STAT 408: Week 7

Tidy Data and Relational Data

3/1/2022

Data Wrangling

Data Wrangling

As a statistician or more generally a data scientist the ability to manipulate, process, clean, and merge datasets is an essential skill.

- These skills are generally referred to as data wrangling or munging.
- In a data analysis or visualization setting, they will undoubtedly require a majority of your time.
- Wrangling data can be a painful process.
- This lecture will provide some tools and examples of organizing data.

Data Wrangling Concepts

- Wide and thin datasets
- Merging and joining relational data
- Dealing with strings
- Dealing with date/time objects

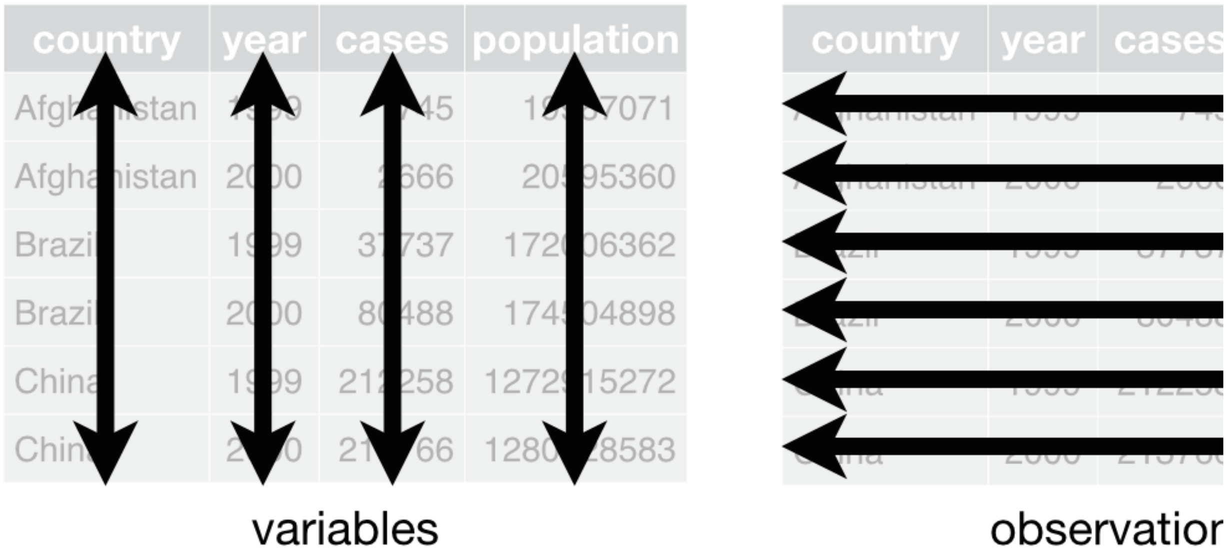
Tidy Data

Rules for Tidy Data

The concept of **tidy data** can be attributed to Hadley Wickham and has three principles for organizing data. [Tidy Data Reference]

- Each variable forms a column.
- Each observation forms a row.
- Each type of observational unit forms a table (with a single value in each cell).

Rules for Tidy Data



Visual Representation of Tidy Data. Source: R4DS

Why use Tidy Data

Tidy datasets are all alike, but every messy dataset is messy in its own way. - Hadley Wickham

- Storing data in a consistent way gives familiarity with methods for manipulating data.
- Tidy data structure takes advantage of vectorised operations in R.
- Many useful packages: such as `dplyr` and `ggplot2` require tidy data.

What are messy data?

Airplanes on Hand in the AAF, By Major Type:
Jul 1939 to Aug 1945

End of Month	Total	Very Heavy Bombers	Heavy Bombers	Medium Bombers	Light Bombers	Fighters	Recon-naissance	Transports	Trainers	Communi-cations
1939										
Jul	2,402	-	16	400	276	494	356	118	735	7
Aug	2,440	-	18	414	276	492	359	129	745	7
[Germany invades Poland, 1 Sep 1939]										
Sep	2,473	-	22	428	278	489	359	136	754	7
Oct	2,507	-	27	446	277	490	365	137	758	7
Nov	2,536	-	32	458	275	498	375	136	755	7
Dec	2,546	-	39	464	274	492	378	131	761	7
1940										
Jan	2,588	-	45	466	271	464	409	128	798	7
Feb	2,658	-	49	470	271	458	415	128	860	7
Mar	2,709	-	54	468	267	453	415	125	920	7
Apr	2,806	-	54	468	263	451	416	125	1,022	7
May	2,906	-	54	470	259	459	410	124	1,123	7
Jun	2,966	-	54	478	166	477	414	127	1,243	7
[France surrenders to Germany, 25 Jun 1940] [Battle of Britain begins, 10 July 1940]										
Jul	3,102	-	56	483	161	500	410	128	1,357	7
Aug	3,295	-	65	485	158	539	407	128	1,506	7

- Year –> should be its own column
- Historical markers –> could make into variables or just use on annotations

Source: Army Air Forces Statistical Digest, WW II

What are messy data?

