Visualizing Frequency Distributions

No exercício anterior aprendemos a criar Tabelas Frequência, e entendemos a importância que ela tem para simplificar os dados e possibilitar análises em bases muito grandes. Ainda assim apenas olhar números, seja absolutos ou relativos (percentuais) pode ser difícil realizar comparações e percorrer o olhar ao longo de vários números e classificações para extrair algum insight.

Com isso as visualizações cumprem um papel importante para facilitar essas análises e de forma mais rápida e assertiva entender o comportamento dos dados.

Explorando dados do WNBA

O dataset pode ser acessado através deste link.

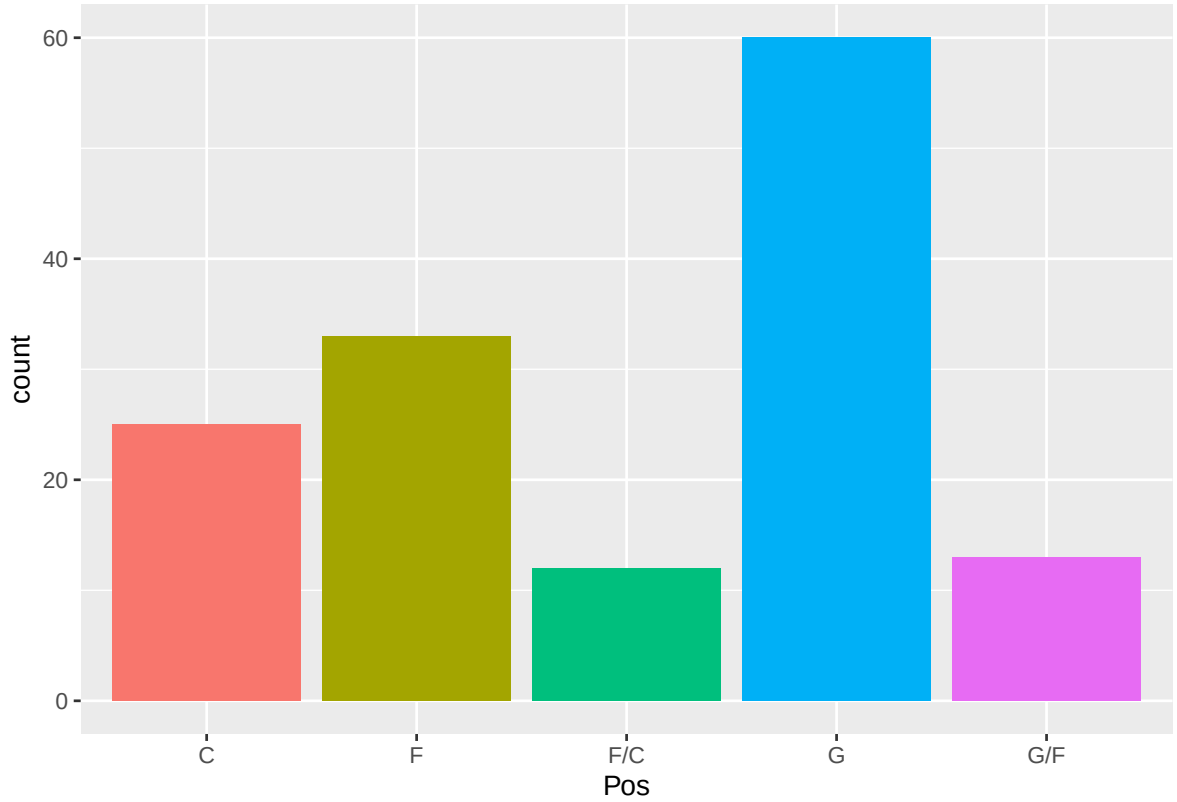
E o glossário dos termos neste link.

```
wnba <- read_csv("WNBA_Stats.csv")
```

```
