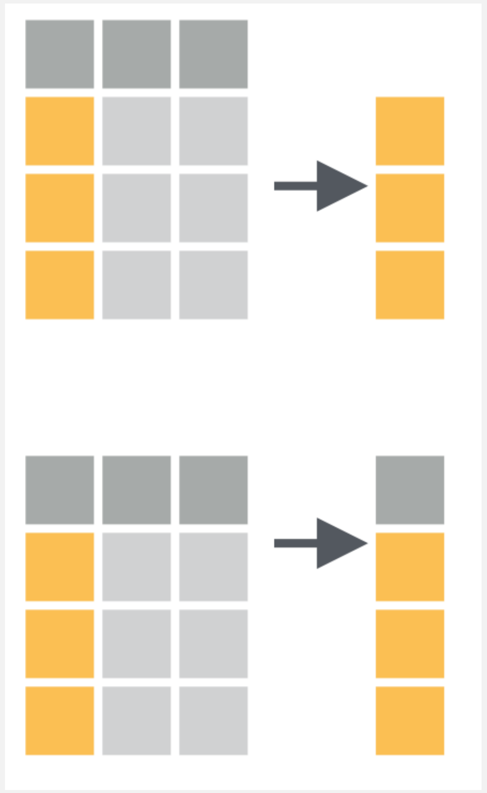
EXTRACTING VALUES

- **`pull(data, var`**`=col names or position)`– extracts the values of a column into a vector



EXTRACTING VALUES

- **pull(data, var**=col names or position)– extracts the values of a column into a vector
- **select(data, var**=col names or position)– extracts the column into a table



EXERCISE TIME!

SUMMARY

- How to check joins with `semi_join` and `anti_join`
- How to compare similar datasets with `intersect`, `setdiff` and `union`.
- How to extract values using `pull` or `select`.