Subject	United States			
	Estimate	Margin of Error	Percent	Percent Margin of Error
EMPLOYMENT STATUS				
Population 16 years and over	255,797,692	+/-17,051	255,797,692	(X)
In labor force	162,184,325	+/-135,158	63.4%	+/-0.1
Civilian labor force	161,159,470	+/-127,501	63.0%	+/-0.1
Employed	150,599,165	+/-138,066	58.9%	+/-0.1
Unemployed	10,560,305	+/-27,385	4.1%	+/-0.1
Armed Forces	1,024,855	+/-10,363	0.4%	+/-0.1
Not in labor force	93,613,367	+/-126,007	36.6%	+/-0.1
Civilian labor force	161,159,470	+/-127,501	161,159,470	(X)
Unemployment Rate	(X)	(X)	6.6%	+/-0.1
Females 16 years and over	131,092,196	+/-11,187	131,092,196	(X)
In labor force	76,493,327	+/-75,824	58.4%	+/-0.1
Civilian labor force	76,350,498	+/-75,238	58.2%	+/-0.1
Employed	71,451,559	+/-79,007	54.5%	+/-0.1
Own children of the householder under 6 years	22,939,897	+/-14,240	22,939,897	(X)
All parents in family in labor force	14,957,537	+/-36,506	65.2%	+/-0.1
Own children of the householder 6 to 17 years	47,007,147	+/-19,644	47,007,147	(X)
All parents in family in labor force	33,238,793	+/-49,036	70.7%	+/-0.1

- Data released in aggregate
- Difficult to get data at the individual level (data privacy issues)

Source: US Census Fact Finder, General Economic Characteristics, ACS 2017

Displaying vs summarizing data

- Summary data might look “tidy”, but its rows are not the observational units
- Raw data can produce summary data, but you can’t go back to raw data from summary data

Displaying vs summarizing data

Raw data or summary data?

```
## # A tibble: 173 × 3
##    Major                                      ShareWomen Unemployment_rate
##    <chr>                                           <dbl>             <dbl>
##  1 PETROLEUM ENGINEERING                           0.121            0.0184
##  2 MINING AND MINERAL ENGINEERING                  0.102            0.117
##  3 METALLURGICAL ENGINEERING                       0.153            0.0241
##  4 NAVAL ARCHITECTURE AND MARINE ENGINEERING       0.107            0.0501
##  5 CHEMICAL ENGINEERING                            0.342            0.0611
##  6 NUCLEAR ENGINEERING                             0.145            0.177
##  7 ACTUARIAL SCIENCE                               0.441            0.0957
##  8 ASTRONOMY AND ASTROPHYSICS                      0.536            0.0212
##  9 MECHANICAL ENGINEERING                          0.120            0.0573
## 10 ELECTRICAL ENGINEERING                          0.196            0.0592
## # … with 163 more rows
```

Displaying vs summarizing data

Raw data or summary data?

```
## # A tibble: 16 × 2
##    Major_category                        ave_med_salary
##    <chr>                                          <dbl>
##  1 Agriculture & Natural Resources               36900 
##  2 Arts                                          33062.
##  3 Biology & Life Science                        36421.
##  4 Business                                      43538.
##  5 Communications & Journalism                   34500 
##  6 Computers & Mathematics                       42745.
##  7 Education                                     32350 
##  8 Engineering                                   57383.
##  9 Health                                        36825 
## 10 Humanities & Liberal Arts                     31913.
## 11 Industrial Arts & Consumer Services           36343.
## 12 Interdisciplinary                             35000 
## 13 Law & Public Policy                           42200 
## 14 Physical Sciences                             41890 
## 15 Psychology & Social Work                      30100 
## 16 Social Science                                37344.
```


Merge / Join

Merging in Base R

Consider the two data frames, how can we merge them and what should be the dimensions of the merged data frame.

```
##   school state
## 1    MSU    MT
## 2     VT    VA
## 3  Mines    CO
```

```
##   school enrollment
## 1  Mines       5794
## 2    MSU      15688
## 3     VT      30598
```


pre-sort

One possibility is to use the arrange the data frames first.

```
df1 <- df1[order(df1$school),]
df2 <- df2[order(df2$school),]
```

pre-sort

One possibility is to use the arrange the data frames first.

```
df1
```

```
##   school state
## 3  Mines    CO
## 1    MSU    MT
## 2     VT    VA
```

```
df2
```

```
##   school enrollment
## 1  Mines       5794
## 2    MSU      15688
## 3     VT      30598
```


rbind() and cbind()