## Rows: 143 Columns: 32
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): Name, Team, Pos, Birth_Place, Birthdate, College, Experience
## dbl (25): Height, Weight, BMI, Age, Games Played, MIN, FGM, FGA, FG%, 15:00,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Para visualizar a tabela frequência da distribuição das jogadoras em cada posição, podemos diretamente usar a biblioteca ggplot2 através da função geom_bar. Não é necessário previamente criar a tabela frequência, apenas informando a base e a coluna de interesse, a própria função vai calcular a frequência e mostrar o resultado através do gráfico.

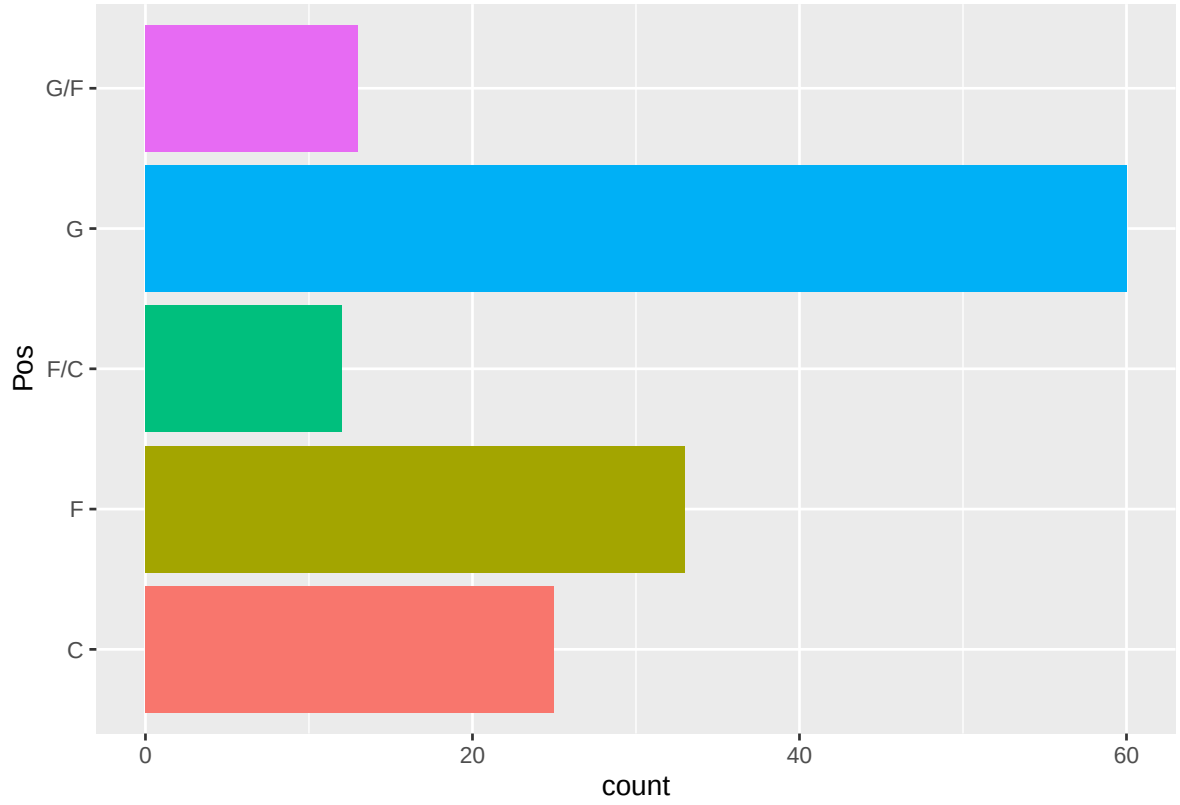
```r
ggplot(wnba,
       aes(x=Pos, fill=Pos))+
  geom_bar()+
  theme(legend.position = "none")
```



