Chapter 7.11~7.15

Contents


```
library("dplyr")
```

```
##
##          : 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library("tidyverse")
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v stringr 1.4.0
## v tidyr   1.1.3     v forcats 0.5.1
## v readr   2.0.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library("dslabs")
data("murders")
library("ggthemes")
library("ggrepel")
r <- murders %>%
  summarize(rate = sum(total) / sum(population) * 10^6) %>%
  pull(rate)
p <- murders %>% ggplot(aes(population/10^6, total, label = abb)) +
  geom_text(nudge_x = 0.05) +
  scale_x_log10() +
```

```
  scale_y_log10() +
  xlab("Populations in millions (log scale)") +
  ylab("Total number of murders (log scale)") +
  ggtitle("US Gun Murders in 2010") +
  geom_abline(intercept = log10(r), lty = 2, color = "darkgrey") +
  geom_point(aes(col=region), size = 3) +
  scale_color_discrete(name = "Region")
```

0.1 7.11 Add-on packages

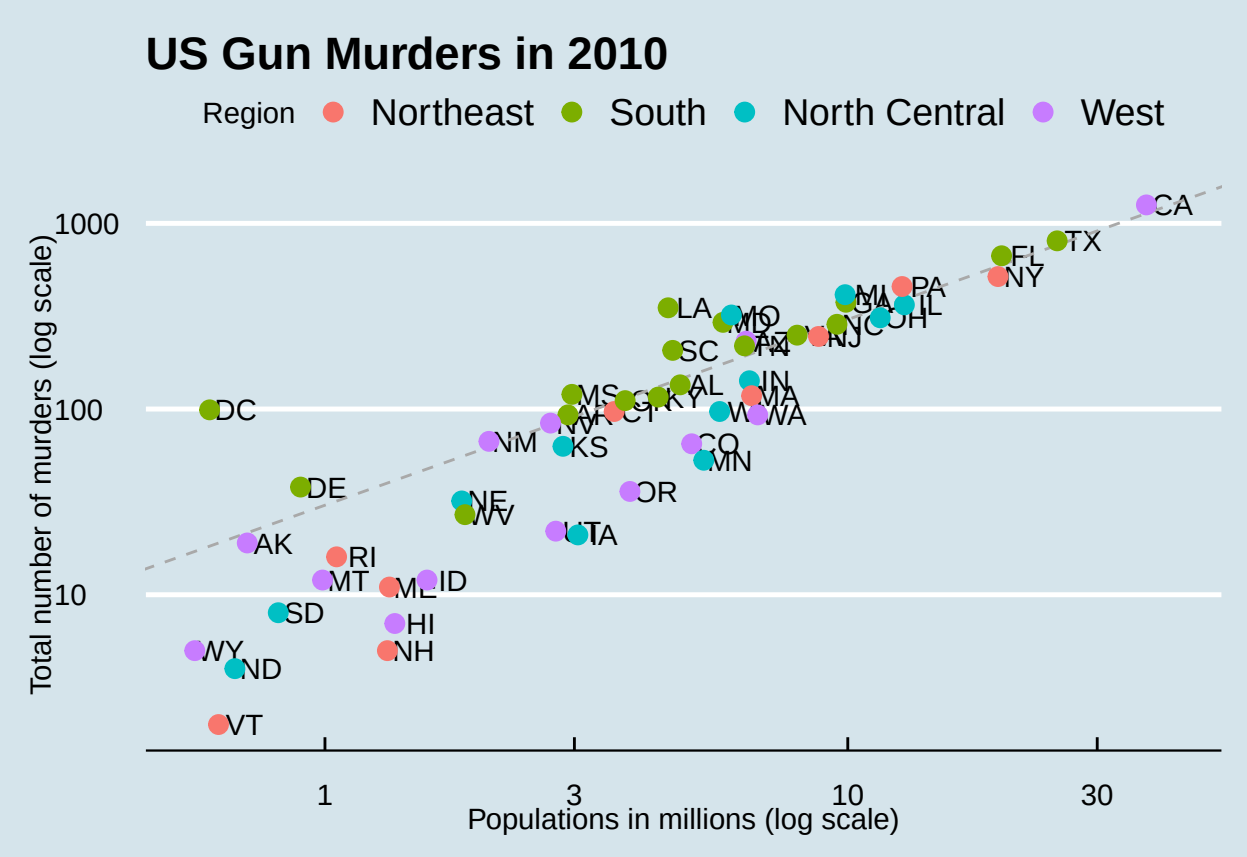
The power of ggplot2 is augmented further due to the availability of add-on packages. The remaining changes needed to put the finishing touches on our plot require the ggthemes and ggrepel packages.

The style of a ggplot2 graph can be changed using the theme functions. Several themes are included as part of the ggplot2 package. In fact, for most of the plots in this book, we use a function in the dslabs package that automatically sets a default theme:

```
ds_theme_set()
```

Many other themes are added by the package ggthemes. Among those are the theme_economist theme that we used. After installing the package, you can change the style by adding a layer like this:

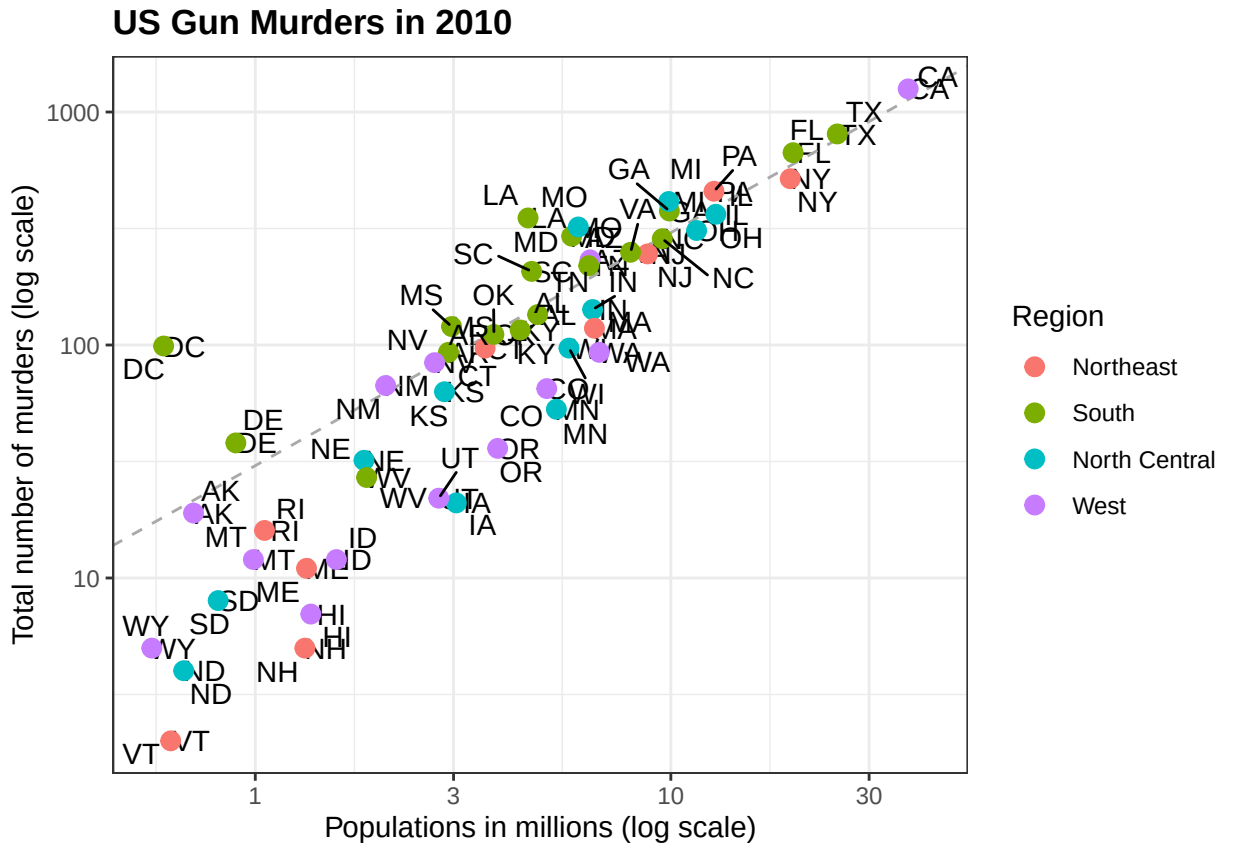
```
library("ggthemes")
p + theme_economist()
```



You can see how some of the other themes look by simply changing the function. For instance, you might try the theme_fivethirtyeight() theme instead.

