One-Variable Graphs

```
# As usual, we load the mosaic and tidyverse packages.
library(mosaic)
library(tidyverse)

# We make use of the run09 data frame, which is included in the
# cherryblossom package.
library(cherryblossom)
head(run09,5)
```

```
## # A tibble: 5 x 14
##   place  time net_time  pace   age gender first  last  city  state country
div
##   <int> <dbl>    <dbl> <dbl> <int> <fct>  <fct>  <fct> <fct> <fct> <fct>
<int>
## 1     1  53.5     53.5  5.37    21 F      Lineth Chep~ Kenya NR    KEN
2
## 2     2  53.9     53.9  5.4     21 F      Belia~ Gebre Ethi~ NR    ETH
2
## 3     3  54.0     54.0  5.4     22 F      Teyba  Naser Ethi~ NR    ETH
2
## 4     4  54.4     54.4  5.45    19 F      Abebu  Gelan Ethi~ NR    ETH
1
## 5     5  54.4     54.4  5.45    36 F      Cathe~ Nder~ Kenya NR    KEN
5
## # ... with 2 more variables: div_place <int>, div_tot <int>
```

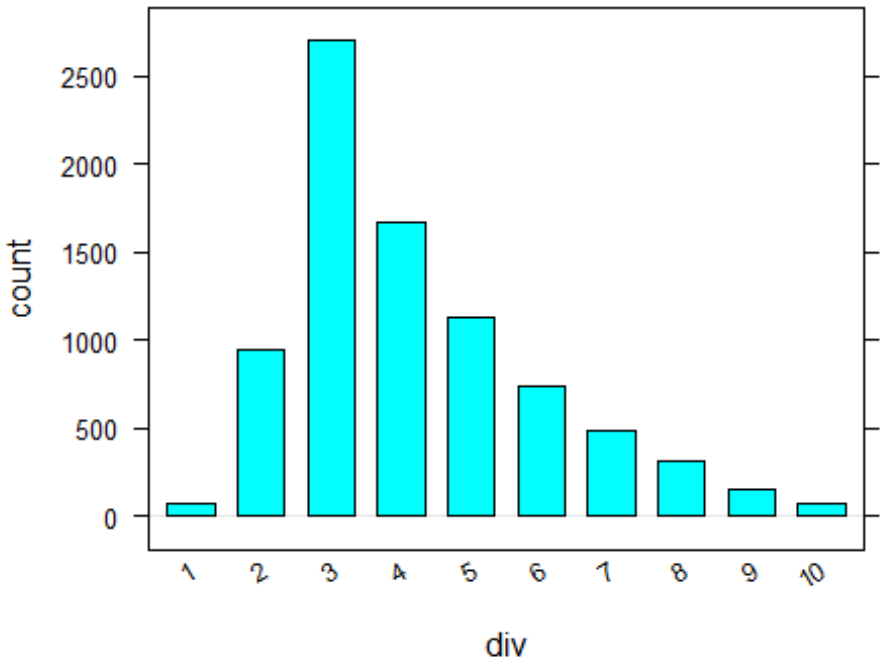
```
# Let's see how many rows there are.
nrow(run09)
```

```
## [1] 14974
```

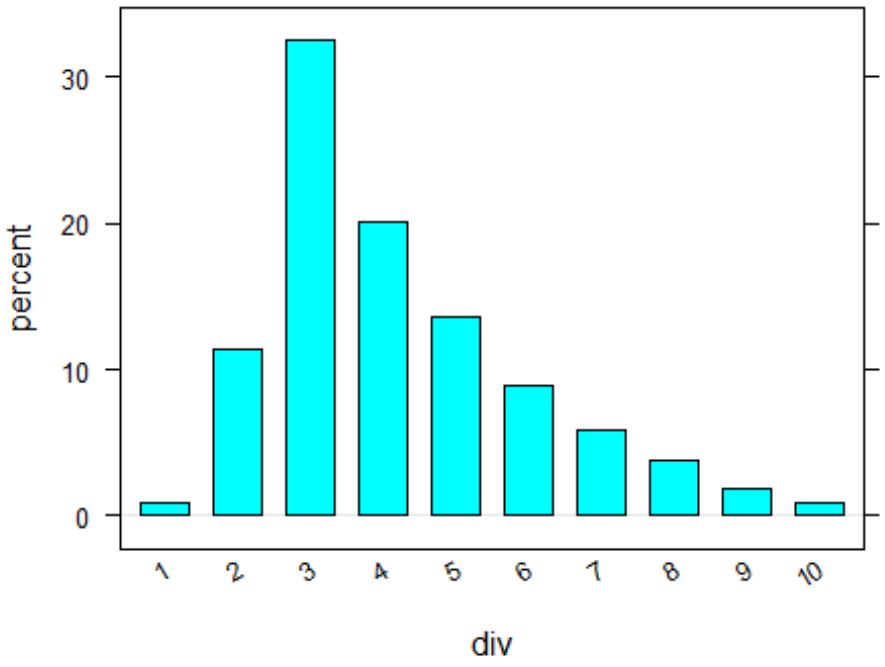
```
# That's a lot!  Let's get a smaller data set to work with.  We
# "filter" the data frame to obtain just the female runners from
# the United States.  We also select just some of the columns and
# combine some of the divisions.
cb09data <- run09 %>%
  filter(gender == "F" & country == "USA" & div != "NA") %>%
  select(place, time = net_time, age, state, div, div_place) %>%
  mutate(div = ifelse(div < 11, div, 10))
head(cb09data,5)
```

```
## # A tibble: 5 x 6
##   place  time   age state   div div_place
##   <int> <dbl> <int> <fct> <dbl>     <int>
## 1     7  54.6    25 NR        3         2
## 2    14  55.7    28 CO        3         5
## 3    15  55.8    26 NR        3         6
```

