Lab 2 - Data wrangling

Jakob Bickley

2026-02-04

```r
library(tidyverse)
```

```
Warning: package 'tidyverse' was built under R version 4.5.2
```

```
Warning: package 'readr' was built under R version 4.5.2
```

```
Warning: package 'dplyr' was built under R version 4.5.2
```

```
Warning: package 'stringr' was built under R version 4.5.2
```

```
Warning: package 'lubridate' was built under R version 4.5.2
```

Questions

Part 1

Question 1

```r
midwest <- midwest
midwest |> group_by(state)|> summarize(n = n()) |> arrange(desc(n))
```

```
# A tibble: 5 x 2
  state     n
  <chr> <int>
1 IL      102
2 IN       92
3 OH       88
4 MI       83
5 WI       72
```

Illinois has the greatest number of counties at 102, and Wisconsin has the lowest at 72, across Midwest states.

Question 2

```r
midwest_duplicate <- midwest |> group_by(county) |> summarize(count = n()) |>
  filter(count >= 2) |> arrange(desc(count))
midwest_duplicate
```

```
# A tibble: 70 x 2
   county     count
   <chr>      <int>
 1 CRAWFORD       5
 2 JACKSON        5
 3 MONROE         5
 4 ADAMS          4
 5 BROWN          4
 6 CLARK          4
 7 CLINTON        4
 8 JEFFERSON      4
 9 LAKE           4
10 WASHINGTON     4
# i 60 more rows
```

Question 3

```
midwest |> filter(popdensity > 25000) |> select(county, state,
  popdensity, poptotal, area) |>
  arrange(desc(popdensity))
```

```
# A tibble: 9 x 5
  county    state popdensity poptotal  area
  <chr>     <chr>      <dbl>    <int> <dbl>
1 COOK      IL        88018.  5105067 0.058
2 MILWAUKEE WI        63952.   959275 0.015
3 WAYNE     MI        60334.  2111687 0.035
4 CUYAHOGA  OH        54313.  1412140 0.026
5 DU PAGE   IL        39083.   781666 0.02
6 MARION    IN        34659.   797159 0.023
7 HAMILTON  OH        34649.   866228 0.025
8 FRANKLIN  OH        28278.   961437 0.034
9 MACOMB    MI        25621.   717400 0.028
```

```
midwest |> select(county, state, popdensity, poptotal, area) |>
  filter(popdensity == max(popdensity))
```

```
# A tibble: 1 x 5
  county state popdensity poptotal  area
  <chr>  <chr>      <dbl>    <int> <dbl>
1 COOK   IL        88018.  5105067 0.058
```

Question 4

The distribution of population density of counties is unimodal and extremely right-skewed. A typical Midwestern county has population density of 1156.20833 people per unit area. The middle 50% of the counties have population densities between c(`25%` = 622.407407) to c(`75%` = 2330) people per unit area. # Numbers are inline code # Used AI to get quantile() function

Question 5

```r
midwest |>mutate(metro = if_else(inmetro == 1, "Yes", "No")) |> group_by(state, metro) |>
  summarise(n = n()) |>
  group_by(state) |>
  mutate(proportion_urban = n/sum(n)) |>
  select(state, metro, proportion_urban)
```

```
`summarise()` has grouped output by 'state'. You can override using the
`.groups` argument.
```

```
# A tibble: 10 x 3
# Groups:   state [5]
   state metro proportion_urban
   <chr> <chr>            <dbl>
 1 IL    No               0.725
 2 IL    Yes              0.275
 3 IN    No               0.598
 4 IN    Yes              0.402
 5 MI    No               0.699
 6 MI    Yes              0.301
 7 OH    No               0.545
 8 OH    Yes              0.455
 9 WI    No               0.722
10 WI    Yes              0.278
```

Question 6

```
midwest |> select(county, state, percbelowpoverty, percollege) |>
  filter(percbelowpoverty > 45, percollege <8)
```

```
# A tibble: 1 x 4
  county    state percbelowpoverty percollege
  <chr>     <chr>            <dbl>      <dbl>
1 MENOMINEE WI                48.7       7.34
```

```
midwest |> select(county, state, percbelowpoverty, percollege) |>
  filter(percbelowpoverty <20 , percollege >40)
```

```
# A tibble: 5 x 4
  county    state percbelowpoverty percollege
  <chr>     <chr>            <dbl>      <dbl>
1 CHAMPAIGN IL               15.6        41.3
2 DU PAGE   IL                2.71       42.8
3 HAMILTON  IN                3.59       42.1
4 WASHTENAW MI               12.2        48.1
5 DANE      WI               10.5        43.6
```