Repetir a coluna no parâmetro fill possibilitou colorir facilmente as barras e reforçar que são categorias distintas. No entanto acaba gerando uma legenda que seria redundante então o parâmetro legend.position = “none” ajudou a oculta-la.

Uma forma de transformar um gráfico de barras verticais em horizontais é usando a função `coord_flip`

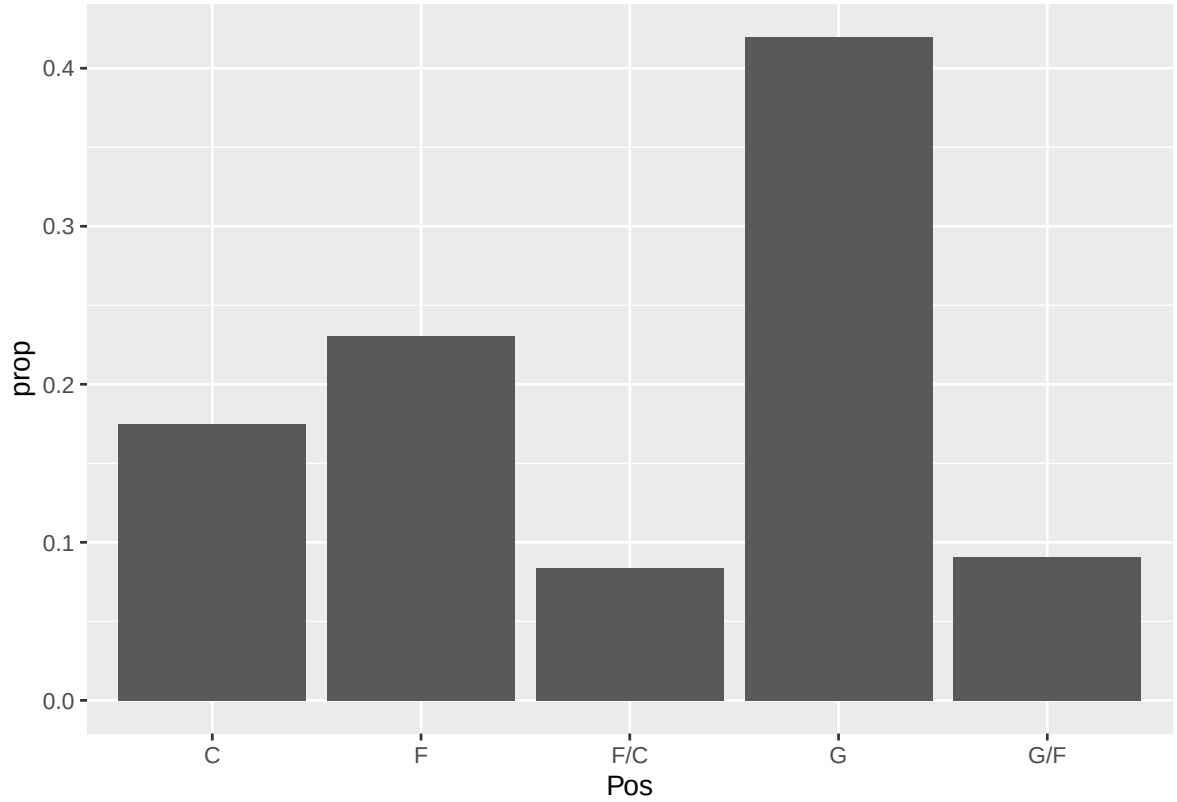
```r
ggplot(wnba,
       aes(x=Pos, fill=Pos))+
  geom_bar()+
  coord_flip()+
  theme(legend.position = "none")
```



É possível no eixo y apresentar a proporção ao invés do valor absoluto da frequência. Basta adicionar o parâmtro y com o valor `..prop..` e o parâmetro group = 1.

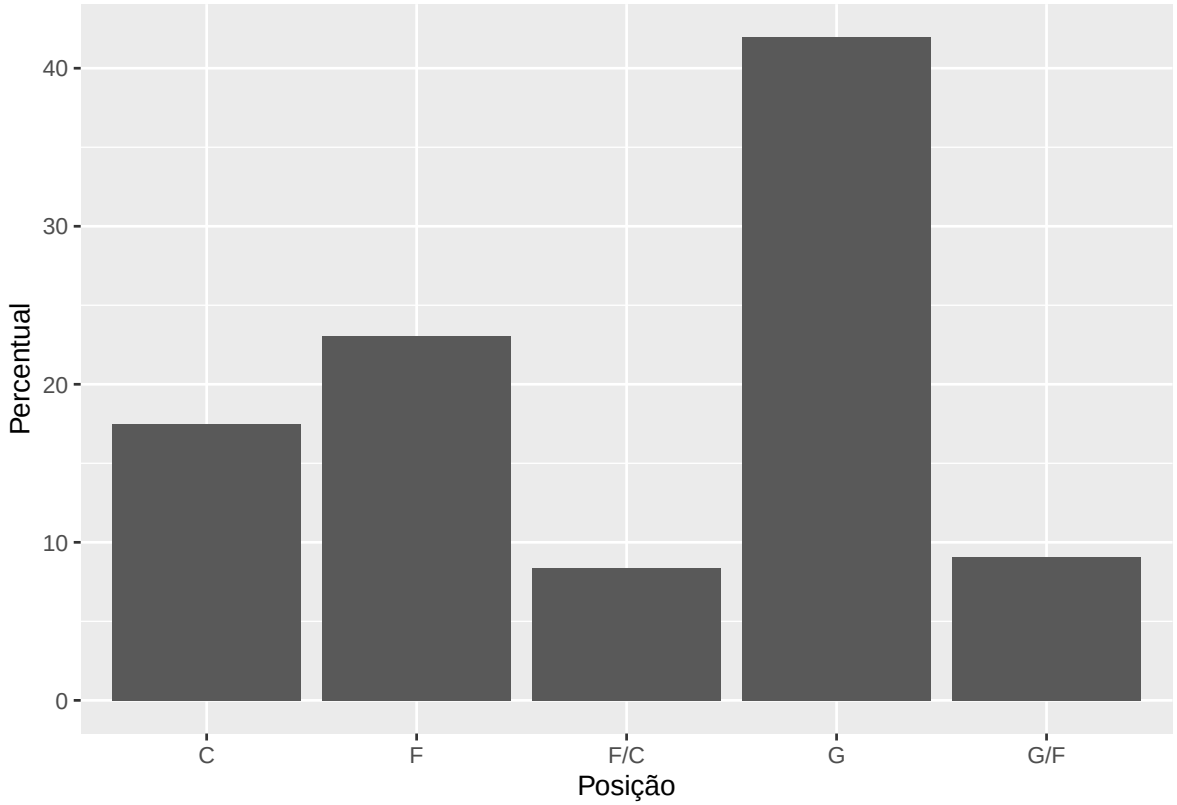
```
ggplot(wnba,
       aes(x=Pos,
           y=..prop..,
           group=1))+
  geom_bar()+
  theme(legend.position = "none")
```

