R4DS

Cohort 4

Wed 6:00 – 7:00 US Central

Twitter: @Rspjut

5-MINUTE ICE BREAKER

What sport do you least understand?

AGENDA

- 5-Minute Ice breaker
- Quick Housekeeping Reminders
- Review Last Week's Topics in Chapter 7
- Finish Chapter 7
- Getting Help
- Next Week

QUICK HOUSEKEEPING REMINDERS

- Video camera is optional, but encouraged.
- I purposely err on the side of going fast. Slowing me down does not hurt my feelings.
- Take time to learn the theory (Grammar of Graphics, Tidy Data whitepaper, Relational Database theory, etc.).
- Please do the chapter exercises. Second-best learning opportunity!
- Please plan on teaching one of the lessons. Best learning opportunity!

DATASET USED IN CHAPTER 7: DIAMONDS

Diamonds (load `tidyverse` then `?diamonds`)

Variable	Format
price	Price in US dollars
carat	Weight of the diamond (0.2 – 5.01)
cut	Quality of the cut (Fair, Good, Very Good, Premium, Ideal)
color	Diamond color from D (best) to J (worst)
clarity	How clear. Worst = I1, SI2, SI1, VS2, VS1, VVS2, VVS1, IF
x	Length in mm
y	Width in mm
z	Depth in mm
depth	Depth percentage
table	Width of top of diamond relative to widest point

`head(diamonds)`

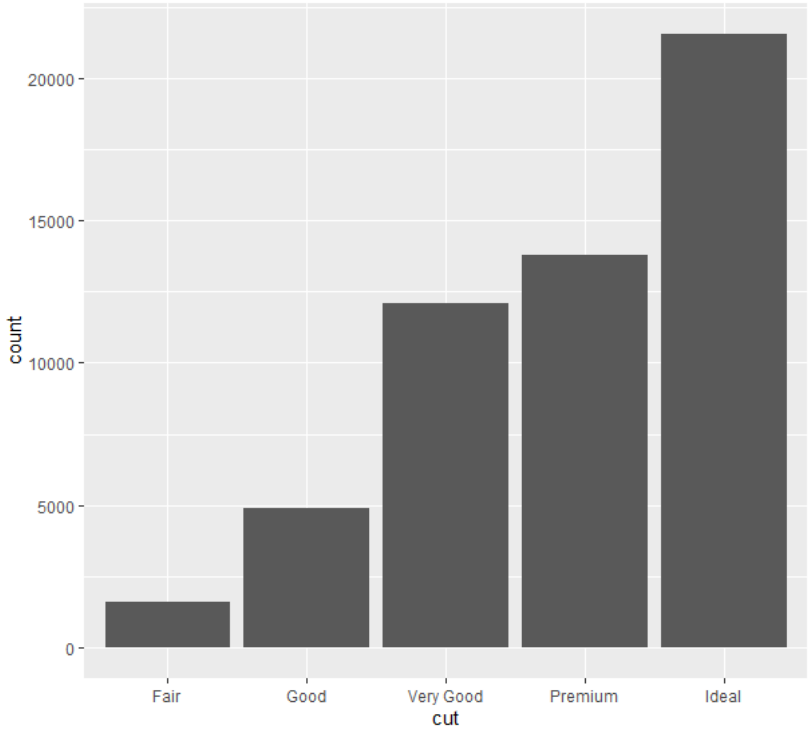
```
> head(diamonds)
# A tibble: 6 x 10
  carat cut       color clarity depth table price     x     y     z
  <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
1 0.23  Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
2 0.21  Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
3 0.23  Good      E     VS1      56.9    65   327  4.05  4.07  2.31
4 0.290 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
5 0.31  Good      J     SI2      63.3    58   335  4.34  4.35  2.75
6 0.24  Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
```

Total Records = 53,940

7.3.1 VISUALIZING DISTRIBUTIONS

Using `diamonds`, let's visualize the number of diamonds that belong to each value of the `cut` variable.

