GVPT Data Science Camp

Data Manipulation

Ryan Bookstein

August 26, 2026

University of Maryland

Learning Objectives

- Understanding basic operations in R

Learning Objectives

- Understanding basic operations in `R`
- Introduction to `dplyr`

Learning Objectives

- Understanding basic operations in R
- Introduction to dplyr
- Cleaning and manipulating data

Objects and Functions

Objects

- Create new objects with <-

```
x <- 6 * 8

x
```

```
x <- 2 * 15

x
```

Functions

- Several functions come pre-installed:

```
seq(15, 30)
```

Functions

- Several functions come pre-installed:

```
seq(15, 30)
```

- You can use functions to create objects:

```
x <- seq(15, 30)

x
```

Introduction to Data Manipulation

What is Data Manipulation?

- Data rarely arrive in exactly the form you need for analysis

What is Data Manipulation?

- Data rarely arrive in exactly the form you need for analysis
- Data manipulation is reshaping data to answer your question: picking out rows and columns, creating new variables, and computing summaries

What is Data Manipulation?

- Data rarely arrive in exactly the form you need for analysis
- Data manipulation is reshaping data to answer your question: picking out rows and columns, creating new variables, and computing summaries
- Today's tool: the `dplyr` package from the tidyverse

Exercise

- Load the packages we used yesterday:

```
library(tidyverse)
library(readxl)
```

Exercise

- Load the packages we used yesterday:

```
library(tidyverse)
library(readxl)
```

- Re-run yesterday's setup code (`read_excel()`, `rename()`, `select()`, and `mutate()`) to re-create the `cel` object, then view the data:

```
view(cel)
```

Data Types

- In `cel`:

- In `cel`:
- `fct` stands for factors, which `R` uses to represent categorical variables with fixed values — like `benchmark` (Below, Meets, Exceeds)

Data Types

- In `cel`:
- `fct` stands for factors, which `R` uses to represent categorical variables with fixed values — like `benchmark` (Below, Meets, Exceeds)
- `dbl` stands for doubles — like `les` and `ideology`

Data Types

- In `cel`:
- `fct` stands for factors, which `R` uses to represent categorical variables with fixed values — like `benchmark` (Below, Meets, Exceeds)
- `dbl` stands for doubles — like `les` and `ideology`
- `chr` stands for strings (character vectors) — like `member`, `state`, and `party`

Data Types

- `int` stands for integer

Data Types

- `int` stands for integer
- `dttm` stands for date-times (a date + time)

Data Types

- `int` stands for integer
- `dttm` stands for date-times (a date + time)
- `lgl` stands for logical, vectors that contain only `TRUE` or `FALSE`

- Five basic functions:

- Five basic functions:
  - `filter()`

- Five basic functions:
  - `filter()`
  - `arrange()`

- Five basic functions:
  - `filter()`
  - `arrange()`
  - `select()`

- Five basic functions:
  - `filter()`
  - `arrange()`
  - `select()`
  - `mutate()`

- Five basic functions:
  - `filter()`
  - `arrange()`
  - `select()`
  - `mutate()`
  - `summarize()`

Filtering and Arranging Data

Filtering Rows

```
filter(cel, state == "MD", congress > 110)
```

```
# A tibble: 64 x 14
  member            congress  year state   dem female votepct ideology
  <chr>                <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Bartlett, Roscoe       111  2009 MD        0      0      58   0.493
2 Cummings, Elijah       111  2009 MD        1      0      80  -0.438
3 Edwards, Donna         111  2009 MD        1      1      86  -0.562
4 Hoyer, Steny           111  2009 MD        1      0      74  -0.380
5 Kratovil, Frank        111  2009 MD        1      0      49  -0.0370
6 Ruppersberger, C.      111  2009 MD        1      0      72  -0.295
# i 58 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

- First argument must always be a data frame

- First argument must always be a data frame
- The arguments that follow describe which columns to operate on, using the variable names (without quotes)

- First argument must always be a data frame
- The arguments that follow describe which columns to operate on, using the variable names (without quotes)
- Output is always a new data frame

Filtering Rows

