Lab 2 - Data wrangling

Edmond Niu

```r
library(tidyverse)
```

Part 2

Question 1

IL has the highest number of counties, at a count of 102. WI has the lowest number of counties, at a count of 72.

```r
midwest |>
  count(state)
```

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

Question 2

Only 3 counties are located in all 5 states in this dataset (Crawford, Jackson, and Monroe).

```
midwest |>
  count(county) |>
  filter(n==5)
```

```
# A tibble: 3 x 2
  county       n
  <chr>    <int>
1 CRAWFORD     5
2 JACKSON      5
3 MONROE       5
```

Question 3

a.

```r
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
```

b.

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

```
# 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 people per unit area. The middle 50% of the counties have population densities between 622 to 2330 people per unit area.

```r
midwest |>
  summarize(
    median = median(popdensity),
    q1 = quantile(popdensity, 0.25),
    q3 = quantile(popdensity, 0.75)
  )
```

```
# A tibble: 1 x 3
  median    q1    q3
   <dbl> <dbl> <dbl>
1  1156.  622.  2330
```

Question 5

Proportion of counties in urban areas in each state:

IL: 0.27

IN: 0.40

MI: 0.30

OH: 0.45

WI: 0.28

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

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

```
# A tibble: 5 x 4
# Groups:   state [5]
  state metro count proportion_inmetro
  <chr> <chr> <int>              <dbl>
1 IL    Yes      28              0.275
2 IN    Yes      37              0.402
3 MI    Yes      25              0.301
4 OH    Yes      40              0.455
5 WI    Yes      20              0.278
```

Question 6

a.

```r
midwest |>
  filter(percbelowpoverty == max(percbelowpoverty)) |>
  select(county,state,percbelowpoverty, percollege)
```

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

b.

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

```
# 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
```

c.

```r
midwest |>
  filter(percollege > 40 | percbelowpoverty == max(percbelowpoverty)) |>
  select(county,state,percbelowpoverty, percollege)
```

```
# 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
```

d.

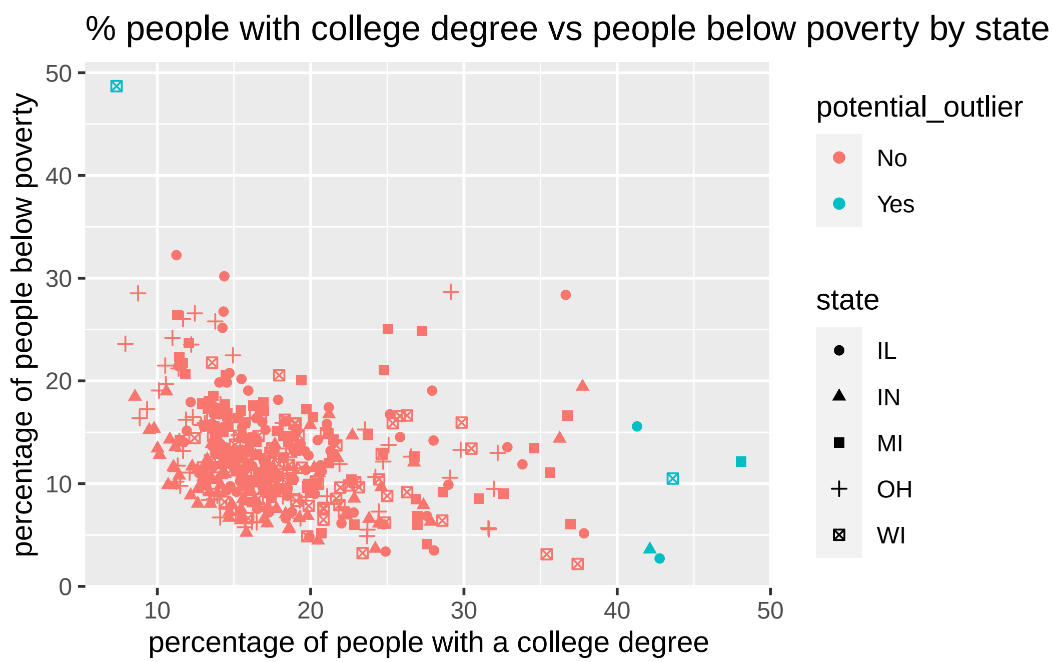
```r
midwest <- midwest |>
  mutate(potential_outlier = if_else(percollege > 40 |
  percbelowpoverty == max(percbelowpoverty), "Yes", "No"))

midwest |>
  select(county, state, percbelowpoverty, percollege, potential_outlier) |>
  arrange(potential_outlier)
```

```
# 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
```

e.