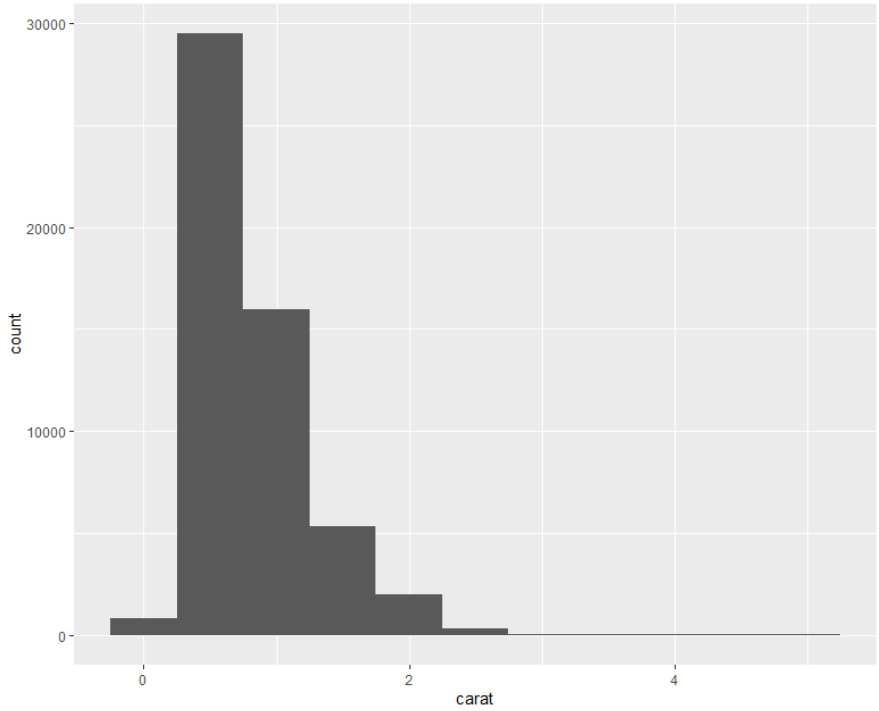
```
ggplot(data = diamonds) + geom_bar(mapping = aes(x = cut))
```



7.3.1 VISUALIZING DISTRIBUTIONS

Using `diamonds`, let's visualize the number of diamonds that belong to each value of the `carat` variable.

```
ggplot(data = diamonds) +
    geom_histogram(mapping = aes(x = carat), binwidth = 0.5)
```