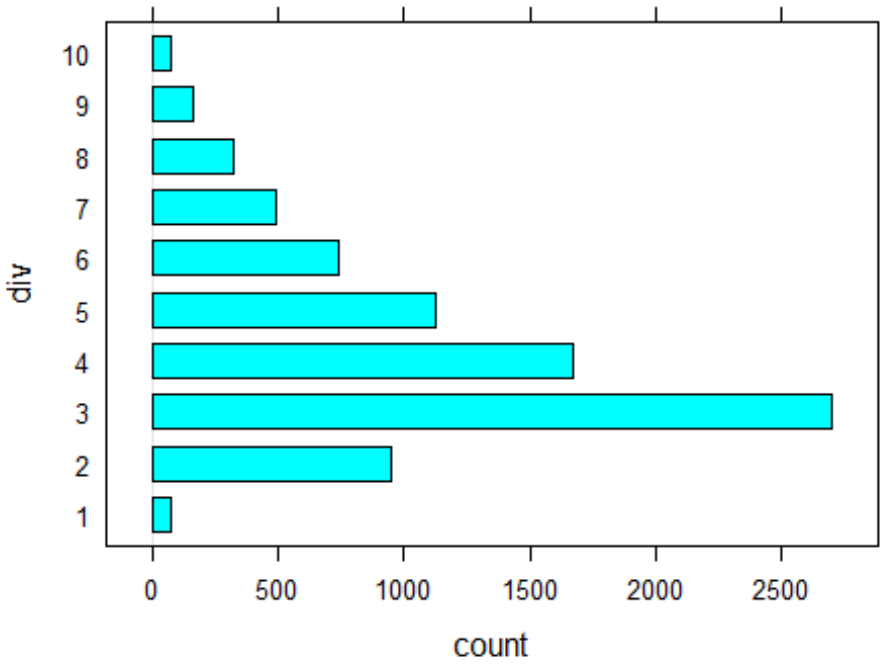
```
## 4    16  55.9    29 NR        3         7
## 5    17  56.3    28 NR        3         8

# Bar Graphs.
# We can produce a bar graph using the bargraph() command from the
# mosaic package.  Here's a bar graph showing the number of runners
# in each division.
bargraph(~ div, data = cb09data)
```



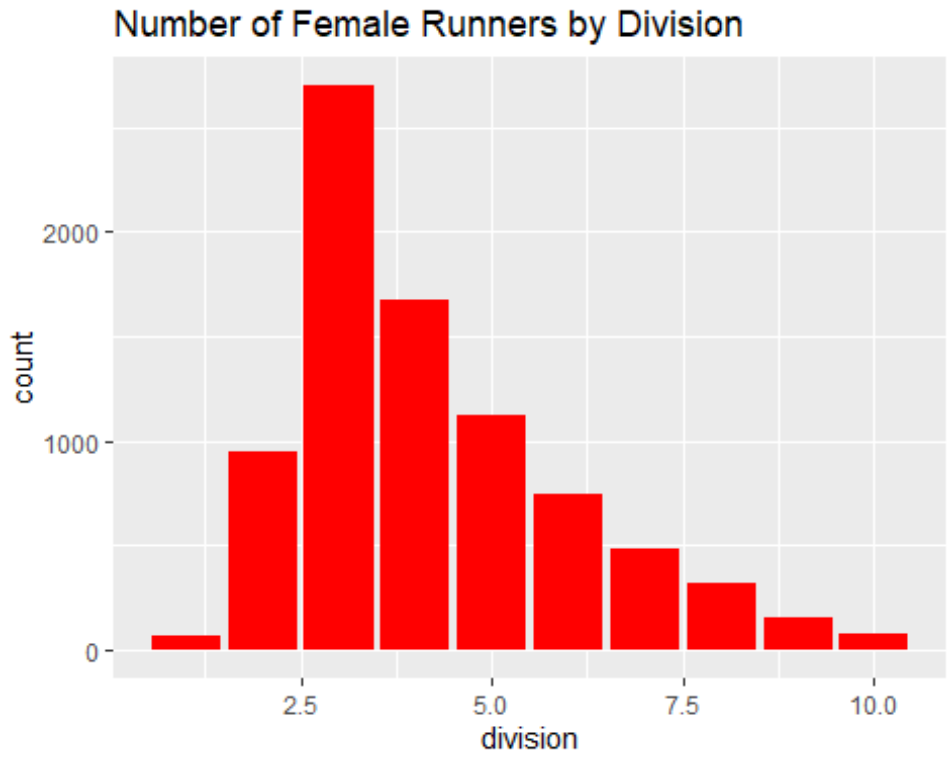

```
# We can show percentages rather than counts using the type option.
bargraph(~ div, data = cb09data, type = "percent")
```