```
filter(cel, state %in% c("MD", "VA"))
```

```
# A tibble: 492 x 14
  member          congress  year state   dem female votepct ideology
  <chr>              <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Bauman, Robert        93  1973 MD        0      0      NA   0.533
2 Broyhill, Joel        93  1973 VA        0      0      56   0.159
3 Butler, M.            93  1973 VA        0      0      55   0.332
4 Byron, Goodloe        93  1973 MD        1      0      65  -0.0140
5 Daniel, Robert        93  1973 VA        0      0      47   0.318
6 Daniel, W.            93  1973 VA        1      0     100   0.161
# i 486 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

Filtering Rows

```
filter(cel, votepct > 50 & votepct < 60)
```

```
# A tibble: 3,023 x 14
  member          congress  year state   dem female votepct ideology
  <chr>              <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Abdnor, James         93  1973 SD        0      0      55    0.241
2 Abzug, Bella          93  1973 NY        1      1      56   -0.597
3 Annunzio, Frank       93  1973 IL        1      0      53   -0.398
4 Arends, Leslie        93  1973 IL        0      0      57    0.301
5 Ashbrook, John        93  1973 OH        0      0      57    0.612
6 Baker, LaMar          93  1973 TN        0      0      55    0.308
# i 3,017 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

Filtering Rows

```
filter(cel, votepct < 50 | votepct > 60)
```

```
# A tibble: 7,617 x 14
  member          congress  year state   dem female votepct ideology
  <chr>              <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Adams, Brock          93  1973 WA        1      0      85   -0.396
2 Addabbo, Joseph       93  1973 NY        1      0      75   -0.402
3 Albert, Carl          93  1973 OK        1      0      93   -0.392
4 Alexander, Bill       93  1973 AR        1      0     100   -0.410
5 Anderson, John        93  1973 IL        0      0      72    0.183
6 Anderson, Glenn       93  1973 CA        1      0      75   -0.269
# i 7,611 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

Exercise

- Find all member-Congress observations where the member ran unopposed (received 100% of the vote)

Exercise

- Find all member-Congress observations where the member ran unopposed (received 100% of the vote)
- Find all members from Maryland

Exercise

- Find all member-Congress observations where the member ran unopposed (received 100% of the vote)
- Find all members from Maryland
- Find all members from Maryland and Virginia

Exercise

- Find all member-Congress observations where the member ran unopposed (received 100% of the vote)
- Find all members from Maryland
- Find all members from Maryland and Virginia
- Find all member-Congress observations with a vote share greater than 60% and less than 70%

Exercise

- Find all member-Congress observations where the member ran unopposed (received 100% of the vote)
- Find all members from Maryland
- Find all members from Maryland and Virginia
- Find all member-Congress observations with a vote share greater than 60% and less than 70%
- Find all member-Congress observations with a vote share less than 60% or greater than 70%

Check Your Answers

```
filter(cel, votepct == 100)
```

```
# A tibble: 750 x 14
  member          congress  year state   dem female votepct ideology
  <chr>              <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Alexander, Bill       93  1973 AR        1      0     100 -0.410
2 Boland, Edward        93  1973 MA        1      0     100 -0.306
3 Breaux, John          93  1973 LA        1      0     100 -0.123
4 Brinkley, Jack        93  1973 GA        1      0     100 -0.0730
5 Burke, James          93  1973 MA        1      0     100 -0.400
6 Burleson, Omar        93  1973 TX        1      0     100  0.00200
# i 744 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

Check Your Answers

```
filter(cel, state == "MD")
```

```
# A tibble: 211 x 14
  member          congress  year state   dem female votepct ideology
  <chr>              <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Bauman, Robert        93  1973 MD        0      0      NA   0.533
2 Byron, Goodloe        93  1973 MD        1      0      65  -0.0140
3 Gude, Gilbert         93  1973 MD        0      0      64  -0.0280
4 Hogan, Lawrence       93  1973 MD        0      0      63   0.158
5 Holt, Marjorie        93  1973 MD        0      1      59   0.368
6 Long, Clarence        93  1973 MD        1      0      66  -0.274
# i 205 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

