Join together for fun

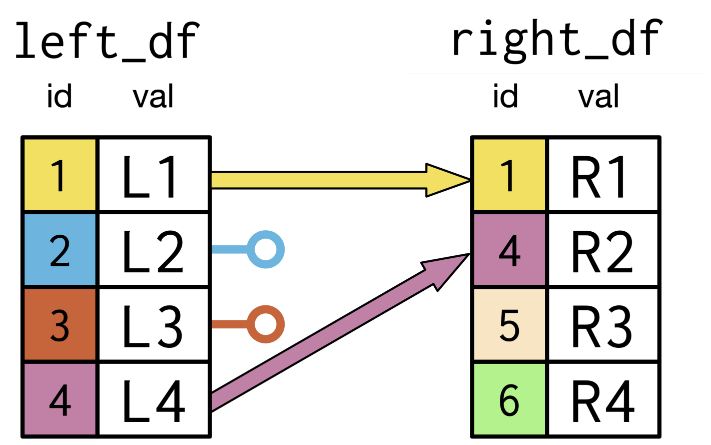
PROGRAMMING WITH DPLYR

Dr. Chester Ismay

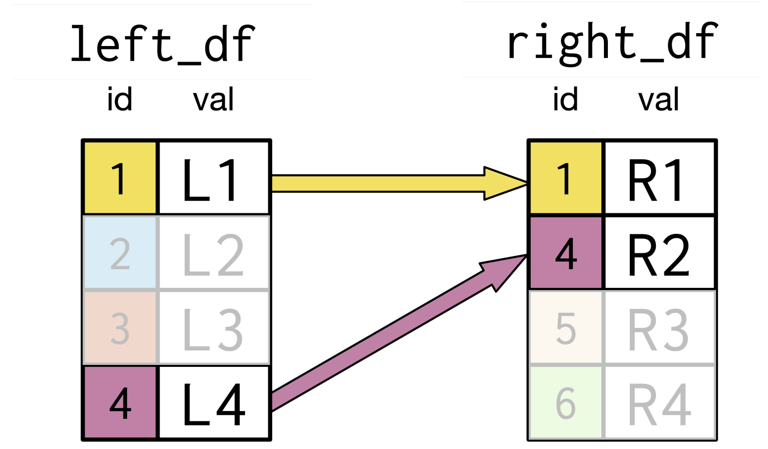
Educator, Data Scientist, and R/Python Consultant

dplyr join diagrams

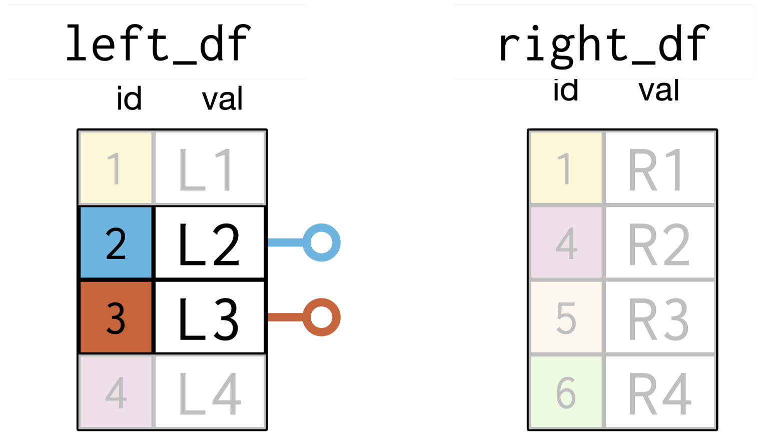
Left join



Inner join



Anti join



Some IMF data for Uruguay

```r
uruguay_imf <- imf_data %>%
  select(iso,
         country,
         year,
         consumer_price_index) %>%
  filter(country == "Uruguay", year > 2010)
uruguay_imf
```

```
# A tibble: 9 x 4
  iso   country  year consumer_price_index
  <chr> <chr>   <int>                <dbl>
1 URY   Uruguay  2011                 105.
2 URY   Uruguay  2012                 114.
3 URY   Uruguay  2013                 123.
4 URY   Uruguay  2014                 134.
5 URY   Uruguay  2015                 146.
6 URY   Uruguay  2016                 160.
7 URY   Uruguay  2017                 170.
8 URY   Uruguay  2018                 183.
9 URY   Uruguay  2019                 197.
```