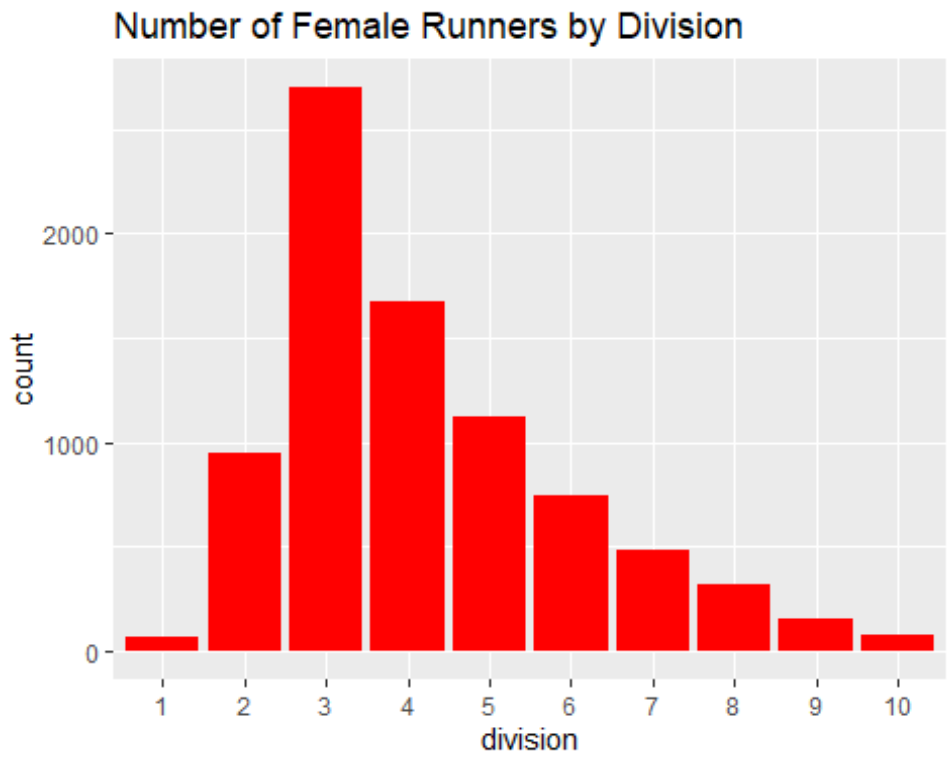

```
# We can make the bars horizontal, rather than vertical.
bargraph(~ div, data = cb09data, horizontal = TRUE)
```



```r
# For more options, use the gf_bar() command.  For example, we can
# change the fill color of the bars, add a label to the horizontal
# axis, and give our graph a title.
gf_bar(~ div, data = cb09data, fill = "red",
       xlab = "division",
       title = "Number of Female Runners by Division")
```



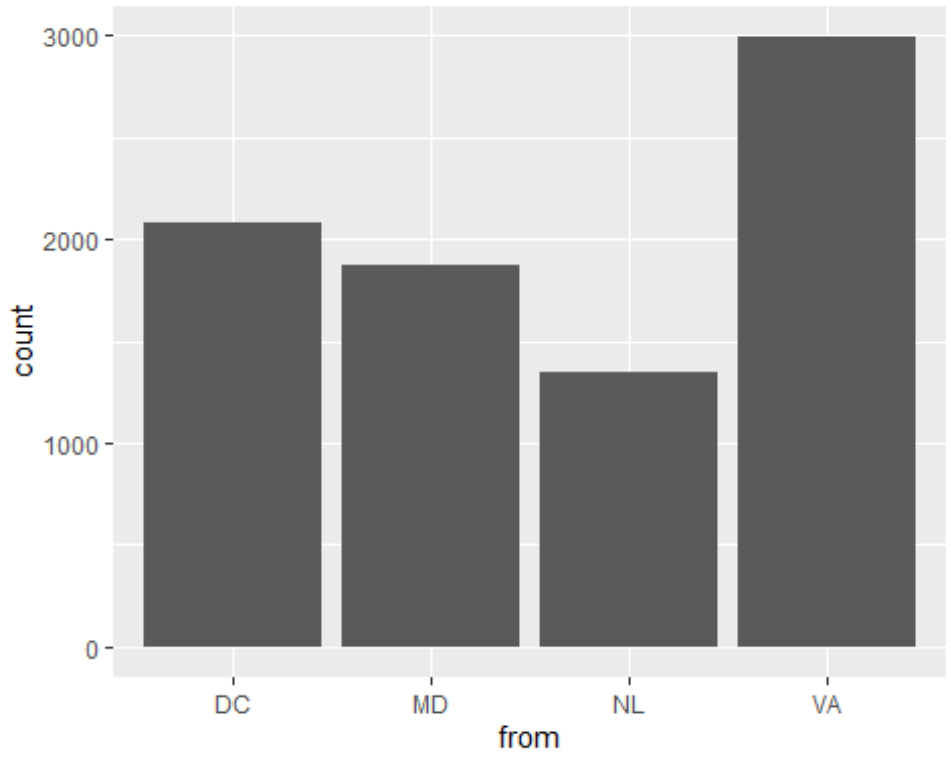

```r
# I don't like the way the horizontal axis is labeled.  This can be
# fixed by interpreting the values of div as factors rather than as
# integers.
gf_bar(~ as.factor(div), data = cb09data, fill = "red",
       xlab = "division",
       title = "Number of Female Runners by Division")
```