Check Your Answers

```
filter(cel, state %in% c("MD", "VA"))
```

```
# A tibble: 492 x 14
  member          congress  year state   dem female votepct ideology
  <chr>              <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Bauman, Robert        93  1973 MD        0      0      NA   0.533
2 Broyhill, Joel        93  1973 VA        0      0      56   0.159
3 Butler, M.            93  1973 VA        0      0      55   0.332
4 Byron, Goodloe        93  1973 MD        1      0      65  -0.0140
5 Daniel, Robert        93  1973 VA        0      0      47   0.318
6 Daniel, W.            93  1973 VA        1      0     100   0.161
# i 486 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

Check Your Answers

```
filter(cel, votepct > 60 & votepct < 70)
```

```
# A tibble: 3,423 x 14
  member             congress  year state   dem female votepct ideology
  <chr>                 <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Armstrong, William       93  1973 CO        0      0      62    0.508
2 Ashley, Thomas           93  1973 OH        1      0      69   -0.350
3 Aspin, Les               93  1973 WI        1      0      64   -0.320
4 Bafalis, L.              93  1973 FL        0      0      62    0.303
5 Barrett, William         93  1973 PA        1      0      66   -0.469
6 Bell, Alphonzo           93  1973 CA        0      0      61    0.183
# i 3,417 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

Check Your Answers

```
filter(cel, votepct < 60 | votepct > 70)
```

```
# A tibble: 7,142 x 14
  member          congress  year state   dem female votepct ideology
  <chr>              <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Abdnor, James         93  1973 SD        0      0      55    0.241
2 Abzug, Bella          93  1973 NY        1      1      56   -0.597
3 Adams, Brock          93  1973 WA        1      0      85   -0.396
4 Addabbo, Joseph       93  1973 NY        1      0      75   -0.402
5 Albert, Carl          93  1973 OK        1      0      93   -0.392
6 Alexander, Bill       93  1973 AR        1      0     100   -0.410
# i 7,136 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

- == is equal to

- `==` is equal to
- `!=` is not equal to

- `==` is equal to
- `!=` is not equal to
- `>` is greater than, `<` is less than

Important Features

- `==` is equal to
- `!=` is not equal to
- `>` is greater than, `<` is less than
- `>=` is greater than or equal to

Important Features

- `==` is equal to
- `!=` is not equal to
- `>` is greater than, `<` is less than
- `>=` is greater than or equal to
- `<=` is less than or equal to

Important Features

- | is or

Important Features

- `|` is or
- `&` is and

Important Features

- `|` is or
- `&` is and
- `%in%` is in

Arrange Rows with arrange()

```
arrange(cel, member, congress)
```

```
# A tibble: 11,606 x 14
  member            congress  year state   dem female votepct ideology
  <chr>                <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Abdnor, James           93  1973 SD        0      0      55    0.241
2 Abdnor, James           94  1975 SD        0      0      68    0.241
3 Abdnor, James           95  1977 SD        0      0      70    0.241
4 Abdnor, James           96  1979 SD        0      0      NA    0.241
5 Abercrombie, Neil       99  1985 HI        1      0      NA   -0.432
6 Abercrombie, Neil      102  1991 HI        1      0      60   -0.432
# i 11,600 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

Arrange Rows with arrange()

```
arrange(cel, member, desc(congress))
```

```
# A tibble: 11,606 x 14
  member            congress  year state   dem female votepct ideology
  <chr>                <dbl> <dbl> <chr> <dbl>  <dbl>   <dbl>    <dbl>
1 Abdnor, James           96  1979 SD        0      0      NA    0.241
2 Abdnor, James           95  1977 SD        0      0      70    0.241
3 Abdnor, James           94  1975 SD        0      0      68    0.241
4 Abdnor, James           93  1973 SD        0      0      55    0.241
5 Abercrombie, Neil      111  2009 HI        1      0      77   -0.432
6 Abercrombie, Neil      110  2007 HI        1      0      69   -0.432
# i 11,600 more rows
# i 6 more variables: seniority <dbl>, born <dbl>, les <dbl>,
#   bills_passed <dbl>, benchmark <fct>, party <chr>
```

Exercise

- Which member-Congress observations have the highest Legislative Effectiveness Score (`les`)?

Exercise

- Which member-Congress observations have the highest Legislative Effectiveness Score (`les`)?
- Which member-Congress observation passed the most bills through the House (`bills_passed`)?