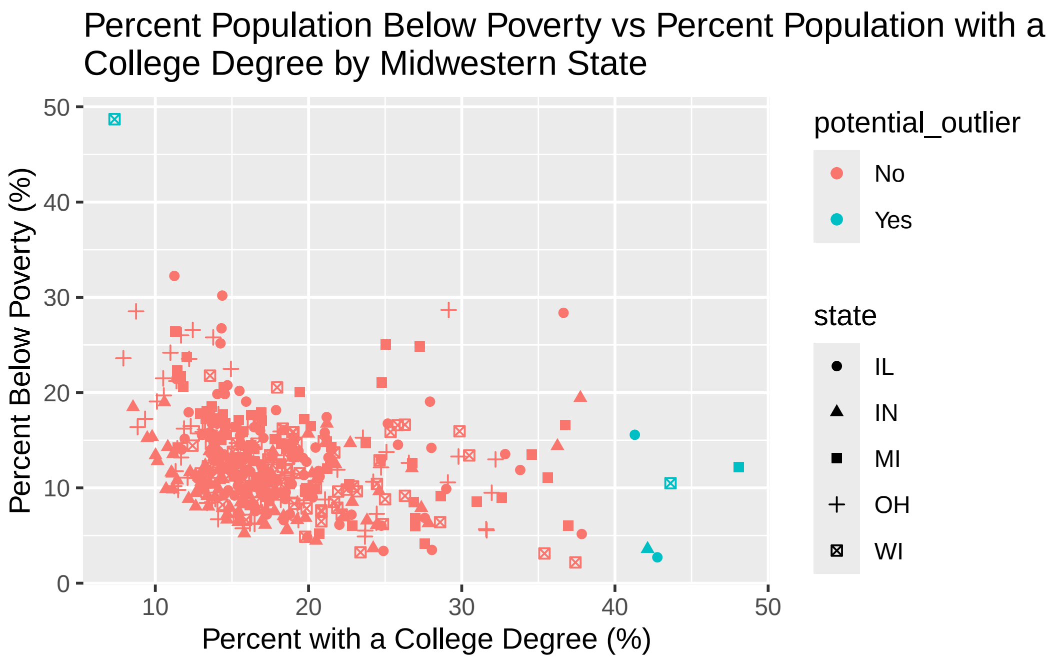
```
midwest |> select(county, state, percbelowpoverty, percollege) |>
  filter((percbelowpoverty <20 & percollege >40) | (percbelowpoverty > 45 & percollege <8))
```

```
# A tibble: 6 x 4
  county    state percbelowpoverty percollege
  <chr>     <chr>            <dbl>      <dbl>
1 CHAMPAIGN IL               15.6       41.3
2 DU PAGE   IL                2.71      42.8
3 HAMILTON  IN                3.59      42.1
4 WASHTENAW MI               12.2       48.1
5 DANE      WI               10.5       43.6
6 MENOMINEE WI               48.7        7.34
```

```
midwest_outliers <- midwest |> select(county, state, percbelowpoverty, percollege) |>
  mutate(potential_outlier = if_else((percbelowpoverty <20 & percollege >40) |
  (percbelowpoverty > 45 & percollege <8), "Yes", "No")) |>
  arrange(potential_outlier)
midwest_outliers
```

```
# A tibble: 437 x 5
   county    state percbelowpoverty percollege potential_outlier
   <chr>     <chr>            <dbl>      <dbl> <chr>
 1 ADAMS     IL               13.2        19.6 No
 2 ALEXANDER IL               32.2        11.2 No
 3 BOND      IL               12.1        17.0 No
 4 BOONE     IL                7.21       17.3 No
 5 BROWN     IL               13.5        14.5 No
 6 BUREAU    IL               10.4        18.9 No
 7 CALHOUN   IL               15.1        11.9 No
 8 CARROLL   IL               11.7        16.2 No
 9 CASS      IL               13.9        14.1 No
10 CHRISTIAN IL               11.7        13.6 No
# i 427 more rows
```

```r
midwest_outliers |>  ggplot(aes(x = percollege, y= percbelowpoverty, shape = state,
color = potential_outlier)) +
geom_point() +
labs(
y = "Percent Below Poverty (%)",
x = "Percent with a College Degree (%)",
title = "Percent Population Below Poverty vs Percent Population with a
College Degree by Midwestern State")
```



Question 7

```
midwest |> group_by(state) |>
  summarise(state_pop = sum(poptotal)) |>
  arrange(desc(state_pop))
```

```
# A tibble: 5 x 2
  state state_pop
  <chr>     <int>
1 IL     11430602
2 OH     10847115
3 MI      9295297
4 IN      5544159
5 WI      4891769
```

```
midwest |> group_by(state) |> summarise(state_pop = sum(poptotal)) |>
  mutate(proportion_pop = state_pop/sum(state_pop)) |>
  arrange(desc(proportion_pop))
```

```
# A tibble: 5 x 3
  state state_pop proportion_pop
  <chr>     <int>          <dbl>
1 IL     11430602          0.272
2 OH     10847115          0.258
3 MI      9295297          0.221
4 IN      5544159          0.132
5 WI      4891769          0.116
```

Illinois is most populous with 27.2% of the population of the Midwest living there, and Wisconsin is least populous with 11.6% of the population living there.