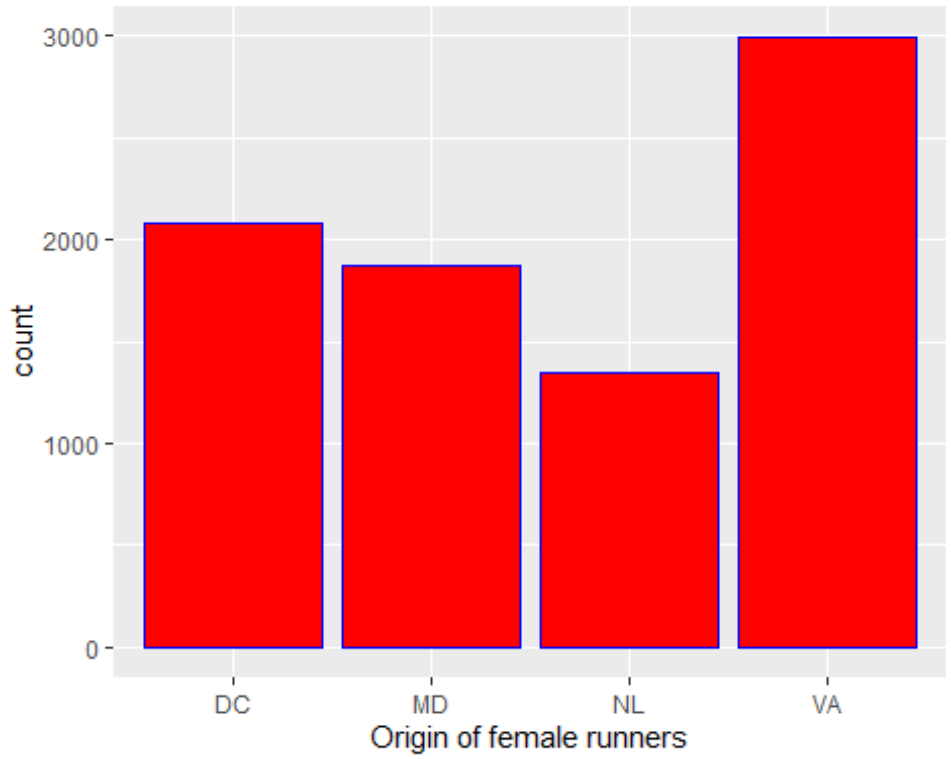

```
# We can get even fancier using ggplot() from the ggplot2 package,
# which is one of the packages that comes with tidyverse.
# To illustrate, we add a column "from" to our data frame to show
# whether a runner is from DC, MD, VA, or some other state
# (NL, for "not local").
cb09rev <- cb09data %>%
  mutate(from = ifelse(state %in% c("DC","MD","VA"),
                       as.character(state), "NL"))
cb09rev[16:20,]
```

```
## # A tibble: 5 x 7
##   place  time   age state   div div_place from
##   <int> <dbl> <int> <fct> <dbl>     <int> <chr>
## 1    30  62.1    27 NY        3        13 NL
## 2    31  62.3    27 VA        3        14 VA
## 3    32  62.4    38 DC        5         5 DC
## 4    33  62.7    25 MD        3        15 MD
## 5    34  62.8    27 DC        3        16 DC
```

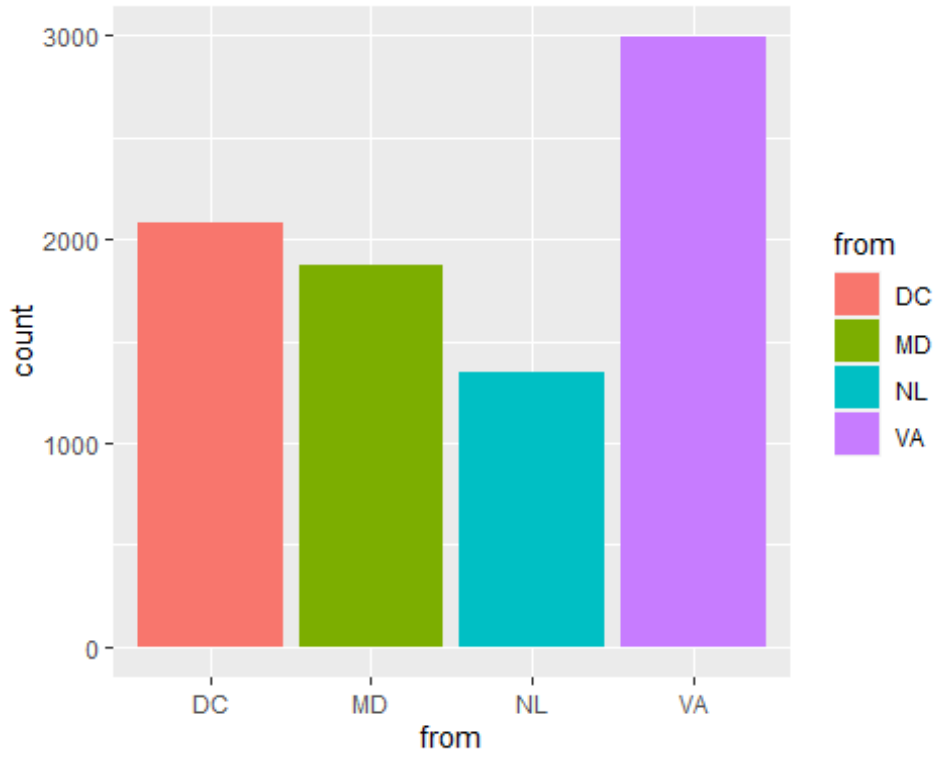
```
# Note: I don't know why I needed to use as.character() in the
# above code, but without it I get strange results.
# Here's a bar graph using ggplot()
plot1 <- ggplot(data = cb09rev, aes(x = from)) +
  geom_bar()
plot1
```



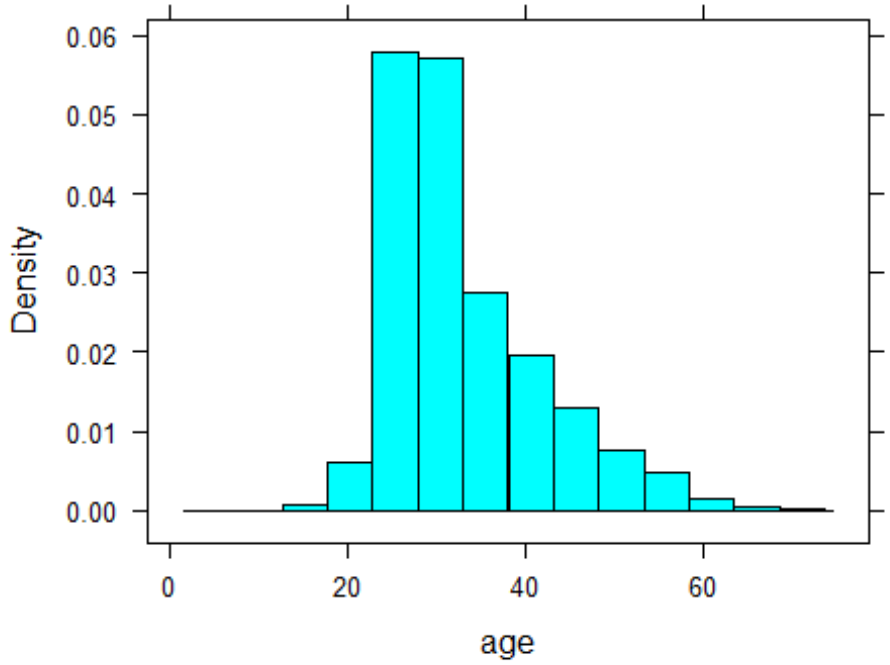

```
# We can change the outline and fill colors of the bars.  We can
# also add a label to the horizontal axis.
plot2 <- plot1 +
  geom_bar(color = "blue", fill = "red") +
  xlab("Origin of female runners")
plot2
```