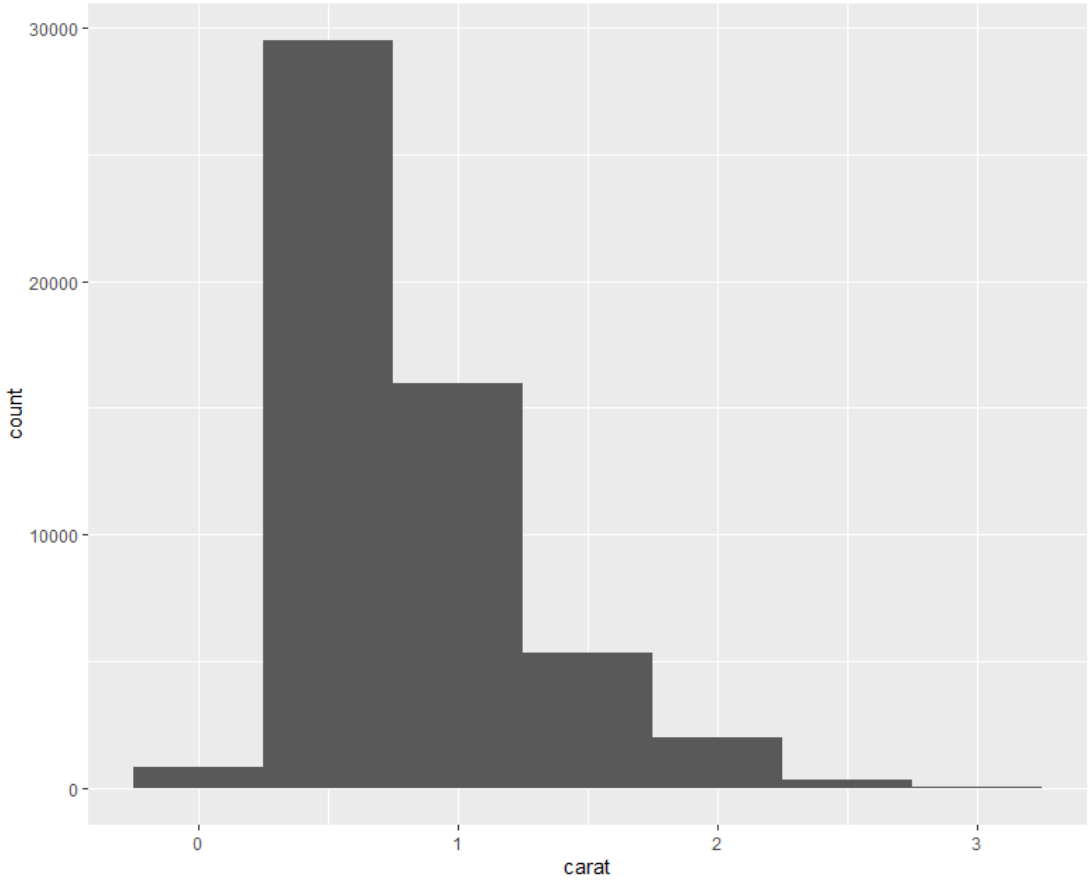


7.3.1 VISUALIZING DISTRIBUTIONS

Let's filter our dataset to only include diamonds under 3.0 carats.

```
diamonds %>%
filter(carat < 3) %>%
ggplot() + geom_histogram(mapping = aes(x = carat), binwidth = 0.5)
```

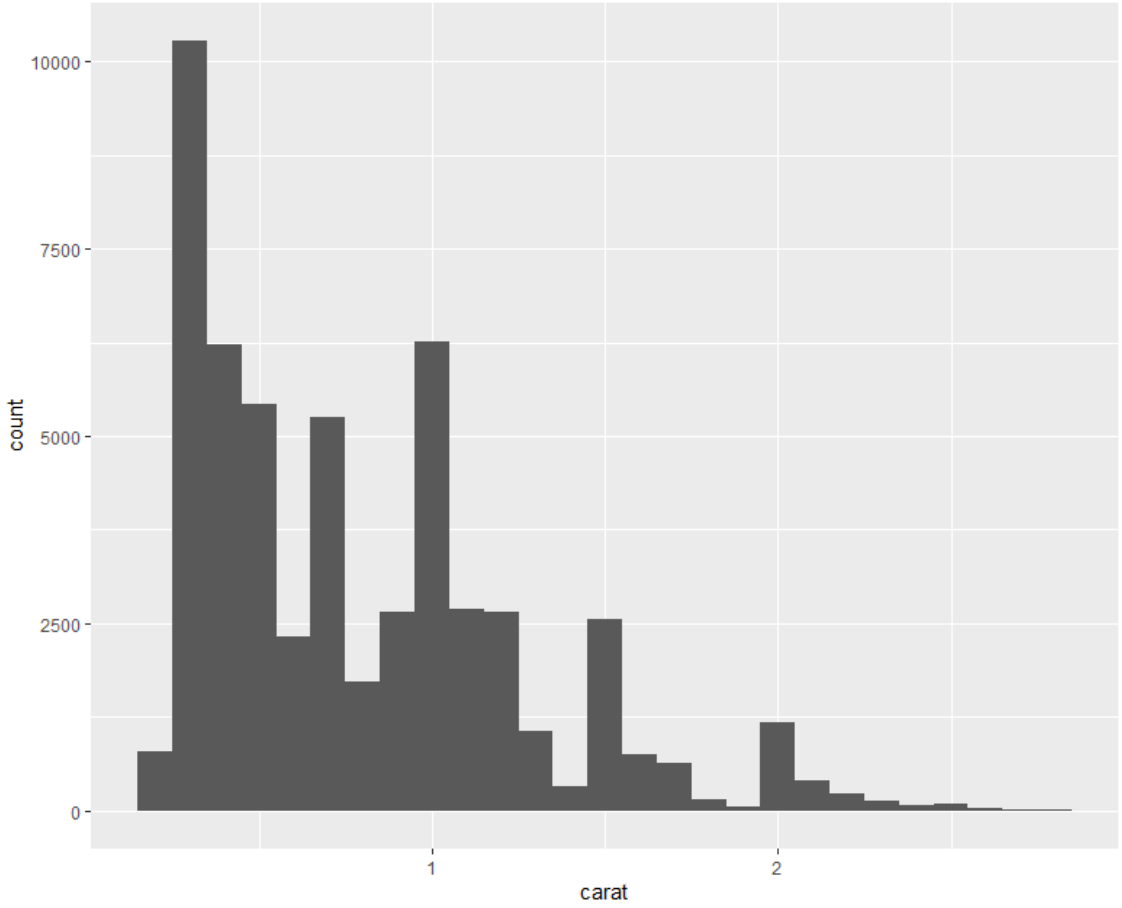
```
diamonds %>%

filter(carat < 3) %>%

ggplot() + geom_histogram(mapping = aes(x = carat), binwidth = 0.5)
```