Now, given that the data frames are both sorted the same way, we can bind the columns together.

```
comb_df <- cbind(df1,df2)
comb_df
```

```
##   school state school enrollment
## 3  Mines    CO  Mines       5794
## 1    MSU    MT    MSU      15688
## 2     VT    VA     VT      30598
```

```
comb_df <- comb_df[,-3]
```

rbind() and cbind()

Now assume we want to add another school to the data frame.

```
new.school <- c('Luther', 'IA',2337)
rbind(comb_df, new.school)
```

```
##   school state enrollment
## 3  Mines    CO       5794
## 1    MSU    MT      15688
## 2     VT    VA      30598
## 4 Luther    IA       2337
```

Note: If your strings are saved as factors, this chunk of code will give you an error.

bind_rows() / bind_cols()

`dplyr` also contains functions for - binding rows: `bind_rows()` - binding columns: `bind_cols()`

Joins in `dplyr`

Data: Women in science

Information on 10 women in science who changed the world.

name
Ada Lovelace
Marie Curie
Janaki Ammal
Chien-Shiung Wu
Katherine Johnson
Rosalind Franklin
Vera Rubin
Gladys West
Flossie Wong-Staal
Jennifer Doudna

Source: Discover Magazine

Inputs

```
professions
```

```
## # A tibble: 10 × 2
##    name               profession
##    <chr>              <chr>
##  1 Ada Lovelace       Mathematician
##  2 Marie Curie        Physicist and Chemist
##  3 Janaki Ammal       Botanist
##  4 Chien-Shiung Wu    Physicist
##  5 Katherine Johnson  Mathematician
##  6 Rosalind Franklin  Chemist
##  7 Vera Rubin         Astronomer
##  8 Gladys West        Mathematician
##  9 Flossie Wong-Staal Virologist and Molecular Biologist
## 10 Jennifer Doudna    Biochemist
```


Inputs

```
dates
```

```
## # A tibble: 8 × 3
##   name               birth_year death_year
##   <chr>                   <dbl>      <dbl>
## 1 Janaki Ammal             1897       1984
## 2 Chien-Shiung Wu          1912       1997
## 3 Katherine Johnson        1918       2020
## 4 Rosalind Franklin        1920       1958
## 5 Vera Rubin               1928       2016
## 6 Gladys West              1930         NA
## 7 Flossie Wong-Staal       1947         NA
## 8 Jennifer Doudna          1964         NA
```

Inputs

```
works
```

```
## # A tibble: 9 × 2
##   name               known_for
##   <chr>              <chr>
## 1 Ada Lovelace       first computer algorithm
## 2 Marie Curie        theory of radioactivity,  discovery of elements polonium a…
## 3 Janaki Ammal       hybrid species, biodiversity protection
## 4 Chien-Shiung Wu    confim and refine theory of radioactive beta decy, Wu expe…
## 5 Katherine Johnson  calculations of orbital mechanics critical to sending the …
## 6 Vera Rubin         existence of dark matter
## 7 Gladys West        mathematical modeling of the shape of the Earth which serv…
## 8 Flossie Wong-Staal first scientist to clone HIV and create a map of its genes…
## 9 Jennifer Doudna    one of the primary developers of CRISPR, a ground-breaking…
```


Desired output

```
## # A tibble: 10 × 5
##    name               profession                 birth_year death_year known_for
##    <chr>              <chr>                           <dbl>      <dbl> <chr>
##  1 Ada Lovelace       Mathematician                      NA         NA first co…
##  2 Marie Curie        Physicist and Chemist              NA         NA theory o…
##  3 Janaki Ammal       Botanist                         1897       1984 hybrid s…
##  4 Chien-Shiung Wu    Physicist                        1912       1997 confim a…
##  5 Katherine Johnson  Mathematician                    1918       2020 calculat…
##  6 Rosalind Franklin  Chemist                          1920       1958 <NA>
##  7 Vera Rubin         Astronomer                       1928       2016 existenc…
##  8 Gladys West        Mathematician                    1930         NA mathemat…
##  9 Flossie Wong-Staal Virologist and Molecular …       1947         NA first sc…
## 10 Jennifer Doudna    Biochemist                       1964         NA one of t…
```

Inputs, reminder

```
names(professions)
## [1] "name"       "profession"
names(dates)
## [1] "name"       "birth_year" "death_year"
names(works)
## [1] "name"      "known_for"

nrow(professions)
## [1] 10
nrow(dates)
## [1] 8
nrow(works)
## [1] 9
```