Check Your Answers

```
arrange(cel, desc(les)) |>
  select(member, congress, les, bills_passed)
```

```
# A tibble: 11,606 x 4
  member            congress   les bills_passed
  <chr>                <dbl> <dbl>        <dbl>
1 Rangel, Charles        110  18.7           33
2 Murphy, John            96  17.9           31
3 Sullivan, Leonor        93  17.6           28
4 Sensenbrenner, F.      109  17.6           25
5 Murphy, John            95  17.2           31
6 Ullman, Al              96  16.4           18
7 Smith, Lamar           112  16.3           20
8 Young, Don             108  16.3           20
# i 11,598 more rows
```

Check Your Answers

```
arrange(cel, desc(bills_passed)) |>
  select(member, congress, les, bills_passed)
```

```
# A tibble: 11,606 x 4
  member            congress   les bills_passed
  <chr>                <dbl> <dbl>        <dbl>
1 Rangel, Charles        110  18.7           33
2 Murphy, John            95  17.2           31
3 Murphy, John            96  17.9           31
4 Sullivan, Leonor        93  17.6           28
5 Sullivan, Leonor        94  12.7           25
6 Sensenbrenner, F.      109  17.6           25
7 Young, Don             107  15.8           22
8 Miller, George         110  10.6           22
# i 11,598 more rows
```

Choosing and Creating Columns

Select Columns with select()

```
select(cel, member, congress, les)
```

```
# A tibble: 11,606 x 3
  member          congress     les
  <chr>              <dbl>   <dbl>
1 Abdnor, James         93 0.110
2 Abzug, Bella          93 0.762
3 Adams, Brock          93 1.24
4 Addabbo, Joseph       93 0.155
5 Albert, Carl          93 0.00103
6 Alexander, Bill       93 1.88
7 Anderson, John        93 0.289
8 Anderson, Glenn       93 0.447
# i 11,598 more rows
```

Select Columns with select()

```
select(cel, member:state)
```

```
# A tibble: 11,606 x 4
  member          congress  year state
  <chr>              <dbl> <dbl> <chr>
1 Abdnor, James         93  1973 SD
2 Abzug, Bella          93  1973 NY
3 Adams, Brock          93  1973 WA
4 Addabbo, Joseph       93  1973 NY
5 Albert, Carl          93  1973 OK
6 Alexander, Bill       93  1973 AR
7 Anderson, John        93  1973 IL
8 Anderson, Glenn       93  1973 CA
# i 11,598 more rows
```

Select Columns with select()

```
select(cel, -(votepct:ideology))
```

```
# A tibble: 11,606 x 12
  member          congress  year state   dem female seniority  born
    les
  <chr>              <dbl> <dbl> <chr> <dbl>  <dbl>     <dbl> <dbl>
    <dbl>
1 Abdnor, James         93  1973 SD        0      0         1  1923
    0.110
2 Abzug, Bella          93  1973 NY        1      1         2  1920
    0.762
3 Adams, Brock          93  1973 WA        1      0         5  1927 1.24
4 Addabbo, Joseph       93  1973 NY        1      0         7  1925
    0.155
# i 11,602 more rows
# i 3 more variables: bills_passed <dbl>, benchmark <fct>, party <chr>
```

Exercise

- Select only the `member`, `congress`, and `votepct` variables from `cel`

Exercise

- Select only the `member`, `congress`, and `votepct` variables from `cel`
- What does the `any_of()` function do? Why might it be useful with this vector?

```
vars <- c("member", "congress", "votepct")
```

Exercise

- Select only the `member`, `congress`, and `votepct` variables from `cel`
- What does the `any_of()` function do? Why might it be useful with this vector?

```
vars <- c("member", "congress", "votepct")
```

- What does this code produce?

```
select(cel, starts_with("b"))
```

Check Your Answers

```
select(cel, member, congress, votepct)
```

```
# A tibble: 11,606 x 3
  member          congress votepct
  <chr>              <dbl>   <dbl>
1 Abdnor, James         93      55
2 Abzug, Bella          93      56
3 Adams, Brock          93      85
4 Addabbo, Joseph       93      75
5 Albert, Carl          93      93
6 Alexander, Bill       93     100
7 Anderson, John        93      72
8 Anderson, Glenn       93      75
# i 11,598 more rows
```