7.3.1 VISUALIZING DISTRIBUTIONS

What if you reduce the `binwidth` from `0.5` to `0.1`?

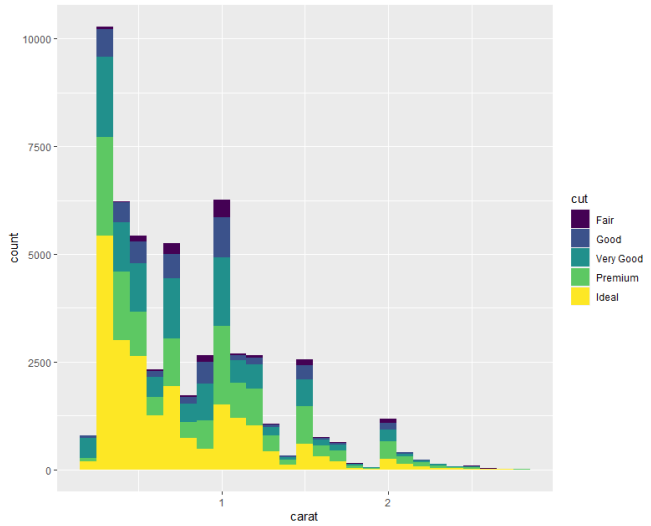
```
diamonds %>%
filter(carat < 3) %>%
ggplot() + geom_histogram(mapping = aes(x = carat), binwidth = 0.1)
```



7.3.1 VISUALIZING DISTRIBUTIONS

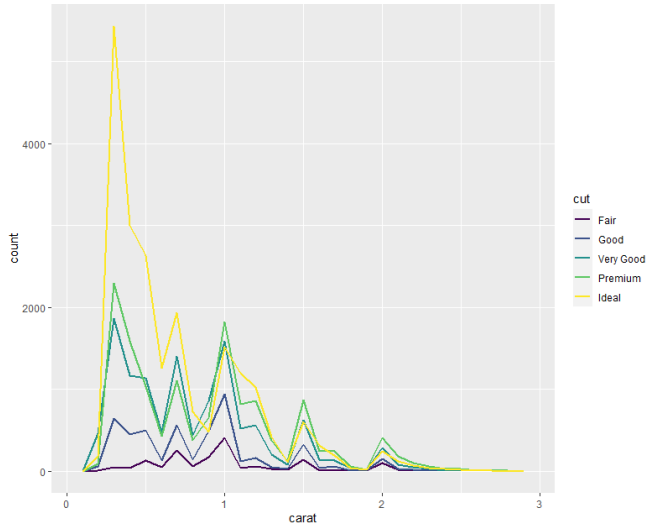
What if we want to fill the histogram bars with color based on `cut`?

```
diamonds %>%
filter(carat < 3) %>%
ggplot() + geom_histogram(mapping = aes(x = carat, fill = cut), binwidth = 0.1)
```



The `freqpoly` visualization is better.
The aesthetic name is `color` instead of `fill`.