Joining data frames

```
something_join(x, y)
```

- `left_join()`: all rows from x
- `right_join()`: all rows from y
- `full_join()`: all rows from both x and y
- `semi_join()`: all rows from x where there are matching values in y, keeping just columns from x
- `inner_join()`: all rows from x where there are matching values in y, return all combination of multiple matches in the case of multiple matches
- `anti_join()`: return all rows from x where there are not matching values in y, never duplicate rows of x
- …

Setup

For the next few slides…

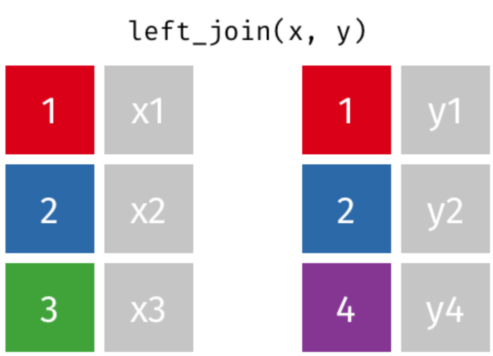
```
x
```

```
## # A tibble: 3 × 2
##      id value_x
##   <dbl> <chr>
## 1     1 x1
## 2     2 x2
## 3     3 x3
```

```
y
```

```
## # A tibble: 3 × 2
##      id value_y
##   <dbl> <chr>
## 1     1 y1
## 2     2 y2
## 3     4 y4
```

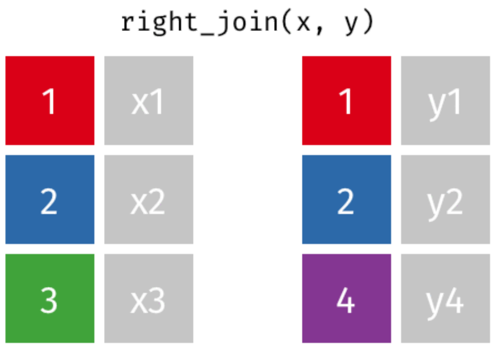

`left_join()`




```
professions %>%
  left_join(dates)
```

```
## # A tibble: 10 × 4
##    name               profession
##    <chr>              <chr>
##  1 Ada Lovelace       Mathematician
##  2 Marie Curie        Physicist and Chemist
##  3 Janaki Ammal       Botanist
##  4 Chien-Shiung Wu    Physicist
##  5 Katherine Johnson  Mathematician
##  6 Rosalind Franklin  Chemist
##  7 Vera Rubin         Astronomer
##  8 Gladys West        Mathematician
##  9 Flossie Wong-Staal Virologist and Molecular
## 10 Jennifer Doudna    Biochemist
```

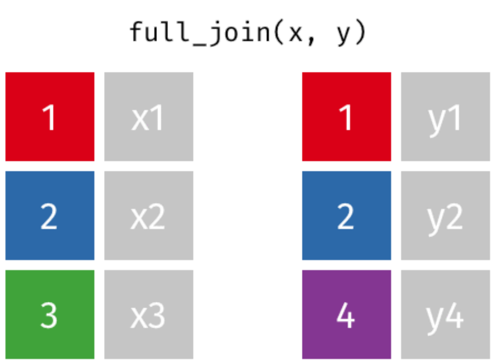
right_join()




```
professions %>%
  right_join(dates)
```

```
## # A tibble: 8 × 4
##   name               profession
##   <chr>              <chr>
## 1 Janaki Ammal       Botanist
## 2 Chien-Shiung Wu    Physicist
## 3 Katherine Johnson  Mathematician
## 4 Rosalind Franklin  Chemist
## 5 Vera Rubin         Astronomer
## 6 Gladys West        Mathematician
## 7 Flossie Wong-Staal Virologist and Molecular
## 8 Jennifer Doudna    Biochemist
```

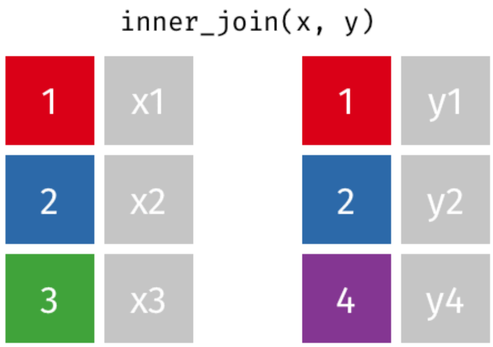

full_join()




```
dates %>%
  full_join(works)
```

```
## # A tibble: 10 × 4
##    name               birth_year death_year kn
##    <chr>                   <dbl>      <dbl> <c
##  1 Janaki Ammal             1897       1984 hy
##  2 Chien-Shiung Wu          1912       1997 cc
##  3 Katherine Johnson        1918       2020 ca
##  4 Rosalind Franklin        1920       1958 <N
##  5 Vera Rubin               1928       2016 ex
##  6 Gladys West              1930         NA ma
##  7 Flossie Wong-Staal       1947         NA fi
##  8 Jennifer Doudna          1964         NA on
##  9 Ada Lovelace               NA         NA fi
## 10 Marie Curie                NA         NA th
```