Check Your Answers

`any_of()` selects any of the named columns that exist, without erroring on ones that don't:

```
vars <- c("member", "congress", "votepct")

select(cel, any_of(vars))
```

```
# A tibble: 11,606 x 3
  member          congress votepct
  <chr>              <dbl>   <dbl>
1 Abdnor, James         93      55
2 Abzug, Bella          93      56
3 Adams, Brock          93      85
4 Addabbo, Joseph       93      75
5 Albert, Carl          93      93
6 Alexander, Bill       93     100
# i 11,600 more rows
```

Check Your Answers

All columns whose names start with "b": `born`, `bills_passed`, and `benchmark`

```
select(cel, starts_with("b"))
```

```
# A tibble: 11,606 x 3
   born bills_passed benchmark
  <dbl>        <dbl> <fct>
1  1923            0 Below
2  1920            1 Meets
3  1927            2 Meets
4  1925            0 Below
5  1908            0 Below
6  1934            2 Meets
7  1922            0 Meets
8  1913            0 Meets
# i 11,598 more rows
```

Add New Variables with mutate()

```
mutate(cel, age = year - born) |>
  select(member, congress, year, born, age)
```

```
# A tibble: 11,606 x 5
  member          congress  year  born   age
  <chr>              <dbl> <dbl> <dbl> <dbl>
1 Abdnor, James         93  1973  1923    50
2 Abzug, Bella          93  1973  1920    53
3 Adams, Brock          93  1973  1927    46
4 Addabbo, Joseph       93  1973  1925    48
5 Albert, Carl          93  1973  1908    65
6 Alexander, Bill       93  1973  1934    39
7 Anderson, John        93  1973  1922    51
8 Anderson, Glenn       93  1973  1913    60
# i 11,598 more rows
```

Select, Manipulate, and Add New Variables with transmute()

```
transmute(cel, member, congress, age = year - born)
```

```
# A tibble: 11,606 x 3
  member          congress   age
  <chr>              <dbl> <dbl>
1 Abdnor, James         93    50
2 Abzug, Bella          93    53
3 Adams, Brock          93    46
4 Addabbo, Joseph       93    48
5 Albert, Carl          93    65
6 Alexander, Bill       93    39
7 Anderson, John        93    51
8 Anderson, Glenn       93    60
# i 11,598 more rows
```

Summaries and the Pipe

Create Summaries with summarize()

```
summarize(cel, avg_bills = mean(bills_passed),
          avg_seniority = mean(seniority))
```

```
# A tibble: 1 x 2
  avg_bills avg_seniority
      <dbl>         <dbl>
1      1.48          5.27
```

Creating Grouped Summaries with group_by() and summarize()

```
cel_party <- group_by(cel, party)

summarize(cel_party, avg_bills = mean(bills_passed),
          avg_seniority = mean(seniority))
```

```
# A tibble: 2 x 3
  party      avg_bills avg_seniority
  <chr>          <dbl>         <dbl>
1 Democrat        1.57          5.75
2 Republican      1.38          4.72
```

Exercise

- Calculate each member's age in each Congress

Exercise

- Calculate each member's age in each Congress
- Find each member's average number of bills passed across all Congresses in the `cel` dataset

Exercise

- Calculate each member's age in each Congress
- Find each member's average number of bills passed across all Congresses in the `cel` dataset
- Find the member with the lowest average bills passed across these Congresses (careful: many members tie at zero)

Exercise

- Calculate each member's age in each Congress
- Find each member's average number of bills passed across all Congresses in the `cel` dataset
- Find the member with the lowest average bills passed across these Congresses (careful: many members tie at zero)
- Find the member with the highest average bills passed across these Congresses

Check Your Answers

```
mutate(cel, age = year - born) |>
  select(member, congress, year, born, age)
```

```
# A tibble: 11,606 x 5
  member          congress  year  born   age
  <chr>              <dbl> <dbl> <dbl> <dbl>
1 Abdnor, James         93  1973  1923    50
2 Abzug, Bella          93  1973  1920    53
3 Adams, Brock          93  1973  1927    46
4 Addabbo, Joseph       93  1973  1925    48
5 Albert, Carl          93  1973  1908    65
6 Alexander, Bill       93  1973  1934    39
7 Anderson, John        93  1973  1922    51
8 Anderson, Glenn       93  1973  1913    60
# i 11,598 more rows
```