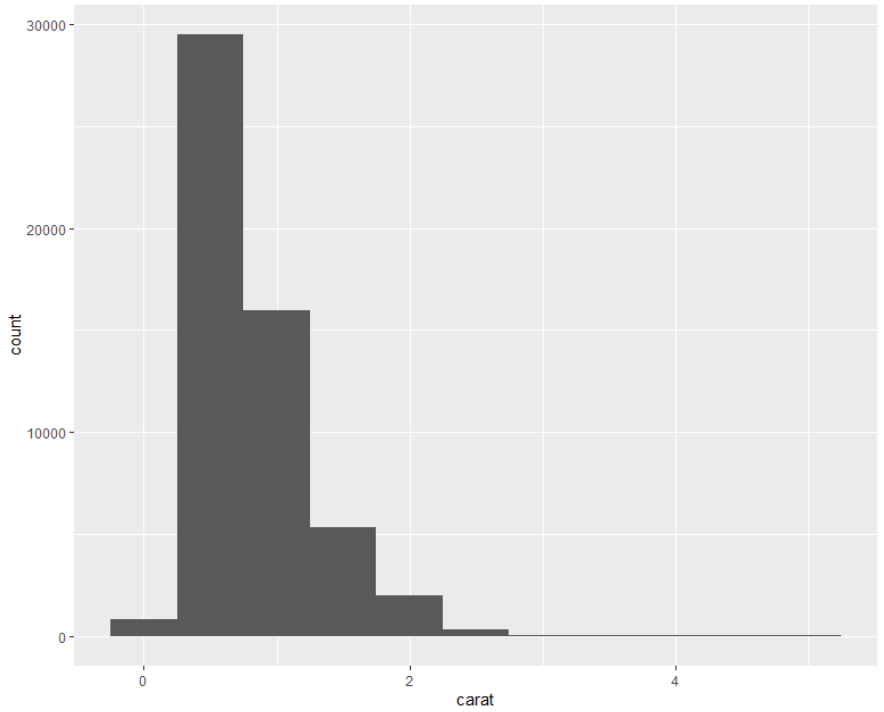
```
diamonds %>%
filter(carat < 3) %>%
ggplot() + geom_freqpoly(mapping = aes(x = carat, color = cut), binwidth = 0.1)
```



7.3.1 VISUALIZING DISTRIBUTIONS

Let's zoom in on the long tail of `diamonds`.
First, using `diamonds`, remove the filter so that we see all values of carat. Use a `binwidth` of 0.5.

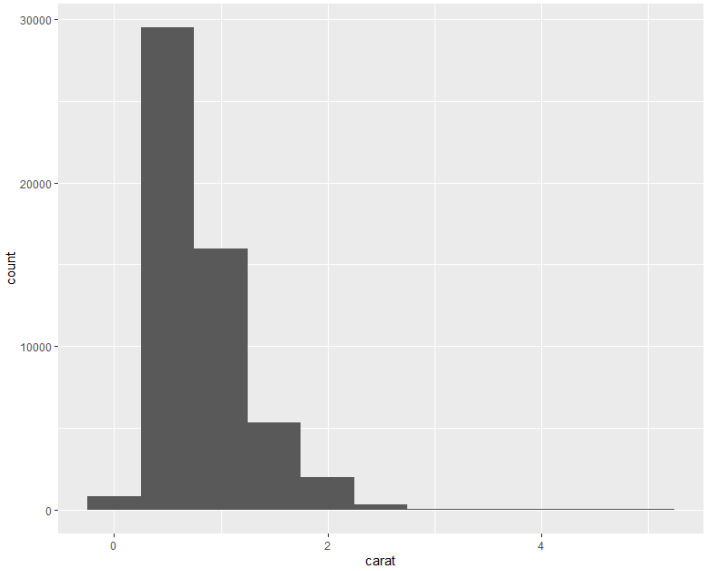
```
ggplot(data = diamonds) +
    geom_histogram(mapping = aes(x = carat), binwidth = 0.5)
```



7.3.1 VISUALIZING DISTRIBUTIONS

Notice the first bar spans to the negatives. This will affect our interpretation of how many diamonds are in each bin.

```
ggplot(data = diamonds) +
    geom_histogram(mapping = aes(x = carat),
    binwidth = 0.5)
```



The reason the bar spans to the negatives is because the lowest carat value (which is 0.2) is in the center of the first 0.5 span.