The final difference has to do with the position of the labels. In our plot, some of the labels fall on top of each other. The add-on package ggrepel includes a geometry that adds labels while ensuring that they don't fall on top of each other. We simply change geom_text with geom_text_repel.

```
p + geom_text_repel()
```



7.12 Putting it all together

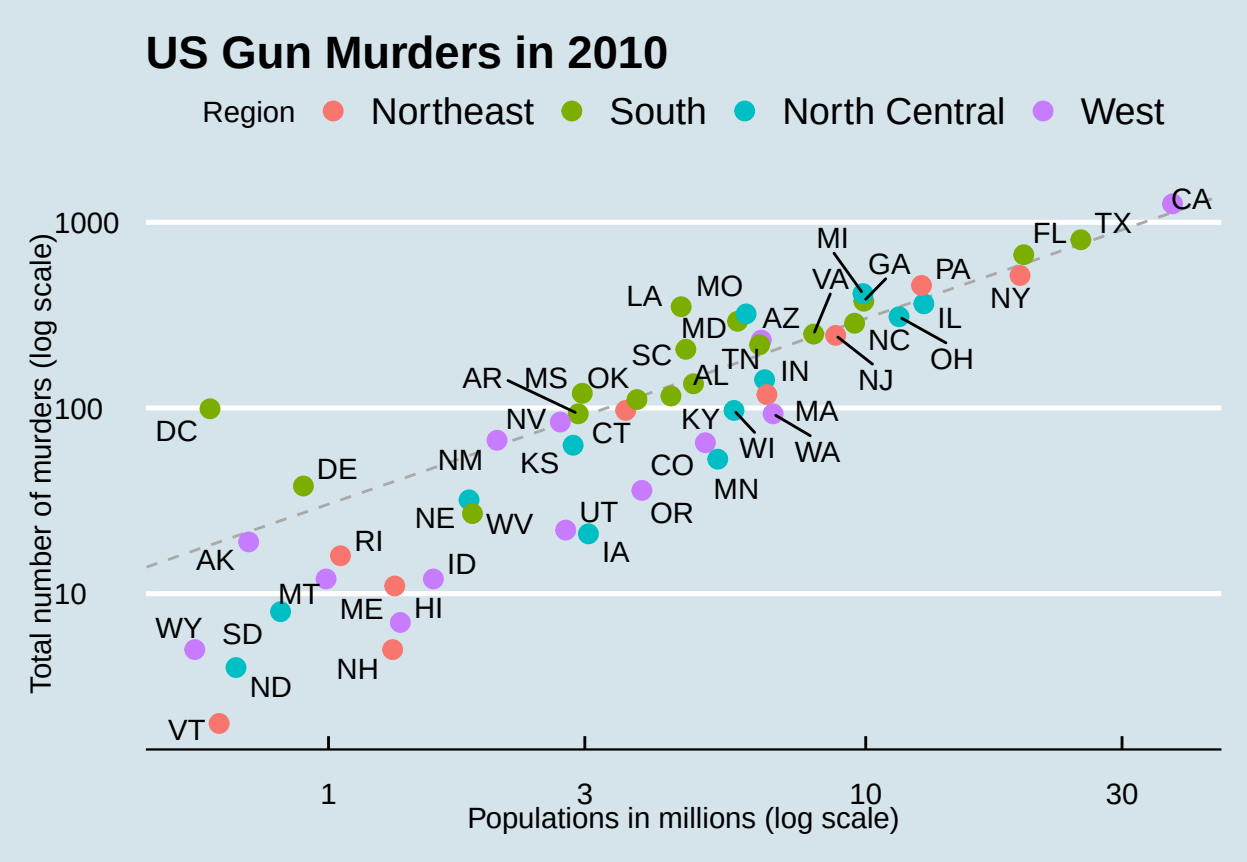
Now that we are done testing, we can write one piece of code that produces our desired plot from scratch.

```
library(ggthemes)
library(ggrepel)

r <- murders %>%
  summarize(rate = sum(total) / sum(population) * 10^6) %>%
  pull(rate)

murders %>% ggplot(aes(population/10^6, total, label = abb)) +
  geom_abline(intercept = log10(r), lty = 2, color = "darkgrey") +
  geom_point(aes(col=region), size = 3) +
  geom_text_repel() +
  scale_x_log10() +
  scale_y_log10() +
  xlab("Populations in millions (log scale)") +
```

```
  ylab("Total number of murders (log scale)") +
  ggtitle("US Gun Murders in 2010") +
  scale_color_discrete(name = "Region") +
  theme_economist()
```



7.13 Quick plots with qplot

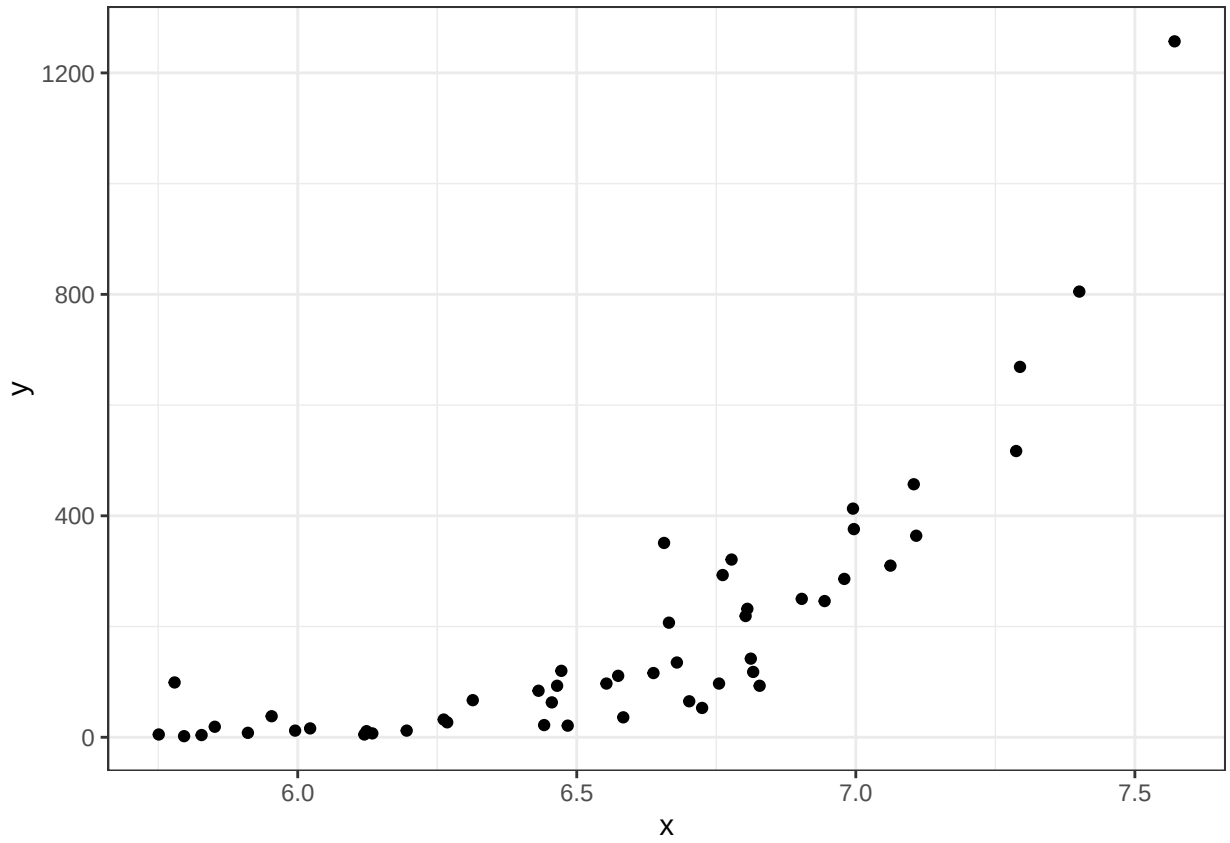
We have learned the powerful approach to generating visualization with ggplot. However, there are instances in which all we want is to make a quick plot of, for example, a histogram of the values in a vector, a scatterplot of the values in two vectors, or a boxplot using categorical and numeric vectors. We demonstrated how to generate these plots with hist, plot, and boxplot. However, if we want to keep consistent with the ggplot style, we can use the function qplot.