Some World Bank data for Uruguay

```r
uruguay_wb <- world_bank_data %>%
  select(iso, country, year, perc_rural_pop) %>%
  filter(country == "Uruguay")
uruguay_wb
```

```
# A tibble: 4 x 4
  iso   country  year perc_rural_pop
  <chr> <chr>   <dbl>          <dbl>
1 URY   Uruguay  2013           5.16
2 URY   Uruguay  2014           5.06
3 URY   Uruguay  2015           4.96
4 URY   Uruguay  2016           4.86
```

```r
uruguay_imf %>%
    left_join(uruguay_wb)
```

```
Joining, by = c("iso", "country", "year")
# A tibble: 9 x 5
  iso   country  year consumer_price_index perc_rural_pop
  <chr> <chr>   <dbl>                <dbl>          <dbl>
1 URY   Uruguay  2011                 105.          NA
2 URY   Uruguay  2012                 114.          NA
3 URY   Uruguay  2013                 123.           5.16
4 URY   Uruguay  2014                 134.           5.06
5 URY   Uruguay  2015                 146.           4.96
6 URY   Uruguay  2016                 160.           4.86
7 URY   Uruguay  2017                 170.          NA
8 URY   Uruguay  2018                 183.          NA
9 URY   Uruguay  2019                 197.          NA
```

Inner join on Uruguayan tibbles

```
uruguay_imf %>%
    inner_join(uruguay_wb,
               by = c("iso", "country", "year"))
```

```
# A tibble: 4 x 5
  iso   country  year consumer_price_index perc_rural_pop
  <chr> <chr>   <dbl>                <dbl>          <dbl>
1 URY   Uruguay  2013                 123.           5.16
2 URY   Uruguay  2014                 134.           5.06
3 URY   Uruguay  2015                 146.           4.96
4 URY   Uruguay  2016                 160.           4.86
```

Anti join on Uruguayan tibbles

```
uruguay_imf %>%
    anti_join(uruguay_wb,
              by = c("iso", "country", "year"))
```

```
# A tibble: 5 x 4
  iso   country   year consumer_price_index
  <chr> <chr>    <int>                <dbl>
1 URY   Uruguay   2011                 105.
2 URY   Uruguay   2012                 114.
3 URY   Uruguay   2017                 170.
4 URY   Uruguay   2018                 183.
5 URY   Uruguay   2019                 197.
```

Let's practice!

PROGRAMMING WITH DPLYR

Lines that intersect are without parallel

PROGRAMMING WITH DPLYR

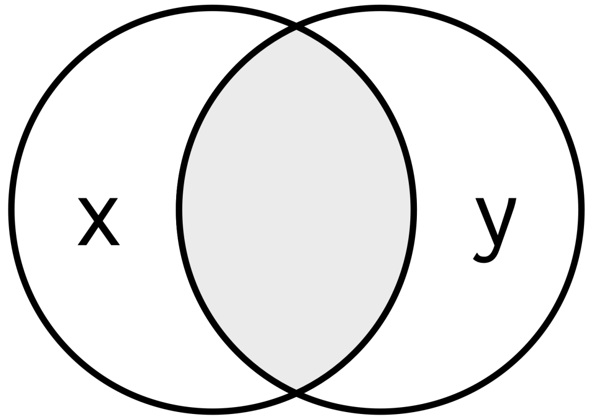
Dr. Chester Ismay

Educator, Data Scientist, and R/Python Consultant

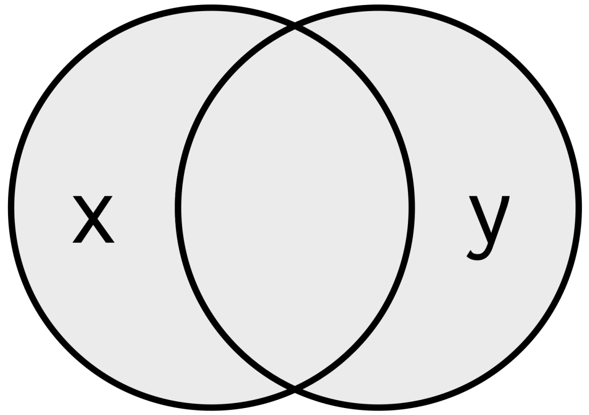
Set theory clauses

- Compare and combine data from two sources
- `dplyr` has several functions to perform set theory clauses on tibbles