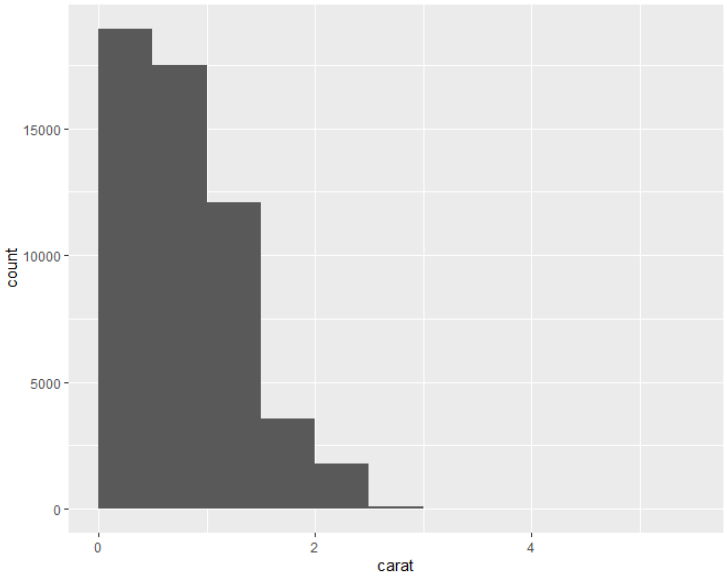
```
diamonds %>%
    count(cut_width(carat, width = 0.5))
```

```
# A tibble: 11 x 2
   `cut_width(carat, 0.5)`     n
   <fct>                   <int>
 1 [-0.25,0.25]              785
 2 (0.25,0.75]             29498
 3 (0.75,1.25]             15977
 4 (1.25,1.75]              5313
 5 (1.75,2.25]              2002
 6 (2.25,2.75]               322
 7 (2.75,3.25]                32
 8 (3.25,3.75]                 5
 9 (3.75,4.25]                 4
10 (4.25,4.75]                 1
11 (4.75,5.25]                 1
```

7.3.1 VISUALIZING DISTRIBUTIONS

We need to correct this before we can move on. Use the `boundary` argument to line up the bars better, starting at 0.

```
ggplot(data = diamonds) +
    geom_histogram(mapping = aes(x = carat),
    binwidth = 0.5, boundary = 0)
```



Now the first bin starts at 0 and goes to 0.5. Notice that the counts align.

```
diamonds %>%
    count(cut_width(carat, width = 0.5,
    boundary = 0))
```

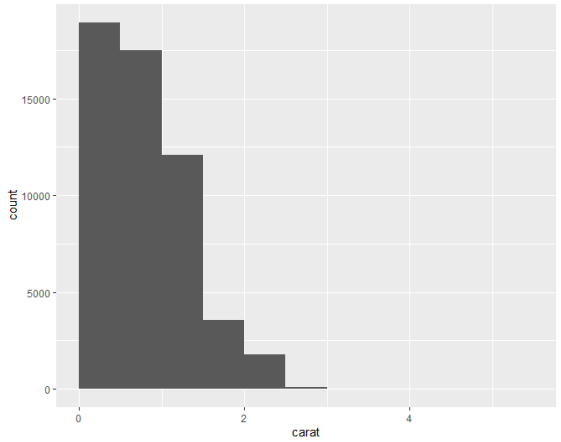
```
# A tibble: 10 x 2
   `cut_width(carat, 0.5, boundary = 0)`     n
   <fct>                                 <int>
 1 [0,0.5]                               18932
 2 (0.5,1]                               17506
 3 (1,1.5]                               12060
 4 (1.5,2]                                3553
 5 (2,2.5]                                1763
 6 (2.5,3]                                  94
 7 (3,3.5]                                  23
 8 (3.5,4]                                   4
 9 (4,4.5]                                   4
10 (5,5.5]                                   1
```

7.3.1 VISUALIZING DISTRIBUTIONS

We see from the table, there are 32 diamonds larger than 3.0 carats. However they are invisible on the graph.

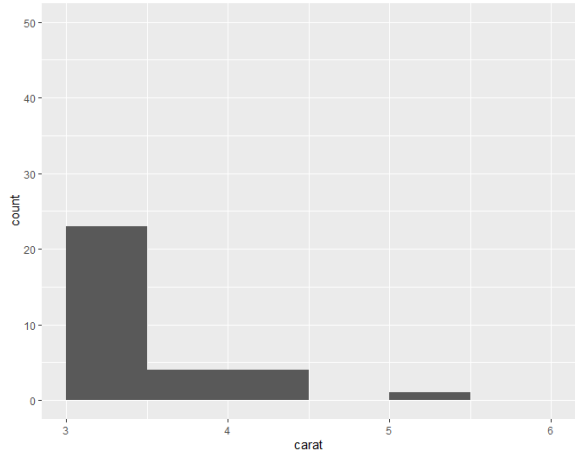
```
diamonds %>%
count(cut_width(carat,
    width = 0.5,
    boundary = 0))
```

```
# A tibble: 10 x 2
   `cut_width(carat, 0.5, boundary = 0)`     n
   <fct>                                 <int>
 1 [0,0.5]                               18932
 2 (0.5,1]                               17506
 3 (1,1.5]                               12060
 4 (1.5,2]                                3553
 5 (2,2.5]                                1763
 6 (2.5,3]                                  94
 7 (3,3.5]                                  23
 8 (3.5,4]                                   4
 9 (4,4.5]                                   4
10 (5,5.5]                                   1
```