If we have values in two vectors, say:

```
data("murders")
x <- log10(murders$population)
y <- murders$total
```

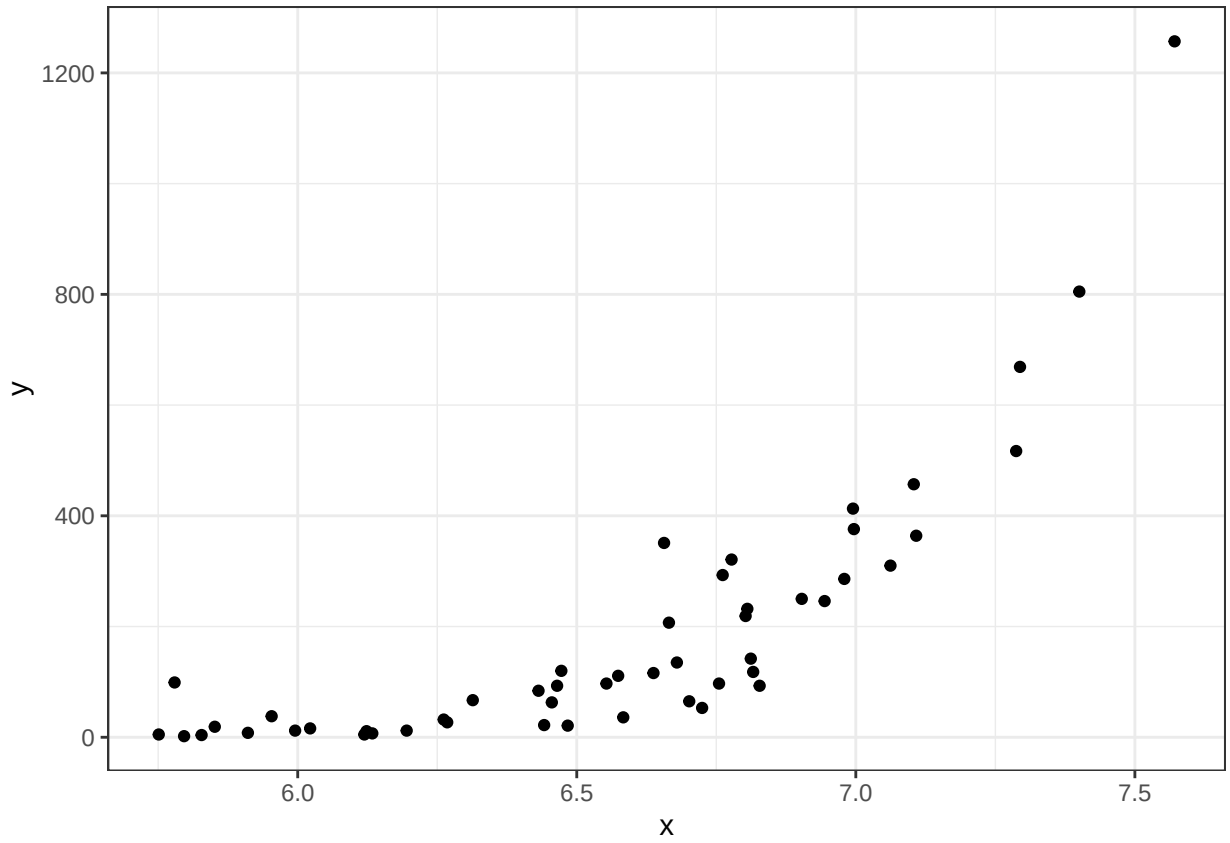
and we want to make a scatterplot with ggplot, we would have to type something like:

```
data.frame(x = x, y = y) %>%
  ggplot(aes(x, y)) +
  geom_point()
```



This seems like too much code for such a simple plot. The qplot function sacrifices the flexibility provided by the ggplot approach, but allows us to generate a plot quickly.

```
qplot(x, y)
```



We will learn more about qplot in Section 8.16

0.2 7.14 Grids of plots

There are often reasons to graph plots next to each other. The gridExtra package permits us to do that:

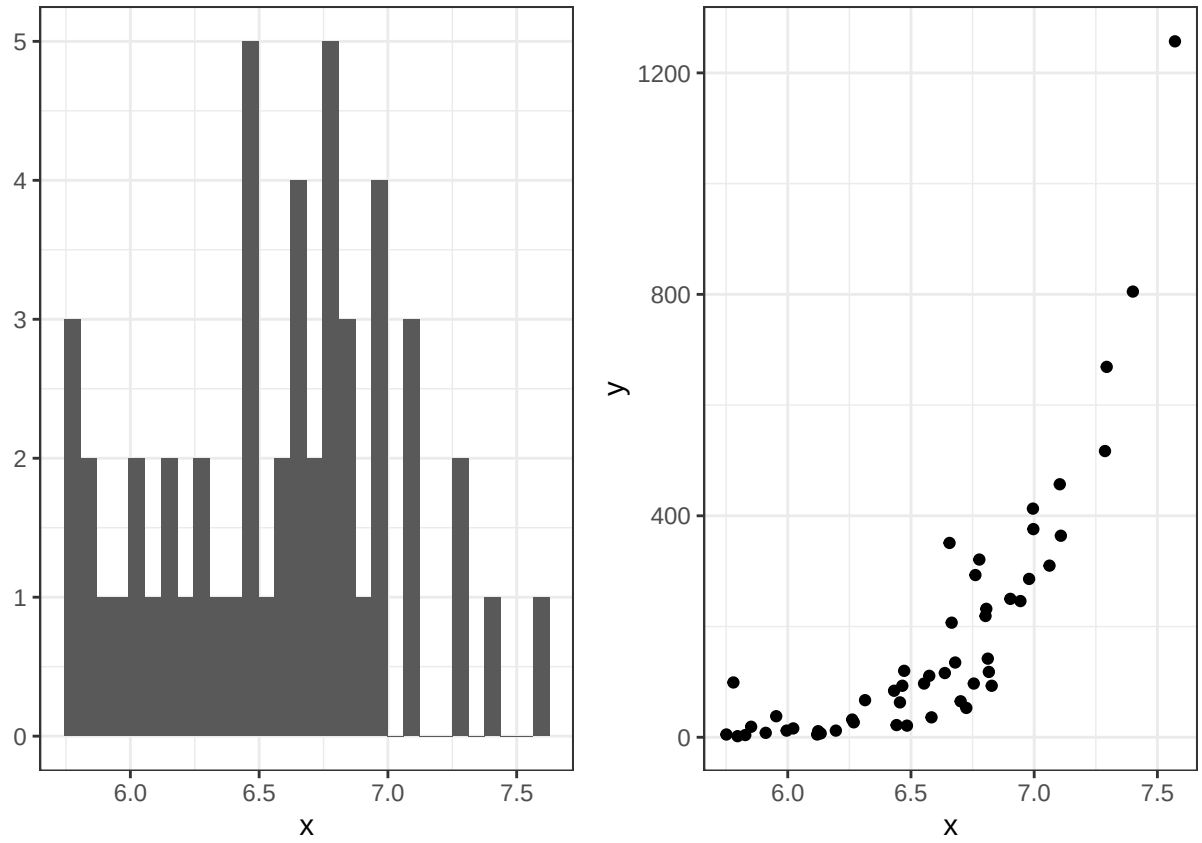
```
library(gridExtra)
```

```
##
##         : 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
p1 <- qplot(x)
p2 <- qplot(x, y)
grid.arrange(p1, p2, ncol = 2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



7.15 Exercises

Start by loading the dplyr and ggplot2 library as well as the murders and heights data.