Question 8

```r
state_poverty <- midwest |> group_by(state) |>
  summarise(mean_percbelowpoverty = mean(percbelowpoverty)) |>
  select(state, mean_percbelowpoverty)
state_poverty |> arrange(mean_percbelowpoverty)
```

```
# A tibble: 5 x 2
  state mean_percbelowpoverty
  <chr>                 <dbl>
1 IN                     10.3
2 WI                     11.9
3 OH                     13.0
4 IL                     13.1
5 MI                     14.2
```

Michigan has the highest mean percent below poverty at 14.2% and Indiana has the lowest at 10.3%

Part 2

Question 9

```r
df <- tibble(
  var_1 = c(10, 20, 30, 40, 50),
  var_2 = c("Pizza", "Burger", "Pizza", "Pizza", "Burger"),
  var_3 = c("Apple", "Apple", "Pear", "Pear", "Banana")
)

df |>
  arrange(var_2)
```

```
# A tibble: 5 x 3
  var_1 var_2  var_3
  <dbl> <chr>  <chr>
1    20 Burger Apple
2    50 Burger Banana
3    10 Pizza  Apple
4    30 Pizza  Pear
5    40 Pizza  Pear
```

```r
df |>
  group_by(var_2)
```

```
# A tibble: 5 x 3
# Groups:   var_2 [2]
  var_1 var_2  var_3
  <dbl> <chr>  <chr>
1    10 Pizza  Apple
2    20 Burger Apple
3    30 Pizza  Pear
4    40 Pizza  Pear
5    50 Burger Banana
```

```r
df |>
  group_by(var_2) |>
  summarize(mean_var_1 = mean(var_1))
```

```
# A tibble: 2 x 2
  var_2  mean_var_1
  <chr>       <dbl>
1 Burger       35
2 Pizza        26.7
```

```r
df |>
  group_by(var_2, var_3) |>
  summarize(mean_var_1 = mean(var_1))
```

```
`summarise()` has grouped output by 'var_2'. You can override using the
`.groups` argument.
```

```
# A tibble: 4 x 3
# Groups:   var_2 [2]
  var_2  var_3  mean_var_1
  <chr>  <chr>       <dbl>
1 Burger Apple          20
2 Burger Banana         50
3 Pizza  Apple          10
4 Pizza  Pear           35
```

```r
df |>
  group_by(var_2, var_3) |>
  summarize(mean_var_1 = mean(var_1), .groups = "drop")
```

```
# A tibble: 4 x 3
  var_2  var_3  mean_var_1
  <chr>  <chr>       <dbl>
1 Burger Apple          20
2 Burger Banana         50
3 Pizza  Apple          10
4 Pizza  Pear           35
```

```r
df |>
  group_by(var_2, var_3) |>
  summarize(mean_var_1 = mean(var_1), .groups = "drop")
```

```
# A tibble: 4 x 3
  var_2  var_3  mean_var_1
  <chr>  <chr>       <dbl>
1 Burger Apple          20
2 Burger Banana         50
3 Pizza  Apple          10
4 Pizza  Pear           35
```

```
df |>
  group_by(var_2, var_3) |>
  mutate(mean_var_1 = mean(var_1))
```

```
# A tibble: 5 x 4
# Groups:   var_2, var_3 [4]
  var_1 var_2  var_3  mean_var_1
  <dbl> <chr>  <chr>       <dbl>
1    10 Pizza  Apple          10
2    20 Burger Apple          20
3    30 Pizza  Pear           35
4    40 Pizza  Pear           35
5    50 Burger Banana         50
```

A) This code chunk arranges in ascending order (Alphabetical because its a string) of the var_2 column.

B) This code alone does not change the displayed dataframe, unlike A) it doesn't reorder anything it stays in order originally input, however, it does group the var_2 into groups of the same string so the pizzas together and the burgers together, and allows for further analysis by the group like averages and max in a particular group with the summarise()

C) This first considers the groups of var_2 so the pizzas and burgers get put together, then it asks what is the mean of the var_1 for the pizzas and for the burgers and outputs it.

D) This groups first into groups with distinct combinations of Var_2 and 3, so for example Burgers with Apples, Burger's with Bananas, and so on. Within these distinct groups it calcualtes the mean of var_1. We also get a message saying it grouped outputs by var_2, which is what group remains after the summary operation, so future functions on the group apply as if its grouped by var_2.

E) Fundamentally the same as D, but does not generate the message because we have used .groups = "drop" to remove all remaining groupings, so you could not for example use another summarize() because there are no more groupings. .groups() controls what grouping if any remains after the summarize operation as it consolidates the rows.

F) The first one is the same as E), but the 2nd one does something quite different as the mutate() does not operate on the groups, it instead creates a new variable that takes the mean(var_1) for EACH row, so essentially in this case does nothing but copy var_1 because there is only 1 entry per row. Mutate() is more used if we wanted to say add 2 vars in each row, and not consider groups, summaries() acts on groups.