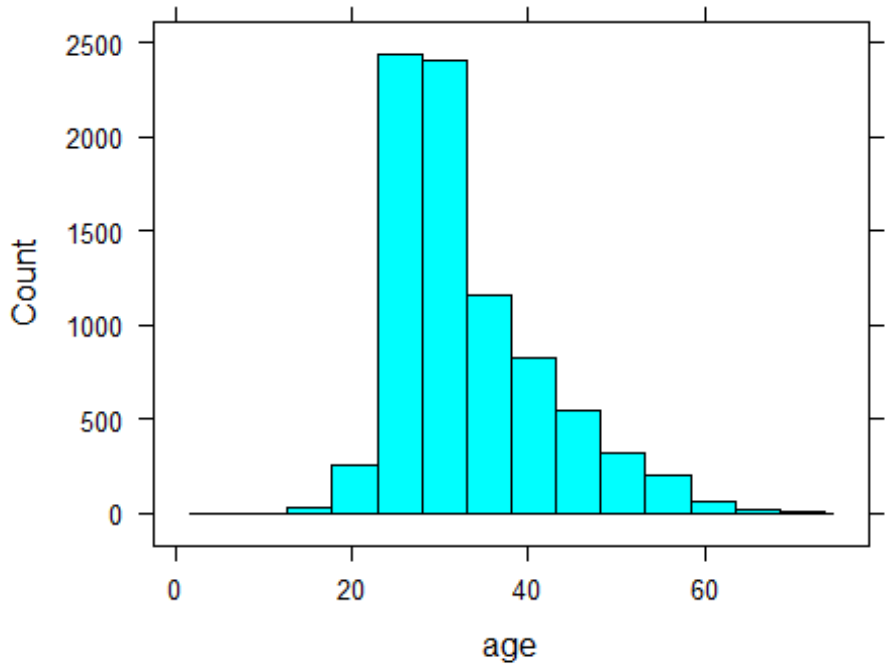

```
# We can even make the fill color of the bars an "aesthetic;"
# that is, have it depend on the value of from.
plot3 <- plot1 +
  geom_bar(aes(fill = from))
plot3
```



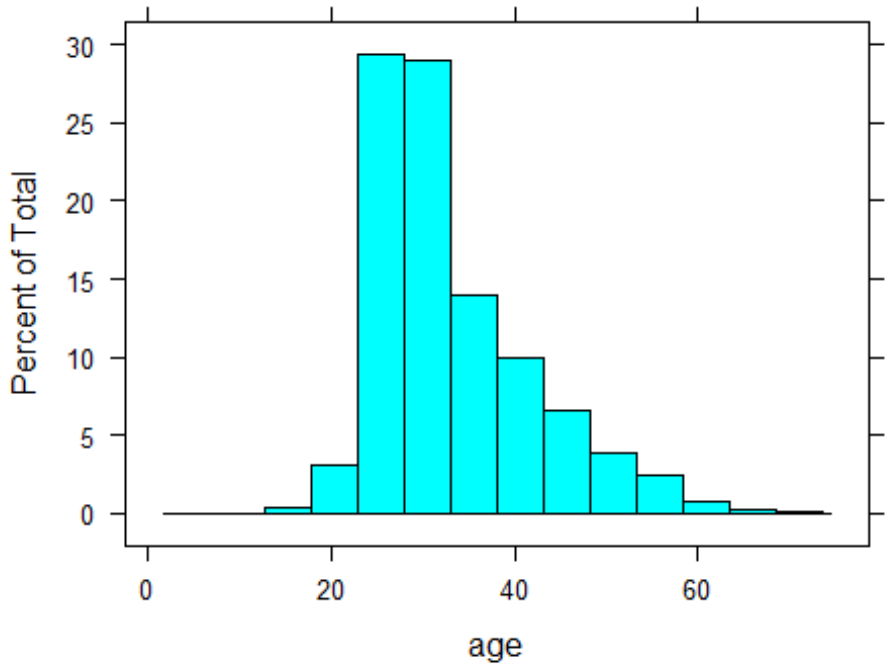

```
# Histograms.
# We can produce a histogram in several ways, among which are the
# following:
# (1) using the histogram() command;
# (2) using the gf_histogram() command;
# (3) using ggplot, along with geom_histogram().
# To illustrate, we produce some histograms for the age variable.
histogram(~ age, data = cb09data)
```