```
library(dplyr)
library(ggplot2)
library(dslabs)
data("heights")
data("murders")
```

1. With ggplot2 plots can be saved as objects. For example we can associate a dataset with a plot object like this

```
p <- ggplot(data = murders)
```

Because data is the first argument we don't need to spell it out

```
p <- ggplot(murders)
```

and we can also use the pipe:

```
p <- murders %>% ggplot()
```

What is class of the object p?

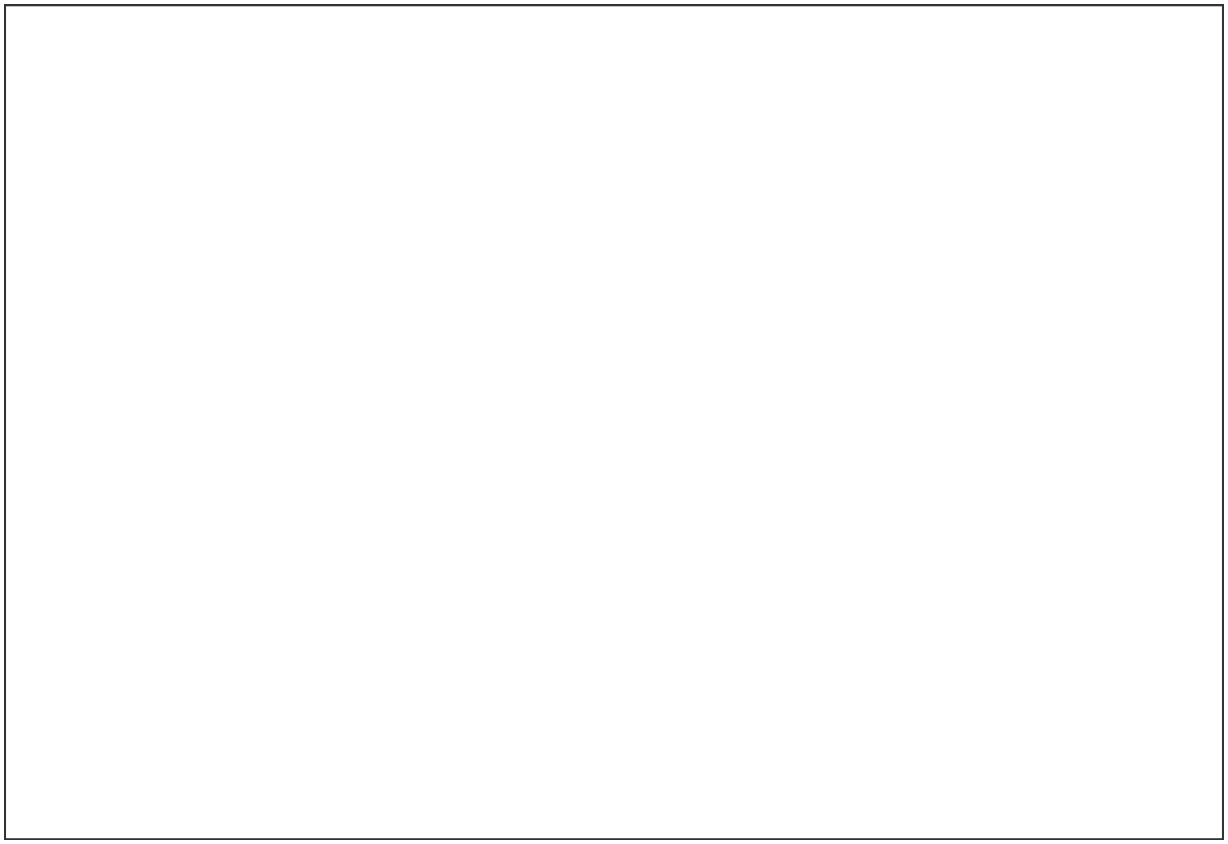
```
class(p)
```

```
## [1] "gg"     "ggplot"
```

2. Remember that to print an object you can use the command print or simply type the object. Print the object p defined in exercise one and describe what you see.

a. Nothing happens.
b. **A blank slate plot.**
c. A scatterplot
d. A histogram

```
print(p)
```



3. Using the pipe %>%, create an object p but this time associated with the heights dataset instead of the murders dataset.

```
p <- heights %>% ggplot()
```

4. What is the class of the object p you have just created?