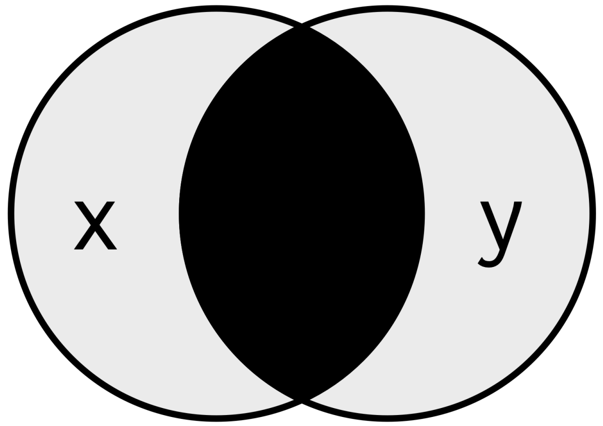
Venn diagrams for set theory



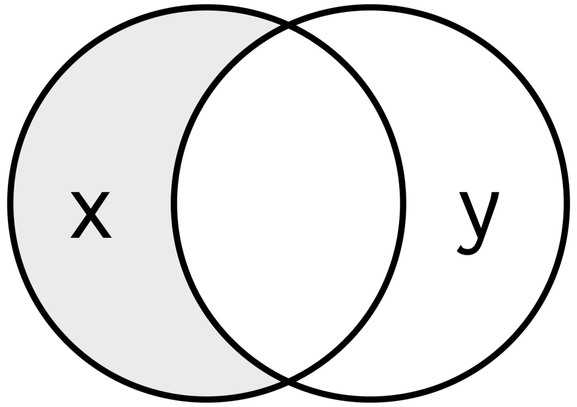
`intersect(x, y)`



`union(x, y)`

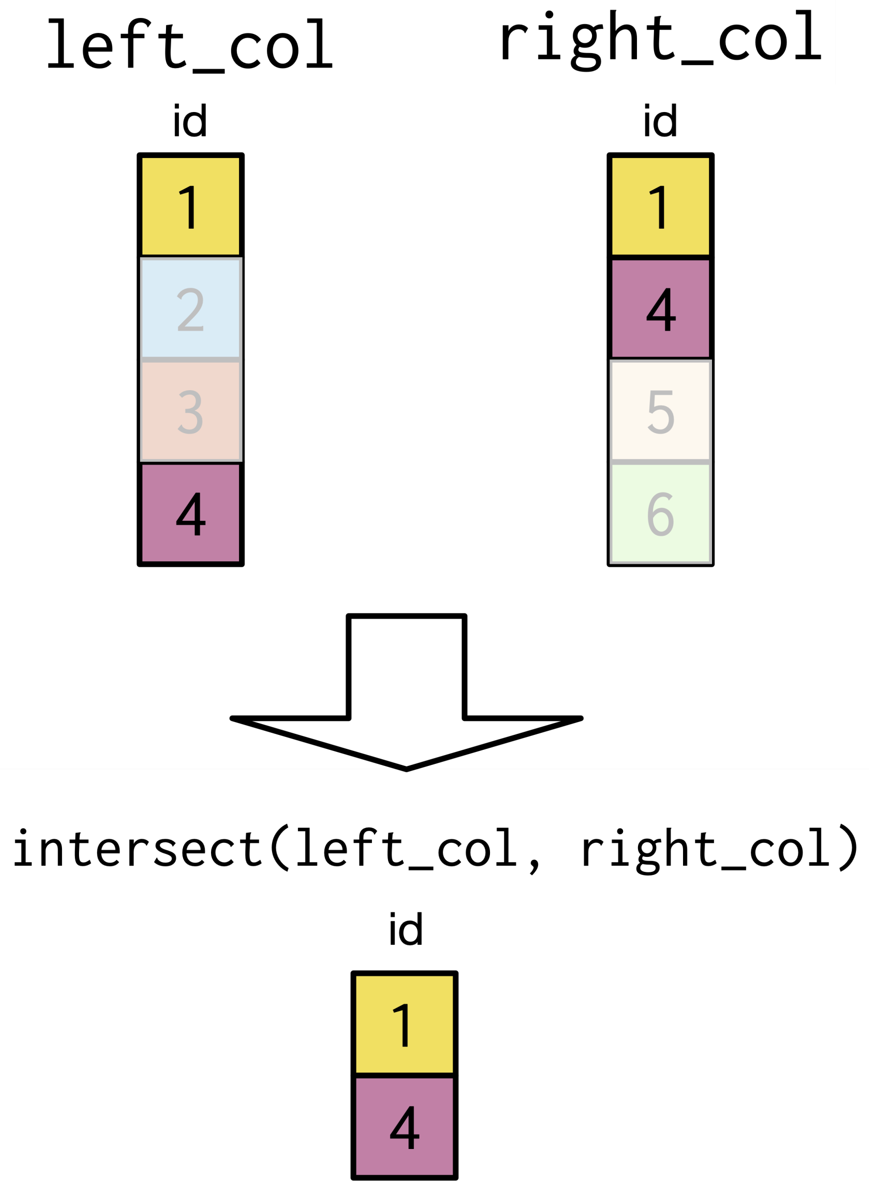


`union_all(x, y)`



`setdiff(x,y)`

intersect diagram



Uruguay tibbles

```
uruguay_imf
```

```
# A tibble: 9 x 4
  iso   country  year consumer_price_index
  <chr> <chr>   <int>                <dbl>
1 URY   Uruguay  2011                 105.
2 URY   Uruguay  2012                 114.
3 URY   Uruguay  2013                 123.
4 URY   Uruguay  2014                 134.
5 URY   Uruguay  2015                 146.
6 URY   Uruguay  2016                 160.
7 URY   Uruguay  2017                 170.
8 URY   Uruguay  2018                 183.
9 URY   Uruguay  2019                 197.
```

```
uruguay_wb
```

```
# A tibble: 4 x 4
  iso   country  year perc_rural_pop
  <chr> <chr>   <dbl>          <dbl>
1 URY   Uruguay  2013           5.16
2 URY   Uruguay  2014           5.06
3 URY   Uruguay  2015           4.96
4 URY   Uruguay  2016           4.86
```

Trying out intersect()

```
intersect(uruguay_imf, uruguay_wb)
```

```
Error: not compatible:
not compatible:
- Cols in y but not x: `perc_rural_pop`.
- Cols in x but not y: `consumer_price_index`.
```

```
intersect(uruguay_imf$year, uruguay_wb$year)
```

```
[1] 2013 2014 2015 2016
```

Difference between intersect() and a join

- `intersect()` looks for **rows** in common
- `inner_join()` looks for individual key entries matching

This is an important distinction.

Let's practice!