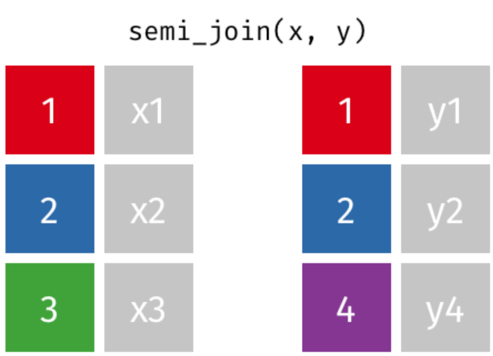
inner_join()




```
dates %>%
  inner_join(works)
```

```
## # A tibble: 7 × 4
##   name                birth_year death_year knc
##   <chr>                    <dbl>      <dbl> <ch
## 1 Janaki Ammal              1897       1984 hyb
## 2 Chien-Shiung Wu           1912       1997 con
## 3 Katherine Johnson         1918       2020 cal
## 4 Vera Rubin                1928       2016 exi
## 5 Gladys West               1930         NA mat
## 6 Flossie Wong-Staal        1947         NA fir
## 7 Jennifer Doudna           1964         NA one
```

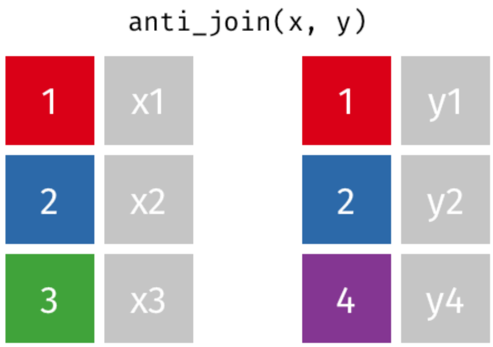

semi_join()




```
dates %>%
  semi_join(works)
```

```
## # A tibble: 7 × 3
##   name                birth_year death_year
##   <chr>                    <dbl>      <dbl>
## 1 Janaki Ammal              1897       1984
## 2 Chien-Shiung Wu           1912       1997
## 3 Katherine Johnson         1918       2020
## 4 Vera Rubin                1928       2016
## 5 Gladys West               1930         NA
## 6 Flossie Wong-Staal        1947         NA
## 7 Jennifer Doudna           1964         NA
```

`anti_join()`




```
dates %>%
  anti_join(works)
```

```
## # A tibble: 1 × 3
##   name              birth_year death_year
##   <chr>                  <dbl>      <dbl>
## 1 Rosalind Franklin       1920       1958
```


Putting it altogether

```
professions %>%
  left_join(dates) %>%
  left_join(works)
```

```
## # A tibble: 10 × 5
##    name               profession                 birth_year death_year known_for
##    <chr>              <chr>                           <dbl>      <dbl> <chr>
##  1 Ada Lovelace       Mathematician                      NA         NA first co…
##  2 Marie Curie        Physicist and Chemist              NA         NA theory o…
##  3 Janaki Ammal       Botanist                         1897       1984 hybrid s…
##  4 Chien-Shiung Wu    Physicist                        1912       1997 confim a…
##  5 Katherine Johnson  Mathematician                    1918       2020 calculat…
##  6 Rosalind Franklin  Chemist                          1920       1958 <NA>
##  7 Vera Rubin         Astronomer                       1928       2016 existenc…
##  8 Gladys West        Mathematician                    1930         NA mathemat…
##  9 Flossie Wong-Staal Virologist and Molecular …       1947         NA first sc…
## 10 Jennifer Doudna    Biochemist                       1964         NA one of t…
```

Exercise: Student records

Student records

- Have:
  - Enrollment: official university enrollment records
  - Survey: Student provided info missing students who never filled it out and including students who filled it out but dropped the class
- Want:
  - Survey info for all enrolled in class
  - Students who are enrolled in class but missing survey
  - Students who took the survey but are no longer enrolled

```
enrollment
```

```
## # A tibble: 3 × 2
##      id name
##   <dbl> <chr>
## 1     1 Dave Friday
## 2     2 Hermine
## 3     3 Sura Selvarajah
```

```
survey
```

```
## # A tibble: 4 × 3
##      id name    username
##   <dbl> <chr>   <chr>
## 1     2 Hermine bakealongwithhermine
## 2     3 Sura    surasbakes
## 3     4 Peter   peter_bakes
## 4     5 Mark    thebakingbuddha
```