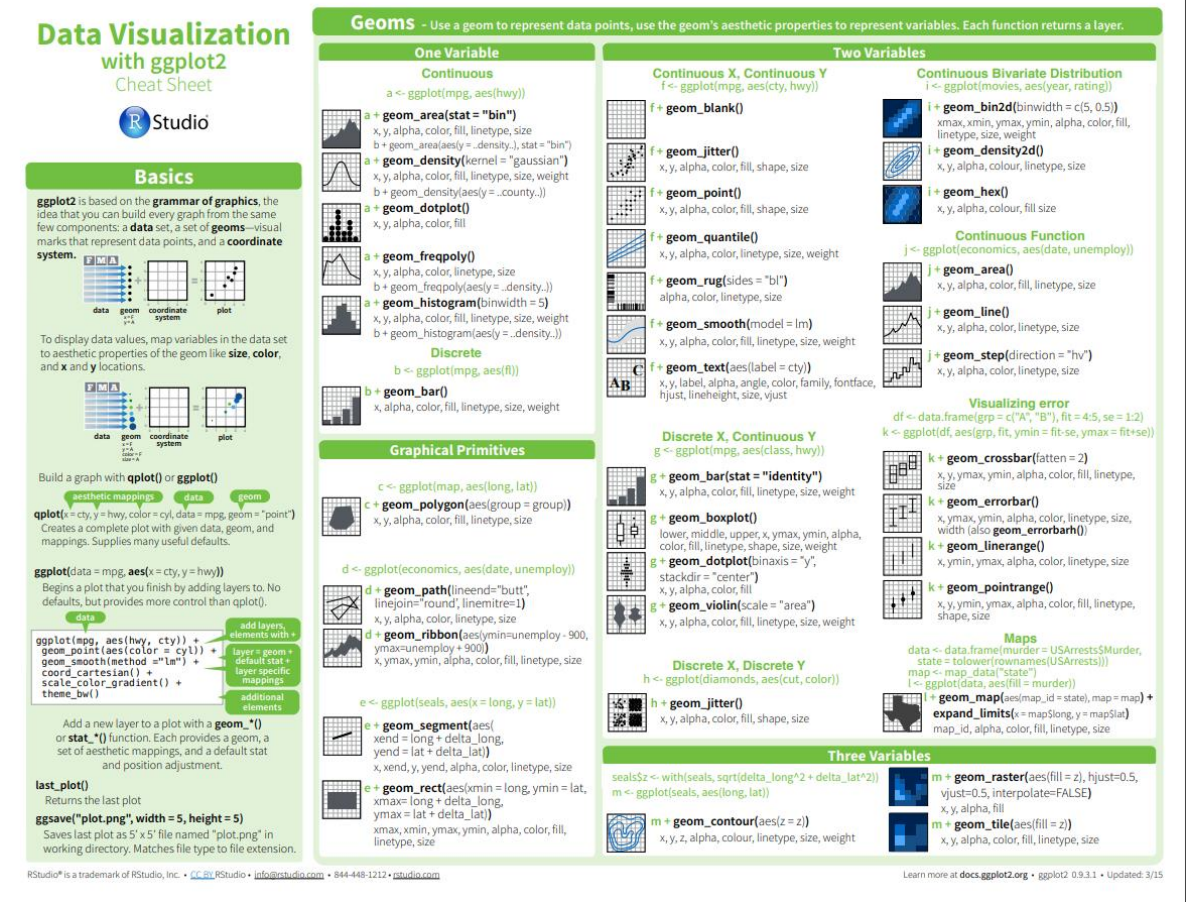


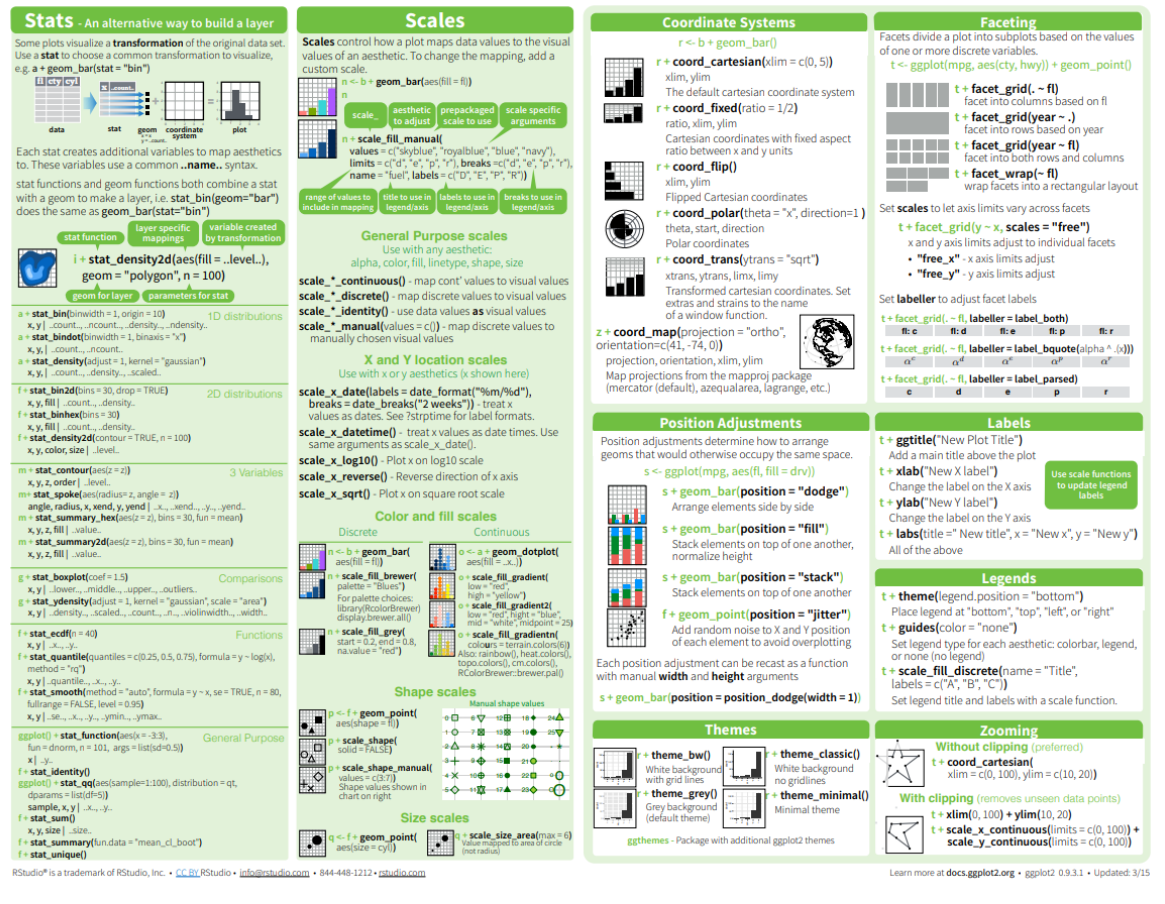
Our task is to zoom in on just these 32 diamonds greater than 3.0 carats. Use xlim and ylim to cut the axes to the intervals you specify.

```
diamonds %>%
filter(carat > 3) %>%
ggplot(data = diamonds) +
    geom_histogram(mapping = aes(x = carat),
    binwidth = 0.5, boundary = 0)  +
    xlim(3, 6) + ylim(0, 50)
```



7.3.1 GGPLOT CHEATSHEET





GETTING HELP

- Ask questions during our call
- Google
- Stack Overflow
- Slack
- Office Hours r4ds.io/calendar
- Twitter #rstats
- r4ds answer keys: Jeff Arnold (preferred) or Bryan Shalloway (also good)
- Cheatsheets

NEXT WEEK…

- Continue Chapter 7: Exploratory Data Analysis


RStudio