```
class(p)
```

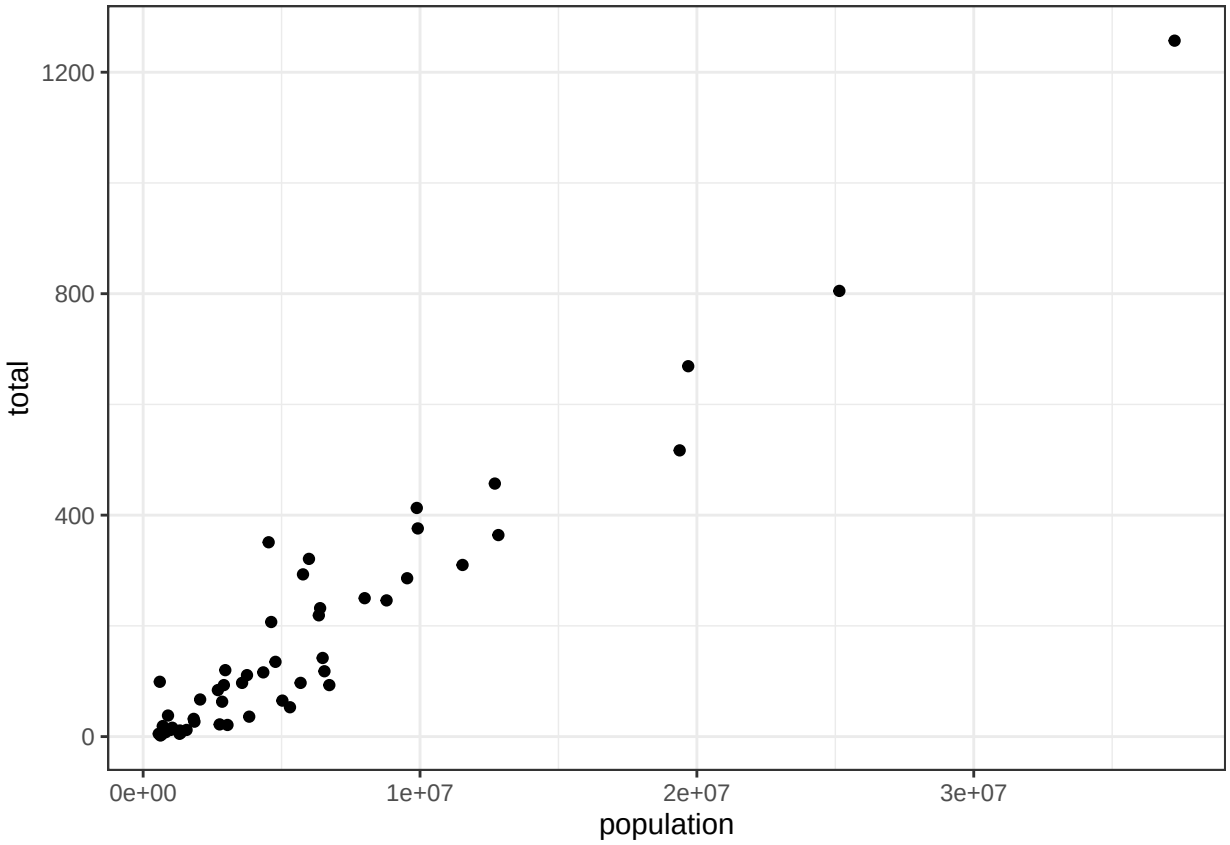
```
## [1] "gg"     "ggplot"
```

5. Now we are going to add a layer and the corresponding aesthetic mappings. For the murders data we plotted total murders versus population sizes. Explore the murders data frame to remind yourself what are the names for these two variables and select the correct answer. Hint: Look at ?murders.

a. state and abb.
b. total_murders and population_size.
c. **total and population.**
d. murders and size.

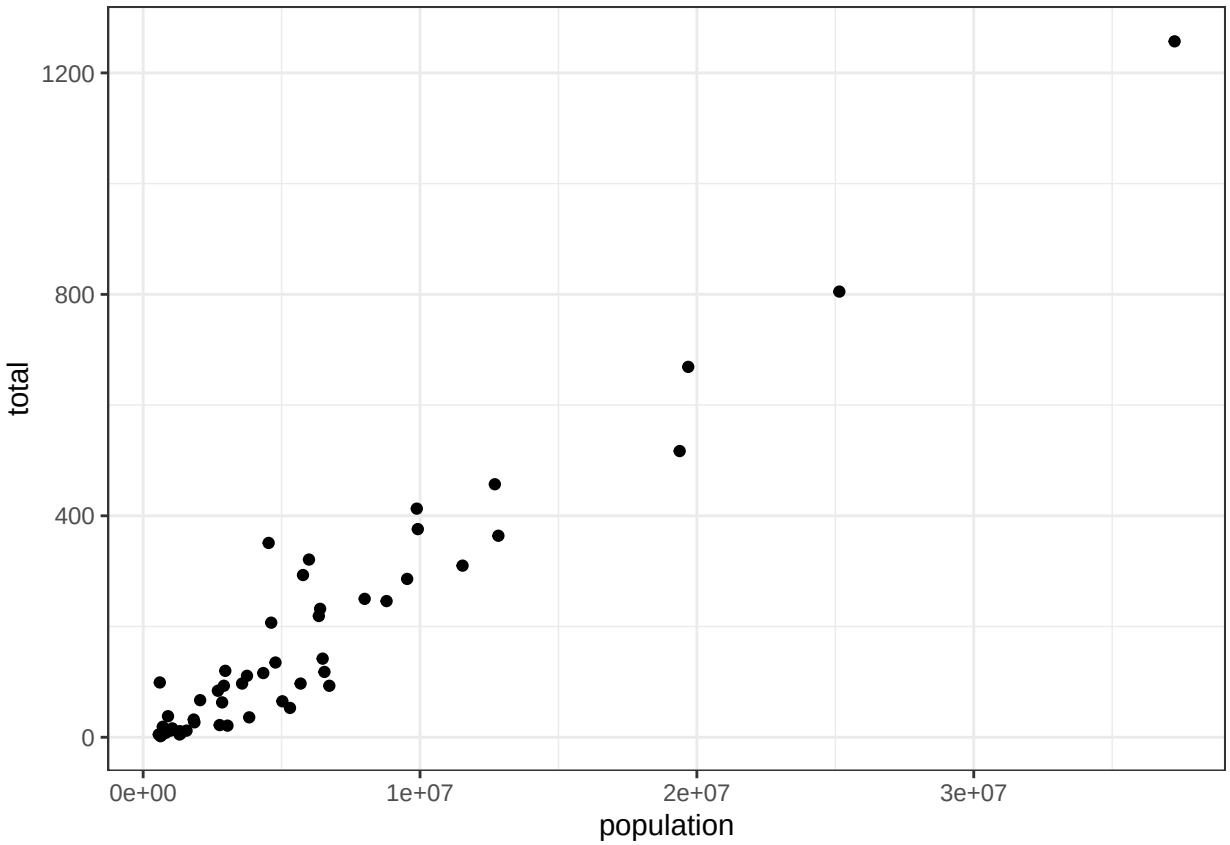
6. To create the scatterplot we add a layer with geom_point. The aesthetic mappings require us to define the x-axis and y-axis variables, respectively. So the code looks like this:

```
murders %>% ggplot(aes(x = population, y = total)) +
  geom_point()
```



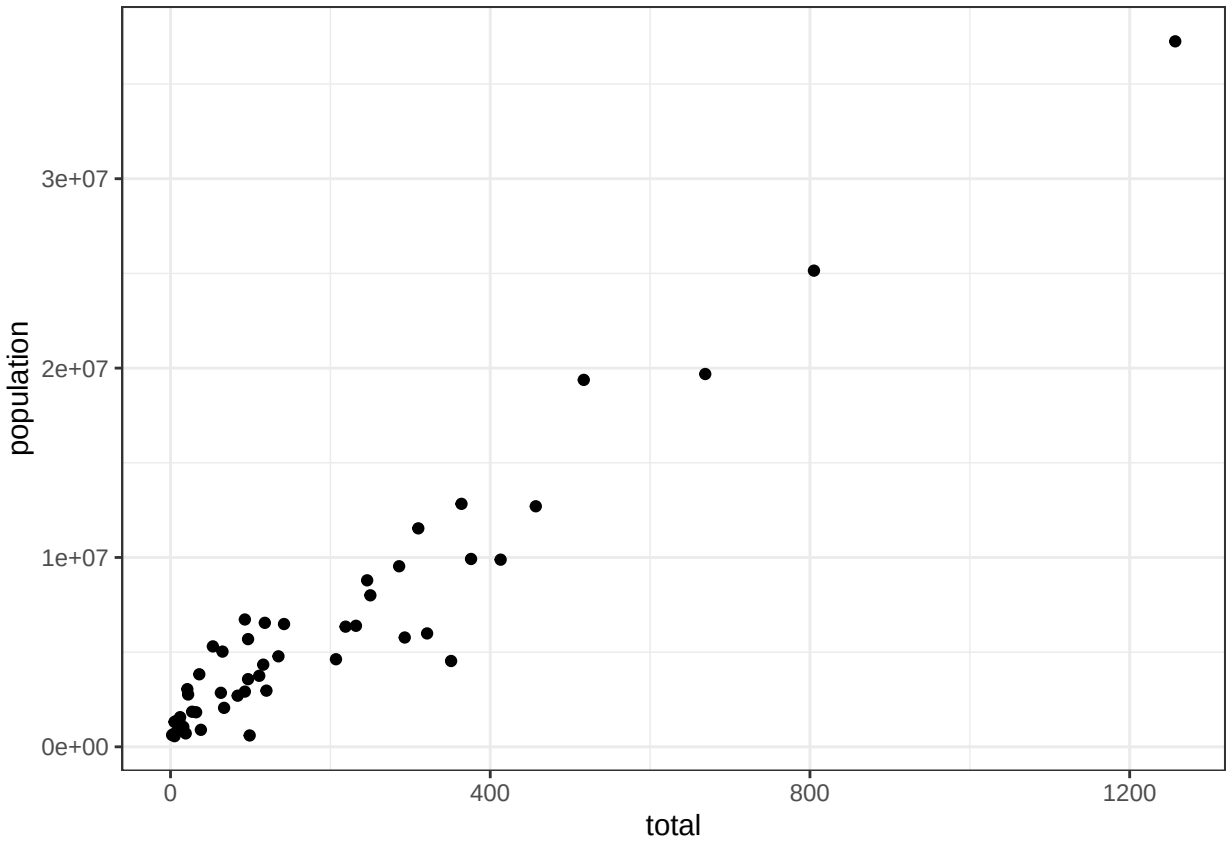
7. Note that if we don't use argument names, we can obtain the same plot by making sure we enter the variable names in the right order like this:

```
murders %>% ggplot(aes(population, total)) +
  geom_point()
```



Remake the plot but now with total in the x-axis and population in the y-axis.