Student records: Solution

In class

```
enrollment %>%
  left_join(survey, by = "id")
```

```
## # A tibble: 3 × 4
##      id name.x          name.y  username
##   <dbl> <chr>           <chr>   <chr>
## 1     1 Dave Friday     <NA>    <NA>
## 2     2 Hermine         Hermine bakealongwithhermine
## 3     3 Sura Selvarajah Sura    surasbakes
```

Survey missing

```
enrollment %>%
  anti_join(survey, by = "id")
```

```
## # A tibble: 1 × 2
##      id name
##   <dbl> <chr>
## 1     1 Dave Friday
```


Dropped

```
survey %>%
  anti_join(enrollment, by = "id")
```

```
## # A tibble: 2 × 3
##      id name  username
##   <dbl> <chr> <chr>
## 1     4 Peter peter_bakes
## 2     5 Mark  thebakingbuddha
```

Exercise: Grocery sales

Grocery sales

- Have:
  - Purchases: One row per customer per item, listing purchases they made
  - Prices: One row per item in the store, listing their prices
- Want:
  - Total revenue (over all customers)
  - Revenue per customer

```r
purchases
```

```
## # A tibble: 5 × 2
##   customer_id item
##         <dbl> <chr>
## 1           1 bread
## 2           1 milk
## 3           1 banana
## 4           2 milk
## 5           2 toilet paper
```

```r
prices
```

```
## # A tibble: 5 × 2
##   item         price
##   <chr>        <dbl>
## 1 avocado       0.5
## 2 banana        0.15
## 3 bread         1
## 4 milk          0.8
## 5 toilet paper  3
```


Grocery sales: Solution

Total revenue

```r
purchases %>%
  left_join(prices)
```

```
## # A tibble: 5 × 3
##   customer_id item         price
##         <dbl> <chr>        <dbl>
## 1           1 bread         1
## 2           1 milk          0.8
## 3           1 banana        0.15
## 4           2 milk          0.8
## 5           2 toilet paper  3
```

```r
purchases %>%
  left_join(prices) %>%
  summarise(total_revenue = sum(price))
```

```
## # A tibble: 1 × 1
##   total_revenue
##           <dbl>
## 1          5.75
```

Revenue by customer

```
purchases %>%
  left_join(prices)
```

```
## # A tibble: 5 × 3
##   customer_id item         price
##         <dbl> <chr>        <dbl>
## 1           1 bread         1
## 2           1 milk          0.8
## 3           1 banana        0.15
## 4           2 milk          0.8
## 5           2 toilet paper  3
```

```
purchases %>%
  left_join(prices) %>%
  group_by(customer_id) %>%
  summarise(total_revenue = sum(price))
```

```
## # A tibble: 2 × 2
##   customer_id total_revenue
##         <dbl>         <dbl>
## 1           1          1.95
## 2           2          3.8
```


Exercise: Ski hills

Ski hills

Combine the following information into a single table sorted alphabetically by the name of the ski hill.

```
ski_acres
```

```
##       ski.hill skiable.acres
## 1      Big Sky          5800
## 2 Bridger Bowl          2000
## 3      Jackson         2500+
## 4    Steamboat          2965
```

```
df_cost
```

```
##     ski.resort ticket.cost
## 1 Bridger Bowl          60
## 2      Big Sky     depends
## 3    Steamboat         145
## 4      Jackson         130
```

```
disco
```

```
## [1] "Discovery" "2200"      "20"
```


Solution option 1

```
df_comb <- ski_acres %>%
  full_join(df_cost, by = c("ski.hill" = "ski.resort")) %>%
  mutate(ski.hill = as.character(ski.hill),
         skiable.acres = as.character(skiable.acres),
         ticket.cost = as.character(ticket.cost)) %>%
  rbind(disco) %>% #<<<
  arrange(ski.hill)
str(df_comb)
```

```
## 'data.frame':    5 obs. of  3 variables:
##  $ ski.hill     : chr  "Big Sky" "Bridger Bowl" "Discovery" "Jackson" ...
##  $ skiable.acres: chr  "5800" "2000" "2200" "2500+" ...
##  $ ticket.cost  : chr  "depends" "60" "20" "130" ...
```

Solution option 2

```
disco_df <- data.frame(matrix(disco, nrow = 1))
names(disco_df) <- c("ski.hill", "skiable.acres", "ticket.cost")

df_comb <- ski_acres %>%
  full_join(df_cost, by = c("ski.hill"= "ski.resort")) %>%
  mutate(ski.hill = as.character(ski.hill),
         skiable.acres = as.character(skiable.acres),
         ticket.cost = as.character(ticket.cost)) %>%
  full_join(disco_df) %>% #<<<
  arrange(ski.hill)
str(df_comb)
```

```
## 'data.frame':    5 obs. of  3 variables:
##  $ ski.hill     : chr  "Big Sky" "Bridger Bowl" "Discovery" "Jackson" ...
##  $ skiable.acres: chr  "5800" "2000" "2200" "2500+" ...
##  $ ticket.cost  : chr  "depends" "60" "20" "130" ...
```


wide / long data

wide(r) / long(er) data

We have data organised in an unideal way for our analysis.