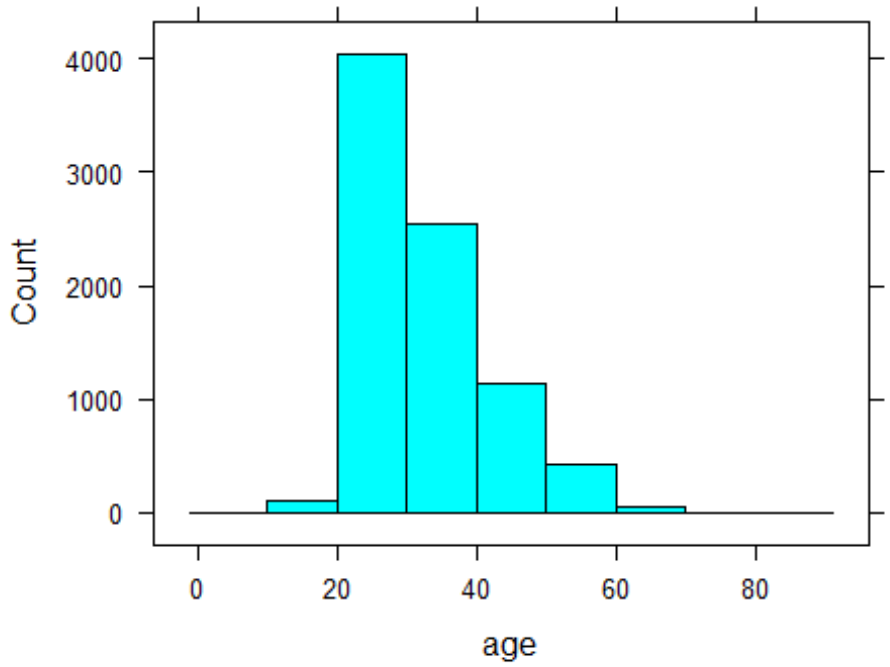

```r
# Not sure what "density" is.  We can use the type option to
# specify whether we want counts or percentages.
histogram(~ age, data = cb09data, type = "count")
```



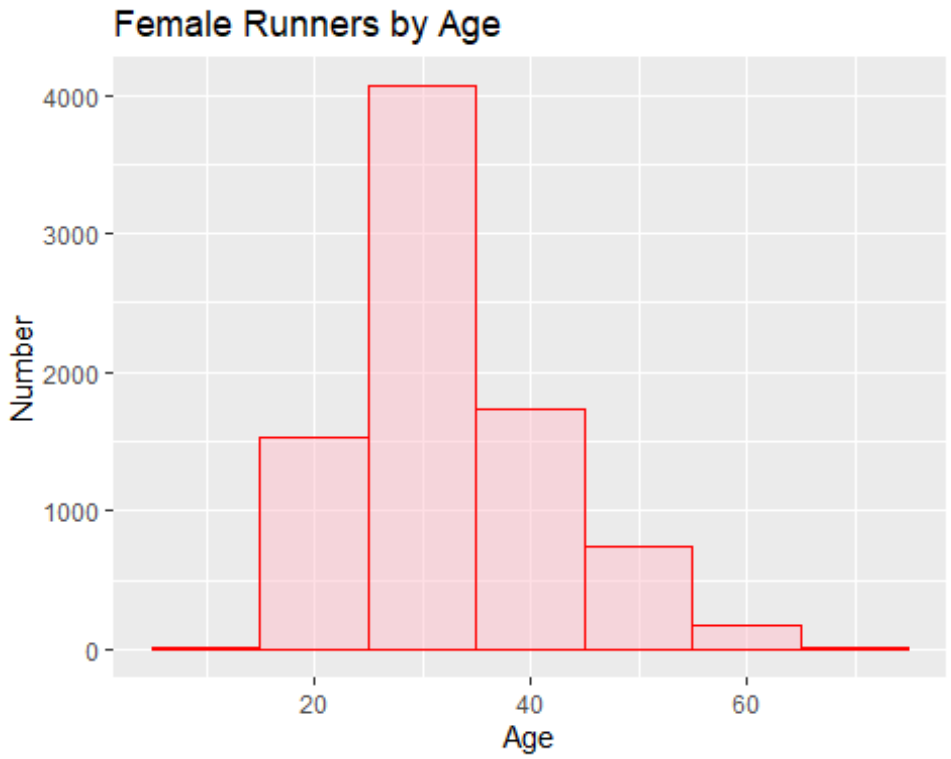
```
histogram(~ age, data = cb09data, type = "percent")
```



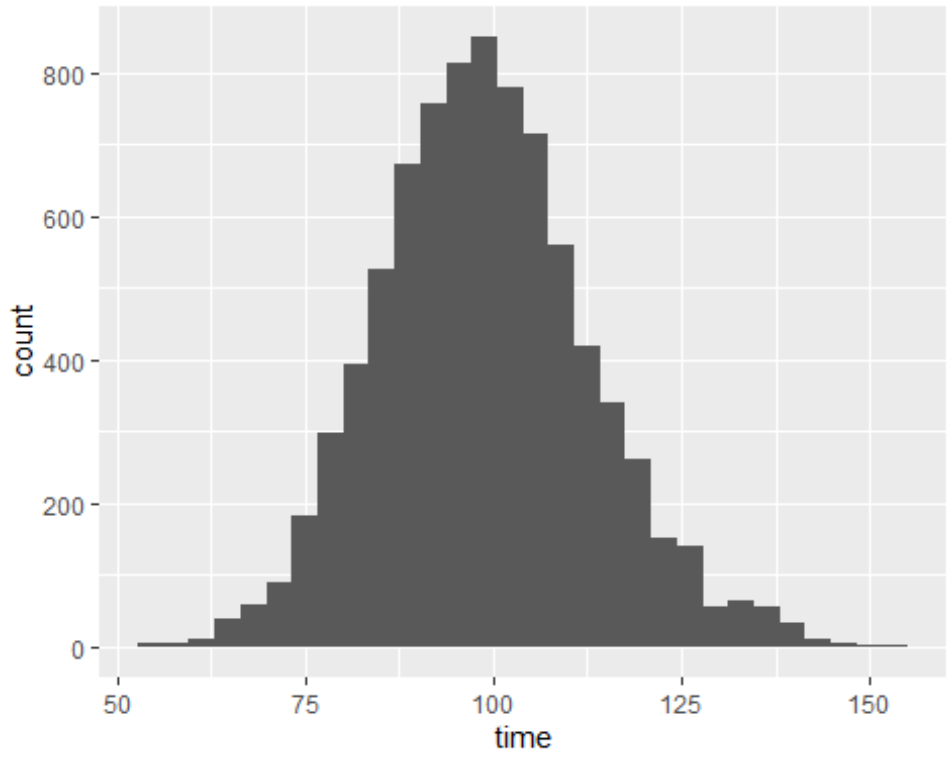

```
# We can control where the break points are for the bins using
# the breaks option.
histogram(~ age, data = cb09data, type = "count",
          breaks = c(0,10,20,30,40,50,60,70,80,90))
```