```
## Warning: The dot-dot notation (`..prop..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(prop)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



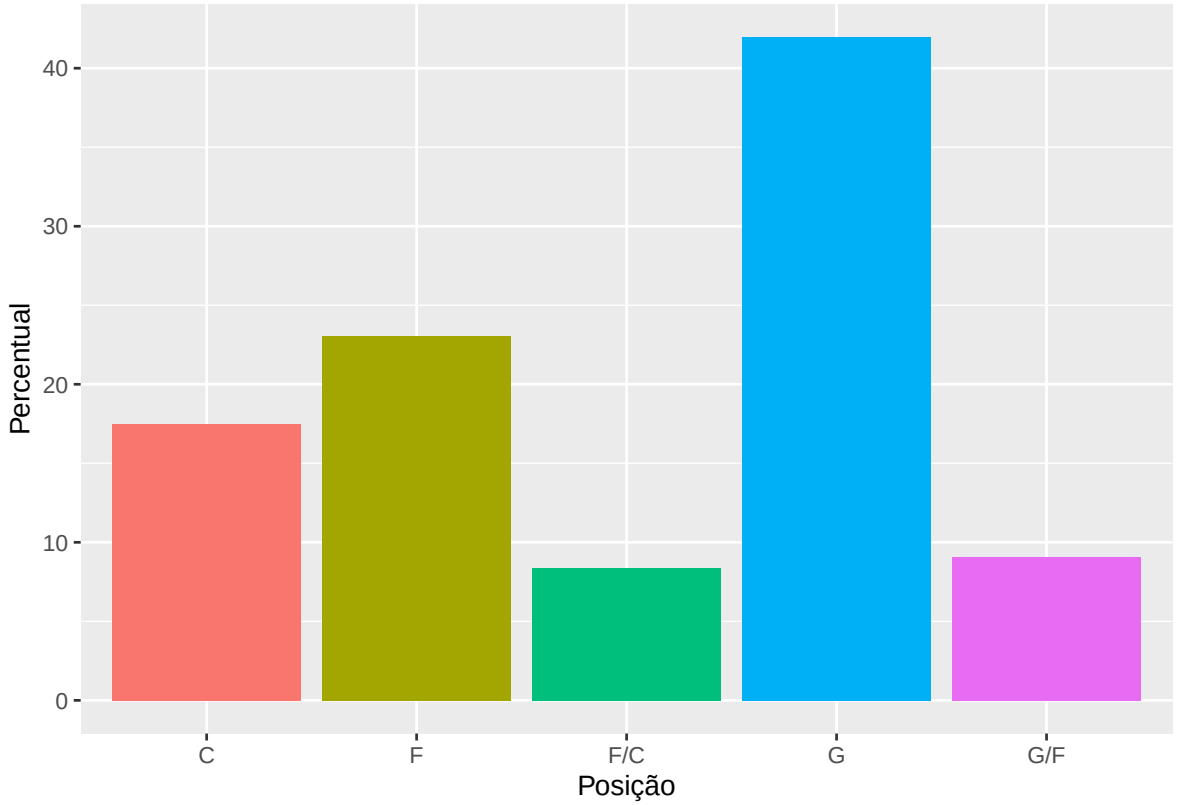
Também é possível apresentar no formato de percentual. Para melhorar o entendimento das informações podemos alterar os labels dos eixos x e y.

```
ggplot(wnba,
       aes(x=Pos,
           y=..prop.. * 100,
           group=1))+
  geom_bar()+
  theme(legend.position = "none") +
  labs(x="Posição",
       y="Percentual")
```