We want to reorganise the data to carry on with our analysis.

Data: Grocery sales

We have…

```
## # A tibble: 2 × 4
##   customer_id item_1 item_2       item_3
##         <dbl> <chr>  <chr>        <chr>
## 1           1 bread  milk         banana
## 2           2 milk   toilet paper <NA>
```

We want…

```
## # A tibble: 6 × 3
##   customer_id item_no item
##         <dbl> <chr>   <chr>
## 1           1 item_1  bread
## 2           1 item_2  milk
## 3           1 item_3  banana
## 4           2 item_1  milk
## 5           2 item_2  toilet paper
## 6           2 item_3  <NA>
```

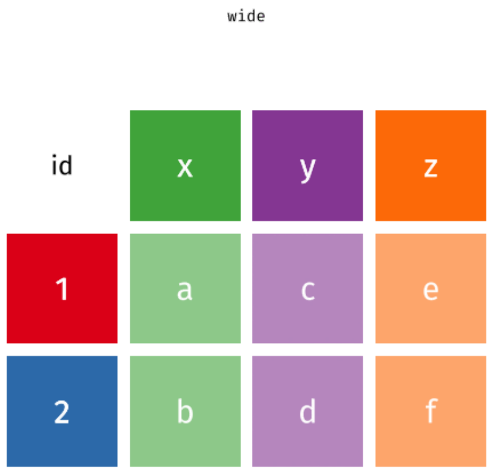
A grammar of data tidying

The goal of tidyr is to help you tidy your data via

- pivoting for going between wide and long data
- splitting and combining character columns
- nesting and unnesting columns
- clarifying how `NA`s should be treated

Not this…

but this!



Wider vs. longer

Wider = more columns

```
## # A tibble: 2 × 4
##   customer_id item_1 item_2       item_3
##         <dbl> <chr>  <chr>        <chr>
## 1           1 bread  milk         banana
## 2           2 milk   toilet paper <NA>
```

Longer = more rows

```
## # A tibble: 6 × 3
##   customer_id item_no item
##         <dbl> <chr>   <chr>
## 1           1 item_1  bread
## 2           1 item_2  milk
## 3           1 item_3  banana
## 4           2 item_1  milk
## 5           2 item_2  toilet paper
## 6           2 item_3  <NA>
```

pivot_longer()

- `data` (as usual)
- `cols`: columns to pivot into longer format
- `names_to`: name of the column where column names of pivoted variables go (character string)
- `values_to`: name of the column where data in pivoted variables go (character string)

```
pivot_longer(
  data,
  cols,
  names_to = "name",
  values_to = "value"
  )
```


Customers → purchases

```
purchases <- customers %>%
  pivot_longer(
    cols = item_1:item_3,  # variables item_1 to item_3
    names_to = "item_no",  # column names -> new column called item_no
    values_to = "item"     # values in columns -> new column called item
    )
```

```
purchases
```

```
## # A tibble: 6 × 3
##   customer_id item_no item
##         <dbl> <chr>   <chr>
## 1           1 item_1  bread
## 2           1 item_2  milk
## 3           1 item_3  banana
## 4           2 item_1  milk
## 5           2 item_2  toilet paper
## 6           2 item_3  <NA>
```

Why pivot?

Most likely, because the next step of your analysis needs it

```
prices
```

```
## # A tibble: 5 × 2
##   item         price
##   <chr>        <dbl>
## 1 avocado       0.5
## 2 banana        0.15
## 3 bread         1
## 4 milk          0.8
## 5 toilet paper  3
```

```
purchases %>%
  left_join(prices)
```

```
## # A tibble: 6 × 4
##   customer_id item_no item         price
##         <dbl> <chr>   <chr>        <dbl>
## 1           1 item_1  bread         1
## 2           1 item_2  milk          0.8
## 3           1 item_3  banana        0.15
## 4           2 item_1  milk          0.8
## 5           2 item_2  toilet paper  3
## 6           2 item_3  <NA>         NA
```


Purchases → customers

- `data` (as usual)
- `names_from`: which column in the long format contains the what should be column names in the wi format
- `values_from`: which column in the long format contains the what should be values in the new columns in the wide format

```
purchases %>%
  pivot_wider(
    names_from = item_no,
    values_from = item
  )
```

```
## # A tibble: 2 × 4
##   customer_id item_1 item_2       item_3
##         <dbl> <chr>  <chr>        <chr>
## 1           1 bread  milk         banana
## 2           2 milk   toilet paper <NA>
```