PROGRAMMING WITH DPLYR

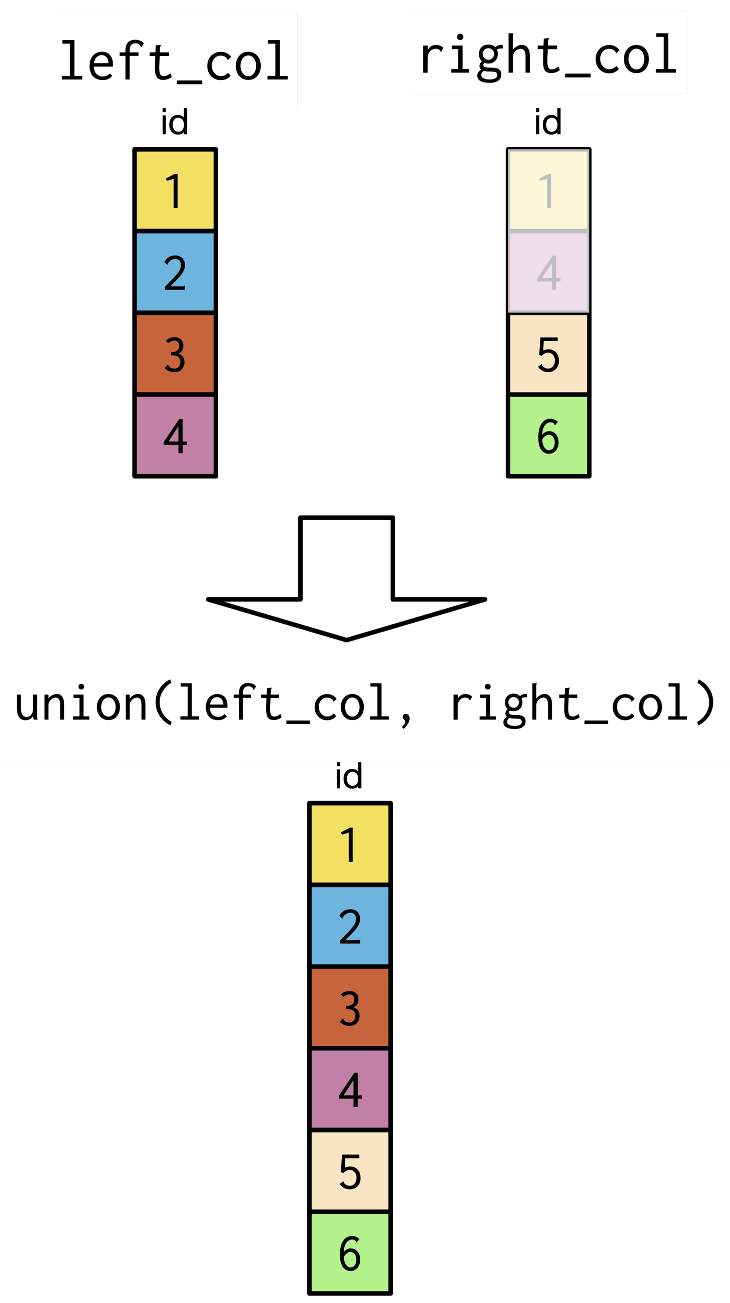
Deliver the state of the union

PROGRAMMING WITH DPLYR

Dr. Chester Ismay

Educator, Data Scientist, and R/Python Consultant

union diagram



Prepping for union with Uruguay

```
uruguay_imf_filtered <- imf_data %>%
  select(iso, country, year) %>%
  filter(country == "Uruguay", between(year, 2010, 2014))
```

```
uruguay_wb_filtered <- world_bank_data %>%
  select(iso, country, year) %>%
  filter(country == "Uruguay")
```

The new tibbles

```
uruguay_imf_filtered
```

```
# A tibble: 5 x 3
  iso   country  year
  <chr> <chr>   <int>
1 URY   Uruguay  2010
2 URY   Uruguay  2011
3 URY   Uruguay  2012
4 URY   Uruguay  2013
5 URY   Uruguay  2014
```

```
uruguay_wb_filtered
```

```
# A tibble: 4 x 3
  iso   country  year
  <chr> <chr>   <dbl>
1 URY   Uruguay  2013
2 URY   Uruguay  2014
3 URY   Uruguay  2015
4 URY   Uruguay  2016
```

union()