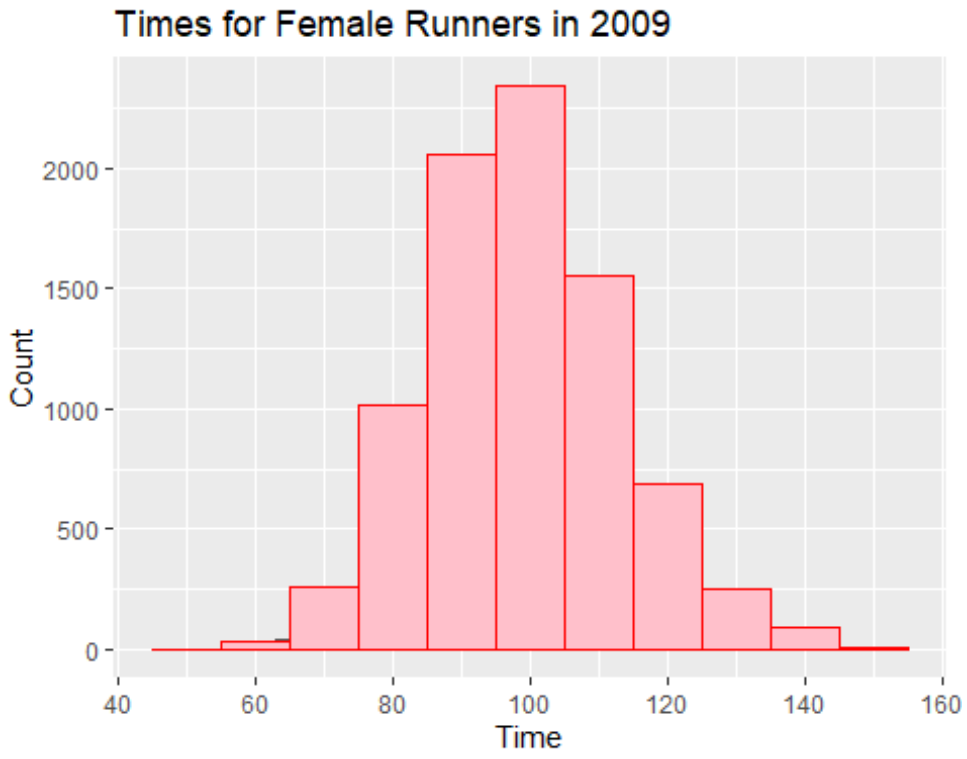

```
# Again, the gf_histogram() command gives us a few more options,
# such as being able to specify the fill and outline colors of the
# bins, and being able to add axes labels and a title.
# Unfortunately, we lose the "breaks" option, although we can
# specify the number and/or the width of the bins.
gf_histogram(~ age, data = cb09data, type = "count",
             binwidth = 10,
             fill = "pink", color = "red",
             xlab = "Age", ylab = "Number",
             title = "Female Runners by Age")
```




```
# I like the "grid" background!
# Let's use ggplot() and geom_histogram() to produce a histogram
# of the time variable.
plot4 <- ggplot(data = cb09data, aes(x = time)) +
  geom_histogram()
plot4

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

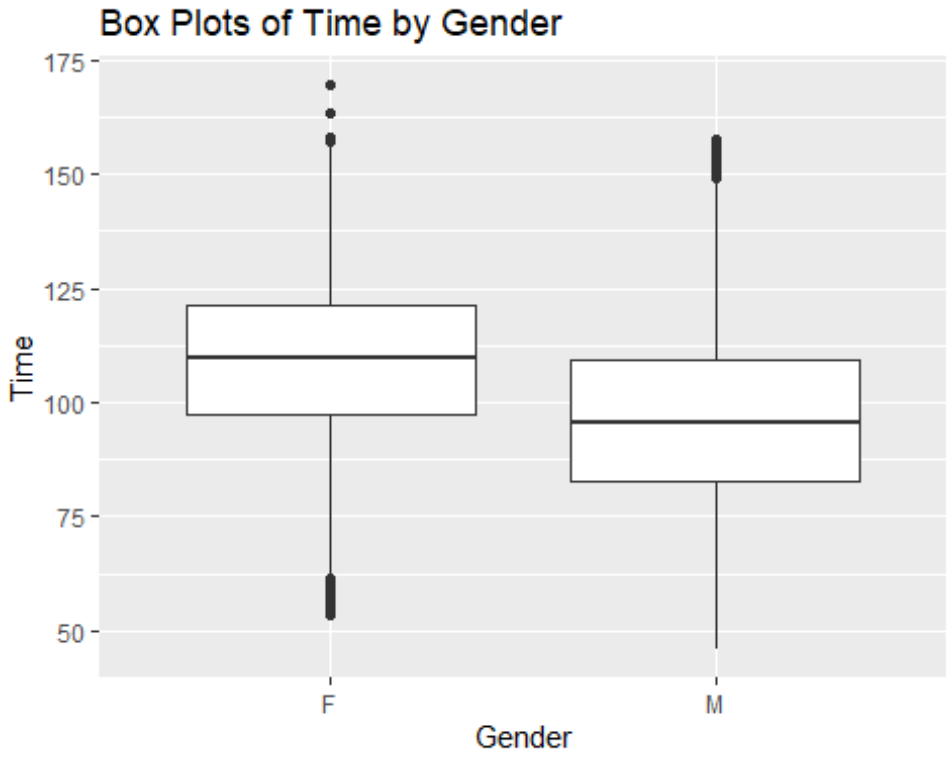