Example: Billboard

Consider the `billboard` dataset (contained in the `tidyr` package) which contains the rank of the song (in 2000) for each week after it first entered the list.

```
billboard
```

```
## # A tibble: 317 × 79
##    artist      track date.entered   wk1   wk2   wk3   wk4   wk5   wk6   wk7   wk8
##    <chr>       <chr> <date>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 2 Pac       Baby… 2000-02-26      87    82    72    77    87    94    99    NA
##  2 2Ge+her     The … 2000-09-02      91    87    92    NA    NA    NA    NA    NA
##  3 3 Doors D… Kryp… 2000-04-08      81    70    68    67    66    57    54    53
##  4 3 Doors D… Loser 2000-10-21      76    76    72    69    67    65    55    59
##  5 504 Boyz    Wobb… 2000-04-15      57    34    25    17    17    31    36    49
##  6 98^0        Give… 2000-08-19      51    39    34    26    26    19     2     2
##  7 A*Teens     Danc… 2000-07-08      97    97    96    95   100    NA    NA    NA
##  8 Aaliyah     I Do… 2000-01-29      84    62    51    41    38    35    35    38
##  9 Aaliyah     Try … 2000-03-18      59    53    38    28    21    18    16    14
## 10 Adams, Yo… Open… 2000-08-26      76    76    74    69    68    67    61    58
## # … with 307 more rows, and 68 more variables: wk9 <dbl>, wk10 <dbl>,
## #   wk11 <dbl>, wk12 <dbl>, wk13 <dbl>, wk14 <dbl>, wk15 <dbl>, wk16 <dbl>,
## #   wk17 <dbl>, wk18 <dbl>, wk19 <dbl>, wk20 <dbl>, wk21 <dbl>, wk22 <dbl>,
## #   wk23 <dbl>, wk24 <dbl>, wk25 <dbl>, wk26 <dbl>, wk27 <dbl>, wk28 <dbl>,
## #   wk29 <dbl>, wk30 <dbl>, wk31 <dbl>, wk32 <dbl>, wk33 <dbl>, wk34 <dbl>,
```

`billboard` data

If we want to identify songs that reach number 1 quickly, the data need to wrangled.

```
billboard %>%
  select(artist, track, date.entered, wk1, wk2) %>%
  pivot_longer(
  cols= c('wk1', 'wk2'),
  names_to = 'week',
  values_to = 'rank',
  values_drop_na = T)
```

```
## # A tibble: 629 × 5
##    artist       track                   date.entered week   rank
##    <chr>        <chr>                   <date>       <chr> <dbl>
##  1 2 Pac        Baby Don't Cry (Keep... 2000-02-26   wk1      87
##  2 2 Pac        Baby Don't Cry (Keep... 2000-02-26   wk2      82
##  3 2Ge+her      The Hardest Part Of ... 2000-09-02   wk1      91
##  4 2Ge+her      The Hardest Part Of ... 2000-09-02   wk2      87
##  5 3 Doors Down Kryptonite              2000-04-08   wk1      81
##  6 3 Doors Down Kryptonite              2000-04-08   wk2      70
##  7 3 Doors Down Loser                   2000-10-21   wk1      76
##  8 3 Doors Down Loser                   2000-10-21   wk2      76
##  9 504 Boyz     Wobble Wobble           2000-04-15   wk1      57
```

Billboard Data Analysis

```
billboard %>%
  pivot_longer(
  cols= starts_with('wk'),
  names_to = 'week',
  values_to = 'rank',
  values_drop_na = T) %>%
  mutate(week_numb= as.numeric(str_replace(week, 'wk',''))) %>%
  filter(rank == 1) %>%
  arrange(week_numb) %>%
  slice(1) %>%
  kable()
```

artist	track	date.entered	week	rank	week_numb
Madonna	Music	2000-08-12	wk6	1	6

Exercise

Determine which song in this dataset spent the most time at #1.

Solution: Code

```
billboard_long <- billboard %>%
  pivot_longer(
  cols= starts_with('wk'),
  names_to = 'week',
  values_to = 'rank',
  values_drop_na = TRUE) %>%
  mutate(week_numb =
    as.numeric(str_replace(week, 'wk',''))) %>%
  filter(rank == 1) %>%
  group_by(track) %>%
  tally() %>%
  arrange(desc(n))
```


Solution: Result

track	n
Independent Women Pa…	11
Maria, Maria	10
Come On Over Baby (A…	4
I Knew I Loved You	4
Music	4
Be With You	3
Doesn’t Really Matte…	3
Say My Name	3
Amazed	2
Incomplete	2
It’s Gonna Be Me	2
What A Girl Wants	2
Bent	1