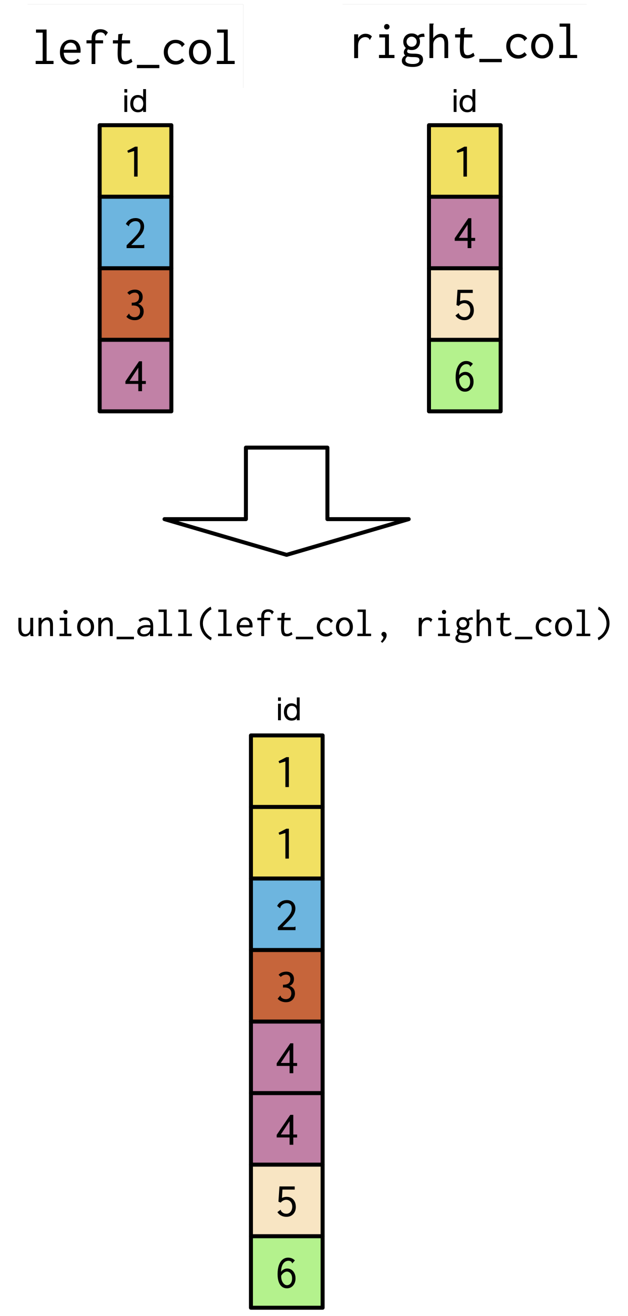
```
union(uruguay_imf_filtered,
      uruguay_wb_filtered)
```

```
# A tibble: 7 x 3
  iso   country   year
  <chr> <chr>    <dbl>
1 URY   Uruguay   2010
2 URY   Uruguay   2011
3 URY   Uruguay   2012
4 URY   Uruguay   2013
5 URY   Uruguay   2014
6 URY   Uruguay   2015
7 URY   Uruguay   2016
```

```
union(uruguay_wb_filtered,
      uruguay_imf_filtered)
```

```
# A tibble: 7 x 3
  iso   country   year
  <chr> <chr>    <dbl>
1 URY   Uruguay   2013
2 URY   Uruguay   2014
3 URY   Uruguay   2015
4 URY   Uruguay   2016
5 URY   Uruguay   2010
6 URY   Uruguay   2011
7 URY   Uruguay   2012
```

union_all diagram



```r
union_all(uruguay_imf_filtered,
          uruguay_wb_filtered)
```

```
# A tibble: 9 x 3
  iso   country  year
  <chr> <chr>   <dbl>
1 URY   Uruguay  2010
2 URY   Uruguay  2011
3 URY   Uruguay  2012
4 URY   Uruguay  2013
5 URY   Uruguay  2014
6 URY   Uruguay  2013
7 URY   Uruguay  2014
8 URY   Uruguay  2015
9 URY   Uruguay  2016
```

```r
union_all(uruguay_wb_filtered,
          uruguay_imf_filtered)
```

```
# A tibble: 9 x 3
  iso   country  year
  <chr> <chr>   <dbl>
1 URY   Uruguay  2013
2 URY   Uruguay  2014
3 URY   Uruguay  2015
4 URY   Uruguay  2016
5 URY   Uruguay  2010
6 URY   Uruguay  2011
7 URY   Uruguay  2012
8 URY   Uruguay  2013
9 URY   Uruguay  2014
```

Let's practice!