```r
ggplot(midwest, aes(x=percollege, y=percbelowpoverty, color = potential_outlier,
      shape=state)) +
  geom_point() +
  labs(
    x = "percentage of people with a college degree",
    y = "percentage of people below poverty",
    title = "% people with college degree vs people below poverty by state"
  )
```


% people with college degree vs people below poverty by state
percentage of people below poverty
percentage of people with a college degree
0
10
20
30
40
50
10
20
30
40
50
potential_outlier
No
Yes
state
IL
IN
MI
OH
WI

Question 7

a.

```
state_population <- midwest |>
  group_by(state) |>
  summarize(total_population = sum(poptotal))

state_population |>
  arrange(desc(total_population))
```

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

b.

```
state_population |>
  mutate(propOf_totalPopulation = total_population / sum(total_population)) |>
  arrange(desc(propOf_totalPopulation))
```

```
# A tibble: 5 x 3
  state total_population propOf_totalPopulation
  <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
```

c. IL is the most populous Midwestern state with 27.2% of the Midwest population living there. WI is the least populous Midwestern state with 11.6% of the Midwest population living there.

Question 8

The state that has the lowest average percentage below poverty across its counties is IN (Indiana). The state that has the highest average percentage below poverty across its countries is MI (Michigan).

```r
state_poverty <- midwest |>
  group_by(state) |>
  summarize(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
```

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
```

```
# A tibble: 5 x 3
  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
```

a. The code chunk below arranges the values in column var_2 by alphabetical order. Arrange() orders rows using column values in either ascending or descending fashion.

```r
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
```

b. The following code groups the data by value in var_2. The group_by() function groups the data by one or more variables (in this instance, var_2). It's different from arrange in part a in that arrange(var_2) groups the data by column var_2 but group_by(var_2) groups the data by column var_3 value.

```
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
```

c. This code chunk groups the data by value in var_2 into two groups (Burger and Pizza). It then takes the corresponding var_1 value and takes the mean of all the var_1 values from all entries for Burgers and all entries for Pizza and computes two means (for the two groups in var_2).

```
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
```

d. This code chunk groups the data by value in var_2 AND var_3 into 4 groups (all combinations of var_2 values with var_3 values). It then takes the corresponding var_1 value(s) from all entries of each combination/grouping of var_2 and var_3 values, and computes the mean of the var_1 value(s).

```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
```

e. This code chunk does the exact same thing as part d, but the .groups = "drop" drops all levels of grouping. So this code chunk groups the data by value in var_2 AND var_3 into 4 groups (all combinations of var_2 values with var_3 values). It then takes the corresponding var_1 value(s) from all entries of each combination/grouping of var_2 and var_3 values, and computes the mean of the var_1 value(s), BUT at the end, the groupings of var_2, var_3 are dropped.

```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
```

f. The first pipeline does exactly what part e is doing. It groups the data by var_2 and var_3 and calculates var_1 means for each grouping made. It then gets rid of the grouping. However, the second pipeline is different. It also groups by var_2 and var_3 but it mutates instead of summarizes. What this means is that each group is NOT summarized down to one row (hence the two rows mean_var_1 for Pizza and Pear). This is why there are 5 rows instead of 4 for the second pipeline. All mutate is doing is changing the values of mean_var_1 to reflect every single row created by the group_by(var_2, var_3).

```
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
```

Question 10

No answer needed here! Just select questions and pages to indicate where your responses are located when you upload your lab PDF to Gradescope and you'll get full points on this question.