```
murders %>% ggplot(aes(total, population)) +
  geom_point()
```



8. If instead of points we want to add text, we can use the geom_text() or geom_label() geometries. The following code

```
# murders %>% ggplot(aes(population, total)) +
#   geom_label()
```

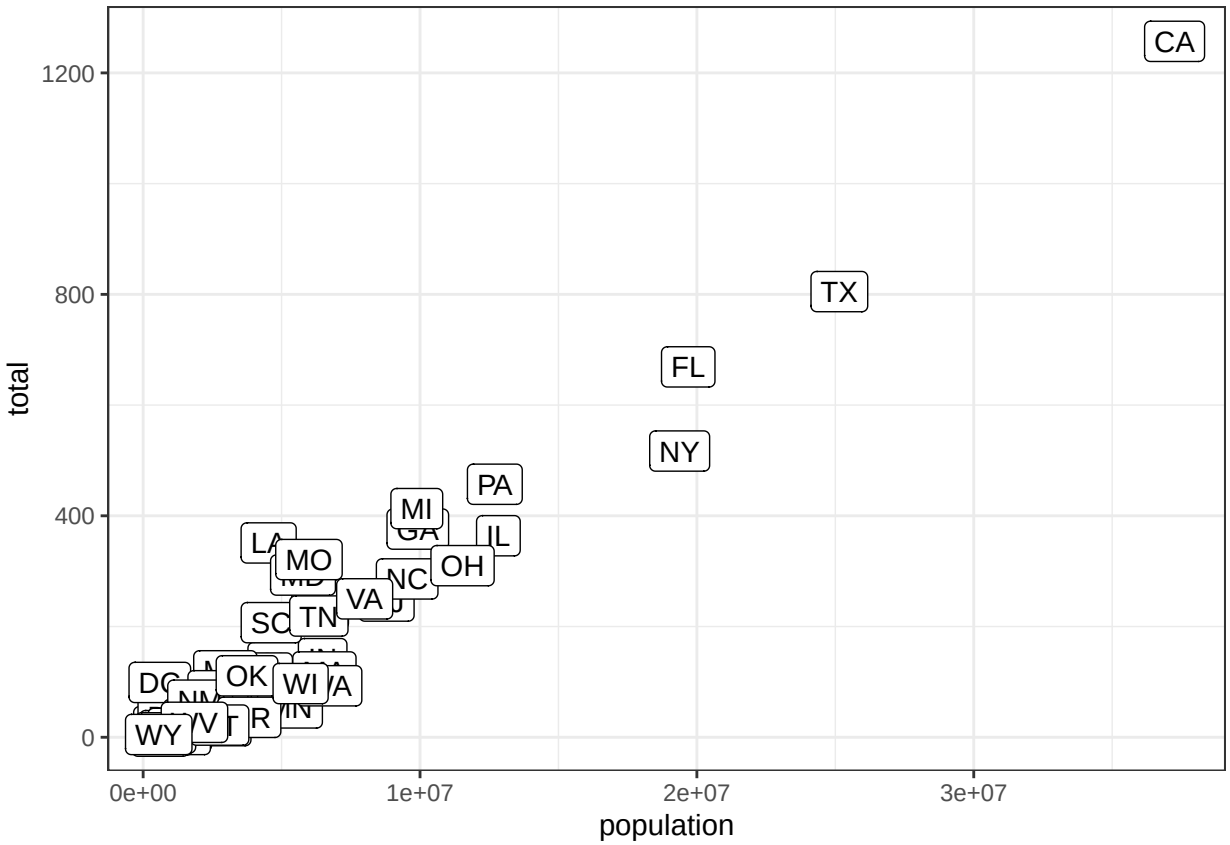
will give us the error message: Error: geom_label requires the following missing aesthetics: label

Why is this?

a. **We need to map a character to each point through the label argument in aes.**
b. We need to let geom_label know what character to use in the plot.
c. The geom_label geometry does not require x-axis and y-axis values.
d. geom_label is not a ggplot2 command.

9. Rewrite the code above to use abbreviation as the label through aes

```
murders %>% ggplot(aes(population, total, label = abb)) +
  geom_label()
```

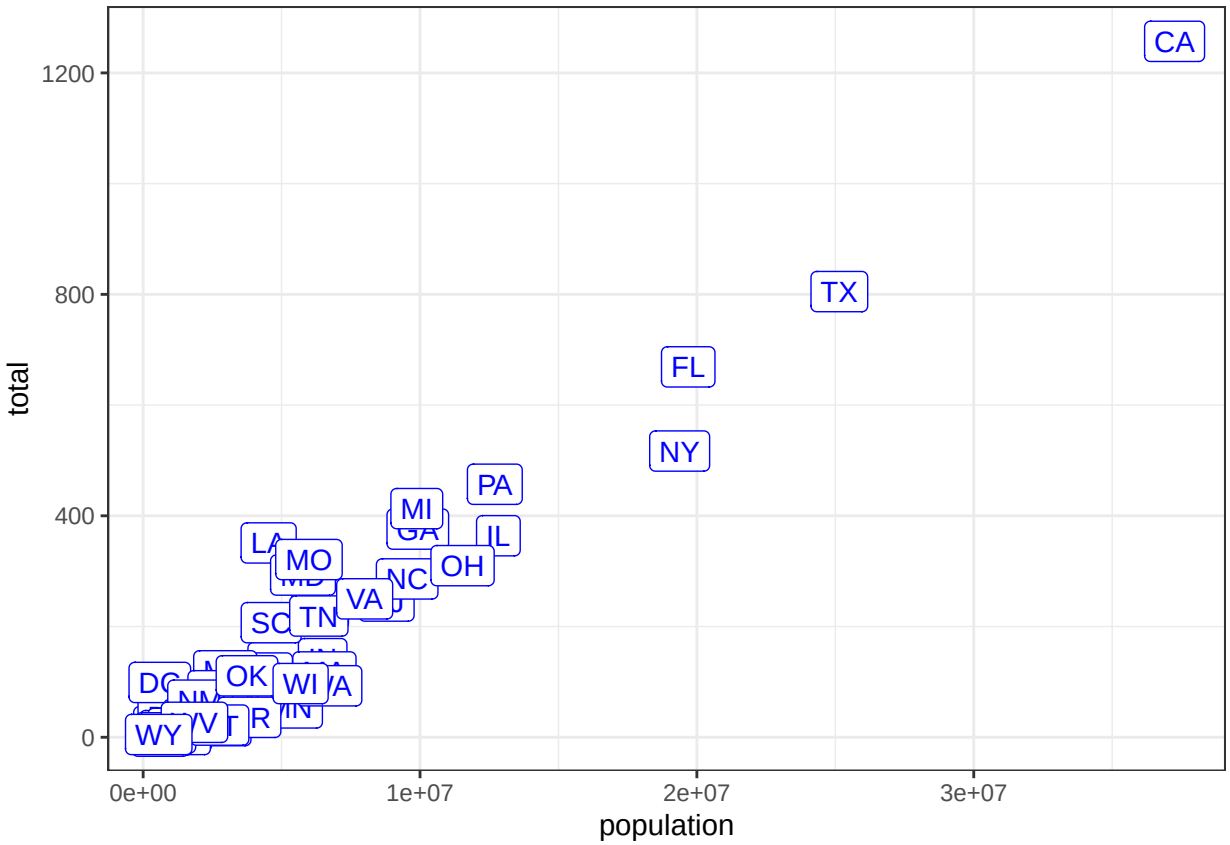


10. Change the color of the labels to blue. How will we do this?

    a. Adding a column called blue to murders.
    b. Because each label needs a different color we map the colors through aes.
    c. Use the color argument in ggplot.
    d. **Because we want all colors to be blue, we do not need to map colors, just use the color argument in geom_label.**

11. Rewrite the code above to make the labels blue.