Check Your Answers

```
cel_member <- group_by(cel, member)

avg_bills_by_member <- summarize(cel_member,
  avg_bills = mean(bills_passed))

avg_bills_by_member
```

```
# A tibble: 2,233 x 2
  member               avg_bills
  <chr>                    <dbl>
1 Abdnor, James            0.5
2 Abercrombie, Neil        0.545
3 Abraham, Ralph           1.33
4 Abzug, Bella             1.5
5 Acevedo-Vila, Anibal     2
# i 2,228 more rows
```

Check Your Answers

```
arrange(avg_bills_by_member, avg_bills)
```

```
# A tibble: 2,233 x 2
  member           avg_bills
  <chr>                <dbl>
1 Albert, Carl             0
2 Ammerman, Joseph         0
3 Amo, Gabe                0
4 Arends, Leslie           0
5 Ashbrook, John           0
6 Austria, Steve           0
7 Bacchus, Jim             0
8 Badillo, Herman          0
# i 2,225 more rows
```

Check Your Answers

```
arrange(avg_bills_by_member, desc(avg_bills))
```

```
# A tibble: 2,233 x 2
  member           avg_bills
  <chr>                <dbl>
1 Sullivan, Leonor      26.5
2 Rogers, Paul          19.7
3 Murphy, John          17
4 Ullman, Al            14.8
5 Rodino, Peter         12.1
6 Mahon, George         12
7 Hays, Wayne           11
8 Katko, John           10.5
# i 2,225 more rows
```

Utilize Multiple Operations with the Pipe

- We created a lot of different objects during our calculations

Utilize Multiple Operations with the Pipe

- We created a lot of different objects during our calculations
- Can we avoid creating unneeded objects?

Utilize Multiple Operations with the Pipe

- We created a lot of different objects during our calculations
- Can we avoid creating unneeded objects?
- We can utilize a feature of the tidyverse: the pipe

Utilize Multiple Operations with the Pipe

- Pipe can be read as:

```
cel |>
  group_by(state) |>
  summarize(avg_les = mean(les)) |>
  arrange(desc(avg_les))
```

Utilize Multiple Operations with the Pipe

- Pipe can be read as:
- Take this |> (and then...)

```
cel |>
  group_by(state) |>
  summarize(avg_les = mean(les)) |>
  arrange(desc(avg_les))
```

Utilize Multiple Operations with the Pipe

- Pipe can be read as:
- Take this |> (and then...)
- do this |> (and then...)

```
cel |>
  group_by(state) |>
  summarize(avg_les = mean(les)) |>
  arrange(desc(avg_les))
```

Utilize Multiple Operations with the Pipe

- Pipe can be read as:
- Take this |> (and then...)
- do this |> (and then...)
- do this

```
cel |>
  group_by(state) |>
  summarize(avg_les = mean(les)) |>
  arrange(desc(avg_les))
```

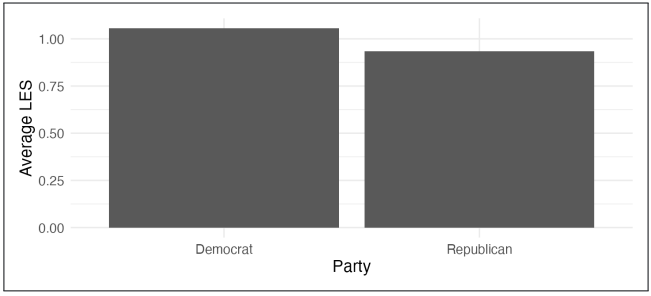
Utilize Multiple Operations with the Pipe

```
cel |>
  group_by(state) |>
  summarize(avg_les = mean(les)) |>
  arrange(desc(avg_les))
```

```
# A tibble: 56 x 2
  state avg_les
  <chr>   <dbl>
1 AK       4.87
2 DC       2.65
3 OR       1.54
4 MT       1.52
5 UT       1.50
6 NM       1.49
7 NJ       1.40
# i 49 more rows
```

Utilize Multiple Operations with the Pipe

```
cel |>
  group_by(party) |>
  summarize(avg_les = mean(les)) |>
  ggplot(aes(x = party, y = avg_les)) +
  geom_col() +
  theme_minimal() +
  labs(x = "Party", y = "Average LES")
```