```
plot5 <- plot4 +
  geom_histogram(binwidth = 10, fill = "pink", color = "red") +
  labs(x = "Time", y = "Count",
       title = "Times for Female Runners in 2009")
plot5

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

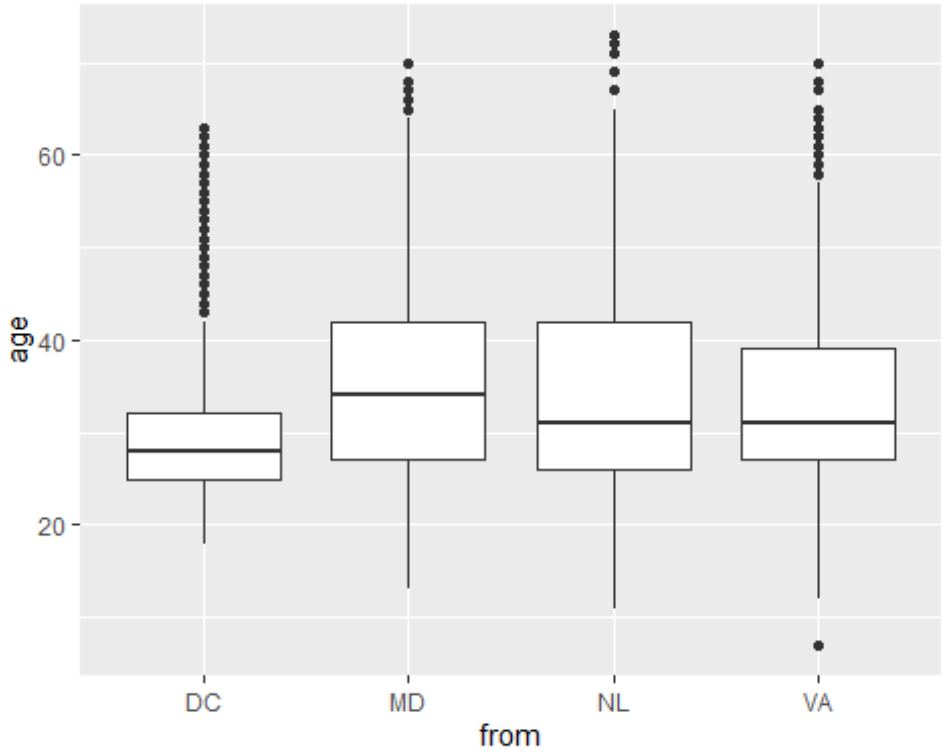



```r
# Box Plots.
# Here is a box plot for the age variable using boxplot():
boxplot(cb09data$age, xlab = "Female Runners", ylab = "Age")
```

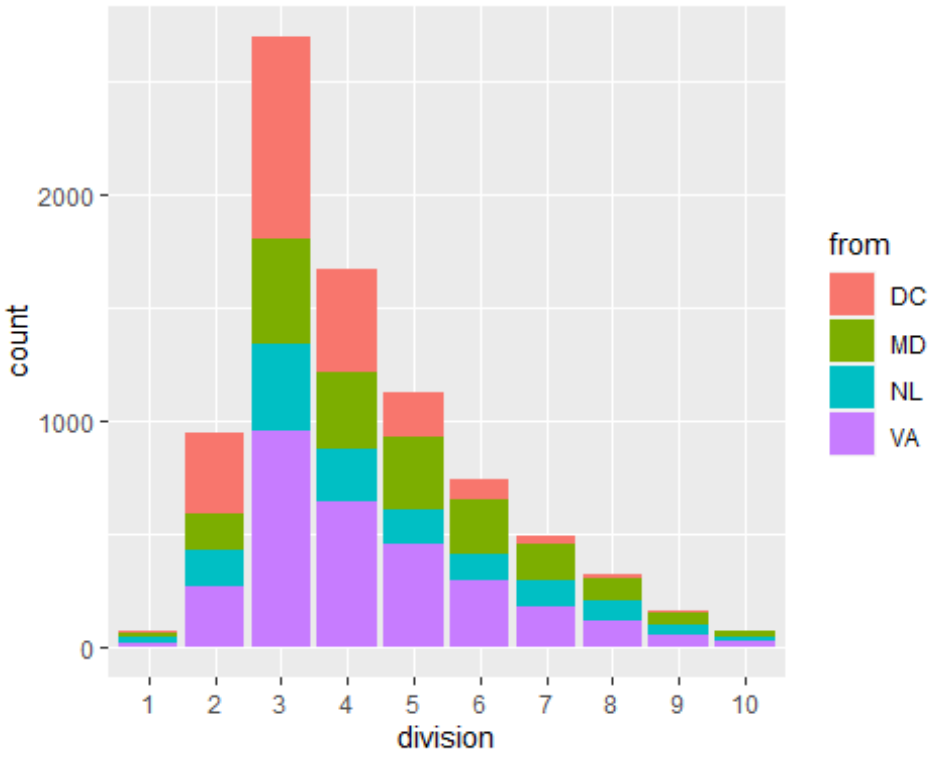
```r
# Separate box plots for the female times and male times, this time
# using gf_boxplot():
gf_boxplot(time ~ gender, data = run09,
           xlab = "Gender", ylab = "Time",
           title = "Box Plots of Time by Gender")
```




```r
# Here is a box plot produced using ggplot() and geom_boxplot().
plot6 <- ggplot(data = cb09rev, aes(x = from, y = age)) +
  geom_boxplot()
plot6
```




```
# Here's something a bit more advanced - a segmented bar graph.
# We produce a bar graph showing the numbers of runners in each
# division, segmented using the "from" variable.
plot7 <- ggplot(data = cb09rev, aes(x = as.factor(div))) +
  geom_bar(aes(fill = from)) + xlab("division")
plot7
```


count
2000
1000
0
1
2
3
4
5
6
7
8
9
10
division
from
DC
MD
NL
VA