```
murders %>% ggplot(aes(population, total, label = abb)) +
  geom_label(color = "blue")
```

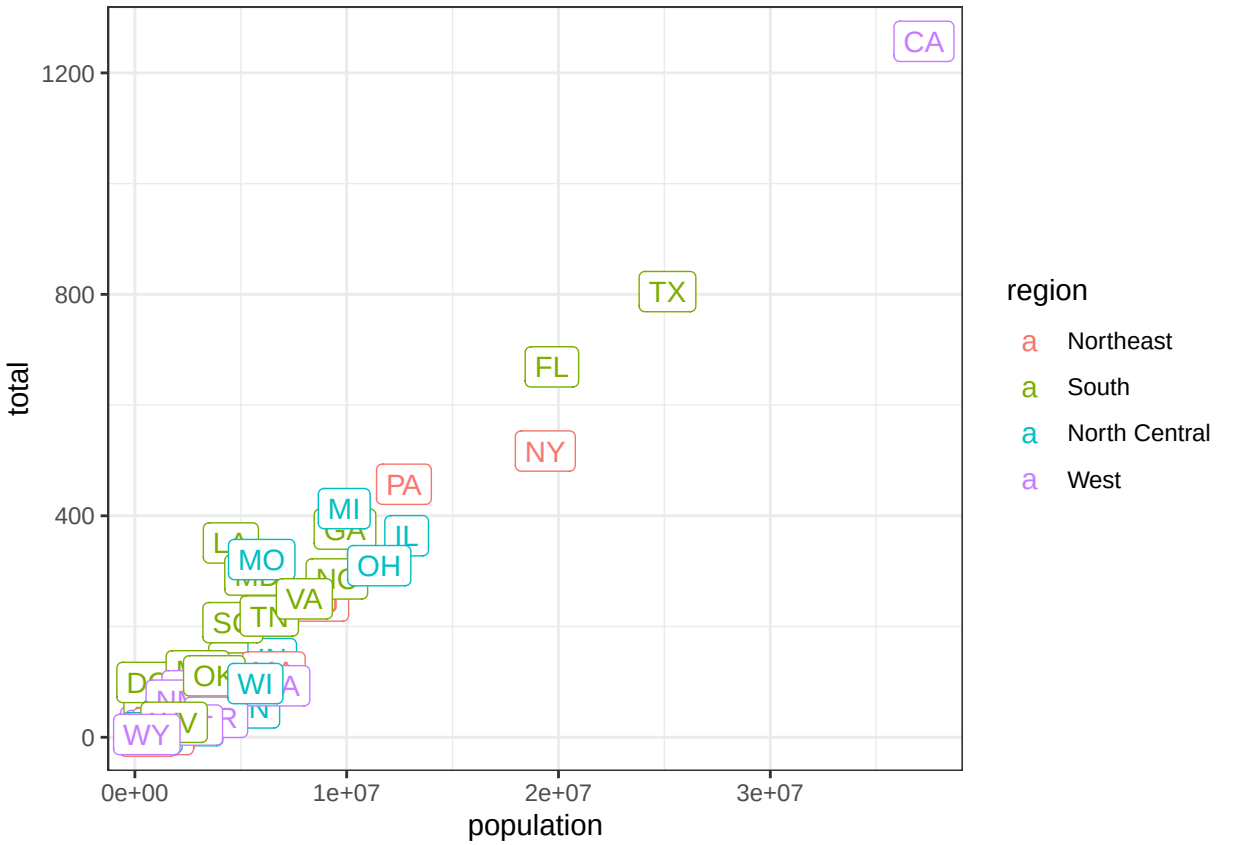


12. Now suppose we want to use color to represent the different regions. In this case which of the following is most appropriate:

a. Adding a column called color to murders with the color we want to use.
b. **Because each label needs a different color we map the colors through the color argument of aes.**
c. Use the color argument in ggplot.
d. Because we want all colors to be blue, we do not need to map colors, just use the color argument in geom_label.

13. Rewrite the code above to make the labels' color be determined by the state's region.

```
murders %>% ggplot(aes(population, total, label = abb, color = region)) +
  geom_label()
```

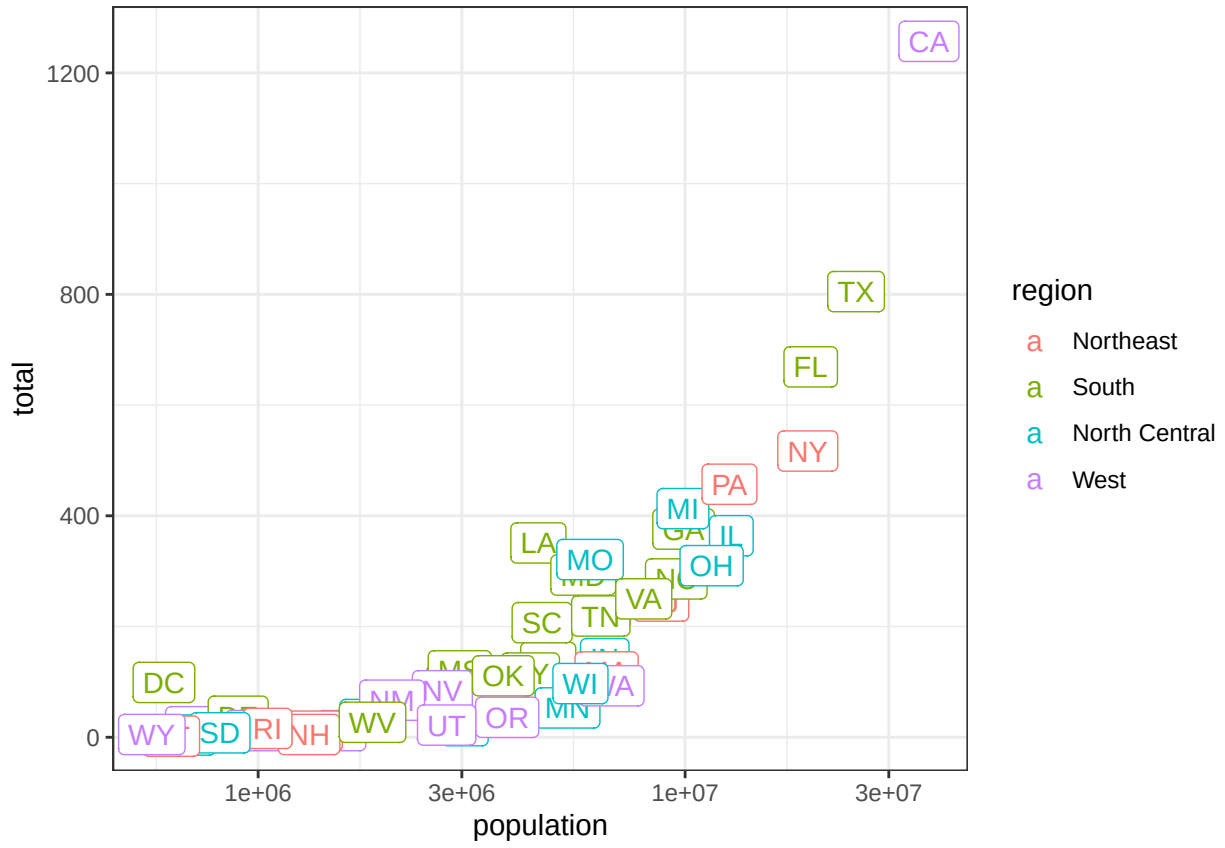


14. Now we are going to change the x-axis to a log scale to account for the fact the distribution of population is skewed. Let's start by defining an object p holding the plot we have made up to now