PROGRAMMING WITH DPLYR

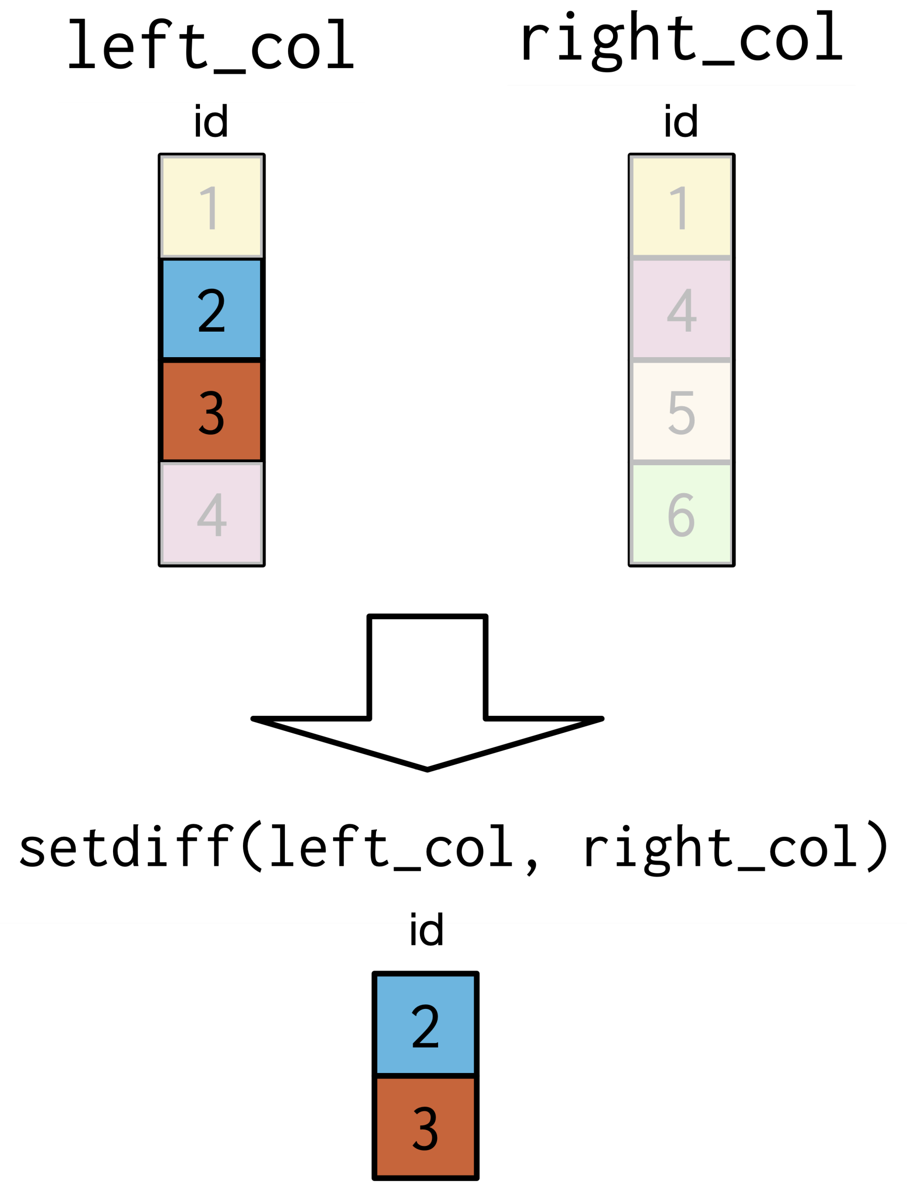
A little too excepting

PROGRAMMING WITH DPLYR

Dr. Chester Ismay

Educator, Data Scientist, and R/Python Consultant

setdiff diagram



The tibbles again

```
uruguay_imf_filtered
```

```
# A tibble: 5 x 3
  iso   country  year
  <chr> <chr>   <int>
1 URY   Uruguay  2010
2 URY   Uruguay  2011
3 URY   Uruguay  2012
4 URY   Uruguay  2013
5 URY   Uruguay  2014
```

```
uruguay_wb_filtered
```

```
# A tibble: 4 x 3
  iso   country  year
  <chr> <chr>   <dbl>
1 URY   Uruguay  2013
2 URY   Uruguay  2014
3 URY   Uruguay  2015
4 URY   Uruguay  2016
```

setdiff()

```
setdiff(uruguay_imf_filtered, uruguay_wb_filtered)
```

```
# A tibble: 3 x 3
  iso   country  year
  <chr> <chr>   <dbl>
1 URY   Uruguay  2010
2 URY   Uruguay  2011
3 URY   Uruguay  2012
```

Previous unions()

```r
union_one_way <- union(uruguay_imf_filtered,
                       uruguay_wb_filtered)
union_one_way
```

```
# A tibble: 7 x 3
  iso   country  year
  <chr> <chr>   <dbl>
1 URY   Uruguay  2010
2 URY   Uruguay  2011
3 URY   Uruguay  2012
4 URY   Uruguay  2013
5 URY   Uruguay  2014
6 URY   Uruguay  2015
7 URY   Uruguay  2016
```

```r
union_other <- union(uruguay_wb_filtered,
                     uruguay_imf_filtered)
union_other
```

```
# A tibble: 7 x 3
  iso   country  year
  <chr> <chr>   <dbl>
1 URY   Uruguay  2013
2 URY   Uruguay  2014
3 URY   Uruguay  2015
4 URY   Uruguay  2016
5 URY   Uruguay  2010
6 URY   Uruguay  2011
7 URY   Uruguay  2012
```

setequal() example

```
setequal(union_one_way, union_other)
```

```
[1] TRUE
```

Let's practice!

PROGRAMMING WITH DPLYR