A Note on the Pipe

- Base pipe: `|>`

A Note on the Pipe

- Base pipe: `|>`
  - Can be used without loading any packages

A Note on the Pipe

- Base pipe: |>
  - Can be used without loading any packages
  - Relatively new: introduced in 2021

A Note on the Pipe

- Tidyverse pipe: %>%

A Note on the Pipe

- Tidyverse pipe: `%>%`
  - Must load `dplyr` or `magrittr` to use

- Calculate the average Legislative Effectiveness Score for Maryland members serving in Congresses after the 110th (2009 onward). Display only the member and average LES variables

Exercise

- Calculate the average Legislative Effectiveness Score for Maryland members serving in Congresses after the 110th (2009 onward). Display only the member and average LES variables
- Plot the results

Check Your Answers

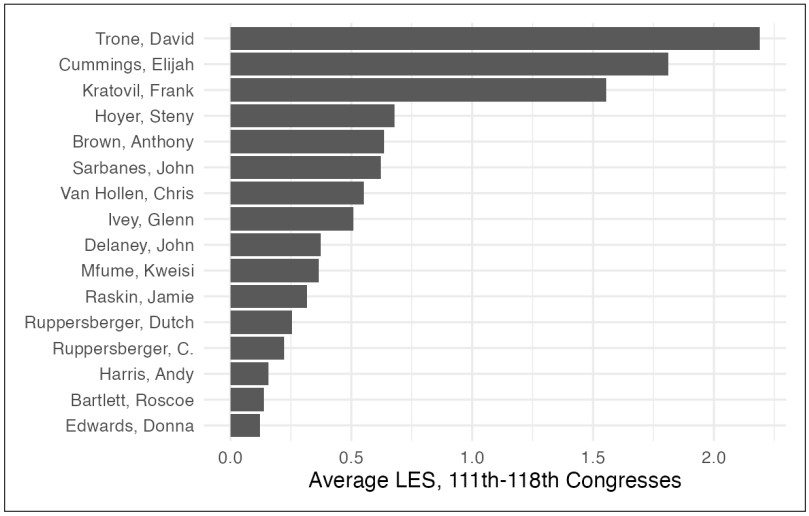
```
cel |>
  filter(state == "MD", congress > 110) |>
  group_by(member) |>
  summarize(avg_les = mean(les))
```

```
# A tibble: 16 x 2
  member           avg_les
  <chr>              <dbl>
1 Bartlett, Roscoe   0.139
2 Brown, Anthony     0.634
3 Cummings, Elijah   1.81
4 Delaney, John      0.374
5 Edwards, Donna     0.122
6 Harris, Andy       0.156
7 Hoyer, Steny       0.679
# i 9 more rows
```

- `fct_reorder()` sorts the bars: it reorders the `member` categories by `avg_les`, so the chart reads top to bottom

```
cel |>
  filter(state == "MD", congress > 110) |>
  group_by(member) |>
  summarize(avg_les = mean(les)) |>
  ggplot(aes(x = avg_les, y = fct_reorder(member, avg_les))) +
  geom_col() +
  theme_minimal() +
  labs(x = "Average LES, 111th-118th Congresses", y = NULL)
```

Check Your Answers



Summary

- Familiarized yourself with basic R syntax

Summary

- Familiarized yourself with basic R syntax
- Learned how to manipulate your data

Summary

- Familiarized yourself with basic R syntax
- Learned how to manipulate your data
- Wrote concise code that is easy to follow

- On Thursday, you will apply the skills learned in this course to an idea that interests you. In doing so, you will need to find a dataset that:

- On Thursday, you will apply the skills learned in this course to an idea that interests you. In doing so, you will need to find a dataset that:
- Is relevant to your research interests

Homework

- On Thursday, you will apply the skills learned in this course to an idea that interests you. In doing so, you will need to find a dataset that:
- Is relevant to your research interests
- Contains continuous and categorical variables

Homework

- On Thursday, you will apply the skills learned in this course to an idea that interests you. In doing so, you will need to find a dataset that:
- Is relevant to your research interests
- Contains continuous and categorical variables
- If needed, reach out to me for dataset recommendations

Questions?