```
p <- murders %>%
  ggplot(aes(population, total, label = abb, color = region)) +
  geom_label()
```

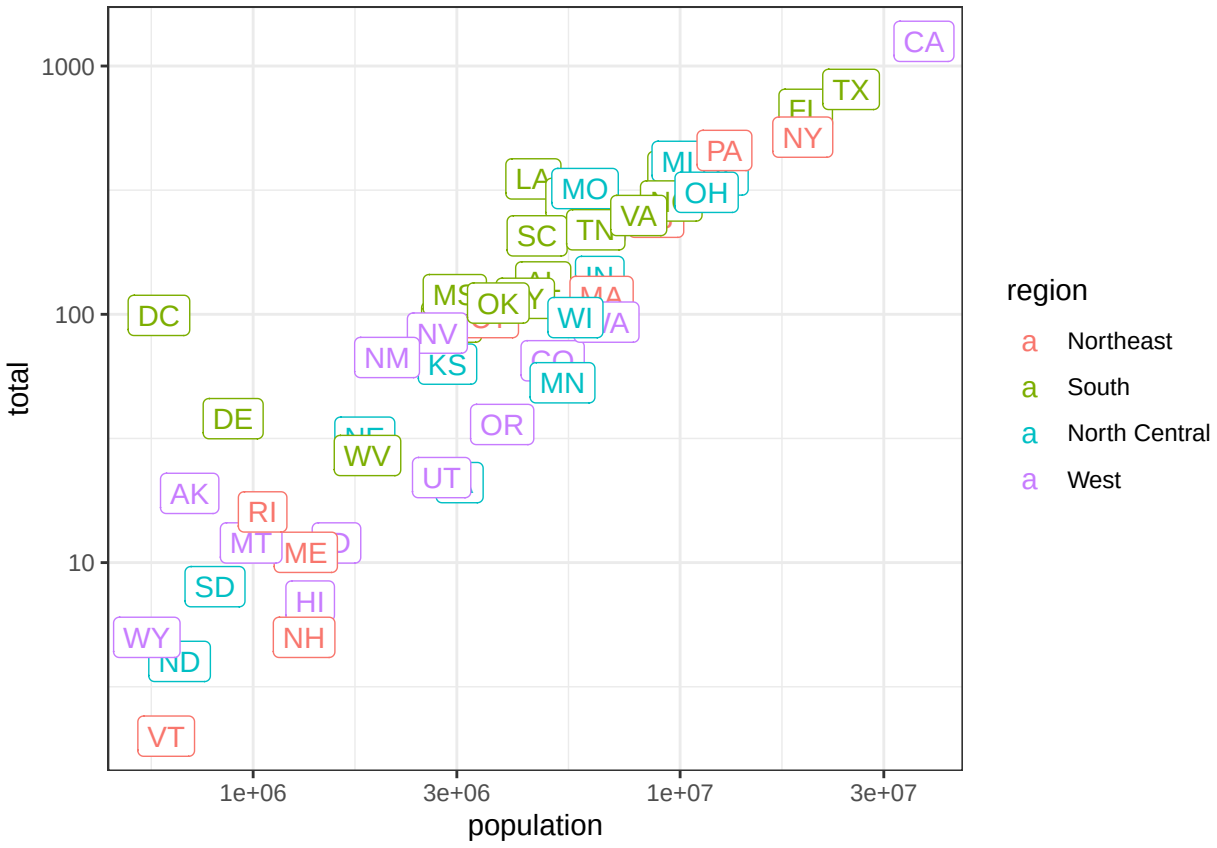
To change the x-axis to a log scale we learned about the scale_x_log10() function. Add this layer to the object p to change the scale and render the plot.

```
p <- murders %>%
  ggplot(aes(population, total, label = abb, color = region)) +
  geom_label() +
  scale_x_log10()
p
```



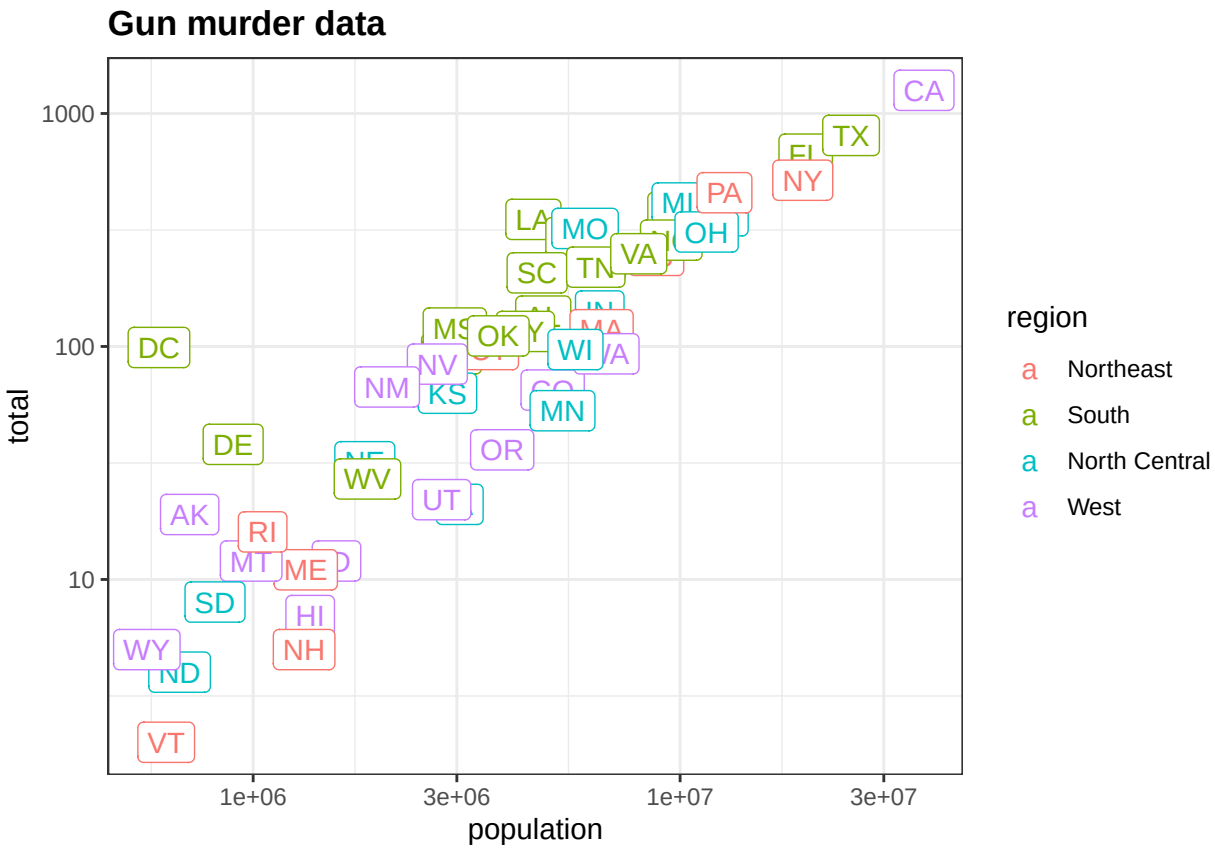
15. Repeat the previous exercise but now change both axes to be in the log scale.

```
p <- murders %>%
  ggplot(aes(population, total, label = abb, color = region)) +
  geom_label() +
  scale_x_log10() +
  scale_y_log10()
p
```



16. Now edit the code above to add the title "Gun murder data" to the plot. Hint: use the ggtitle function.

```
p <- murders %>%
  ggplot(aes(population, total, label = abb, color = region)) +
  geom_label() +
  scale_x_log10() +
  scale_y_log10() +
  ggtitle("Gun murder data")
p
```


Gun murder data
1000
100
10
total
1e+06
3e+06
1e+07
3e+07
population
CA
TX
NY
PA
OH
MO
VA
SC
OK
WI
MN
NM
KS
NV
DC
DE
WV
OR
UT
AK
RI
MT
ME
SD
HI
NH
WY
VT
region
a Northeast
a South
a North Central
a West