Utilizando propoção ou percentual fica um pouco diferente para conseguir colorir as barras com o parâmetro fill.

```
ggplot(wnba,
       aes(x=Pos,
           y=..prop.. * 100,
           group=1,
       fill=factor(..x..)))+
  geom_bar()+
  theme(legend.position = "none") +
  labs(x="Posição",
       y="Percentual")
```



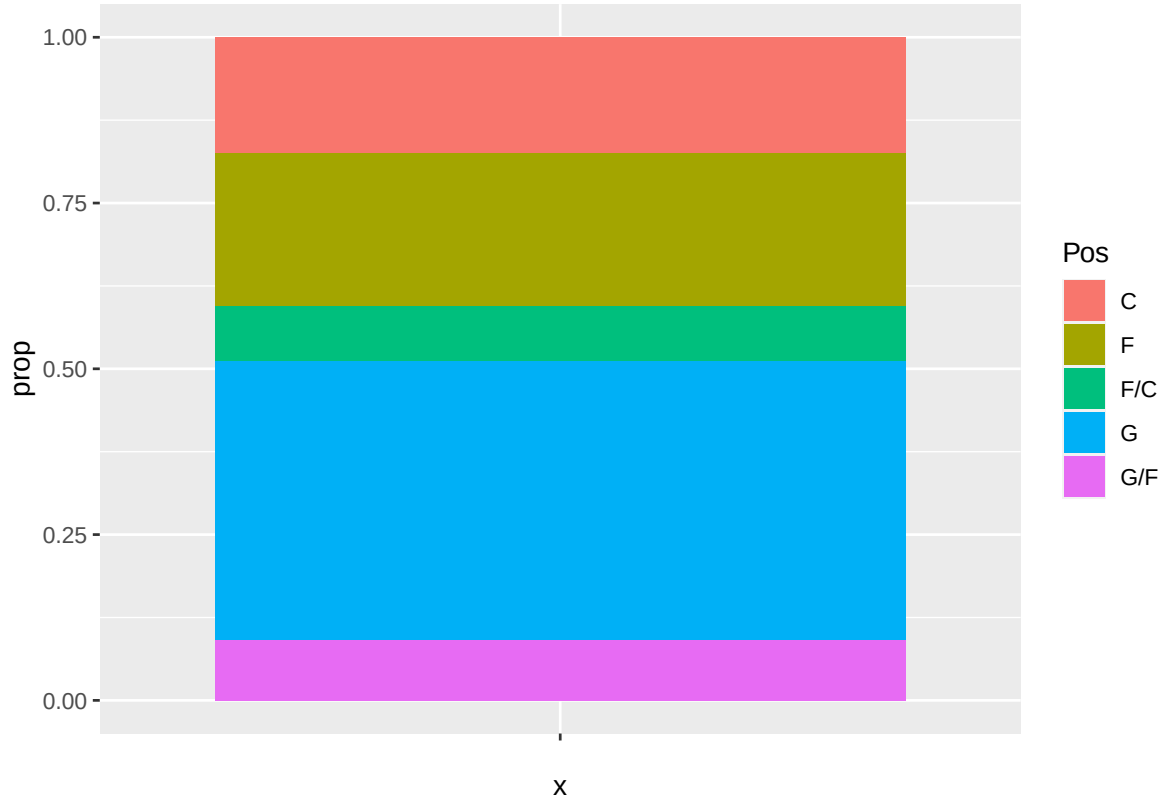
Stacked Bar

Esse é um tipo de gráfico de barra onde empilhamos uma barra na outra formando uma única barra com várias informações que representam o todo.

No exemplo que estamos utilizando, a barra representa a base completa que é dividida nas 5 posições existentes. E o tamanho de cada parte da barra corresponde à proporção daquela informação na base.

Para isso vamos calcular as proporções e em seguida usar no gráfico de barras com algumas pequenas mudanças.

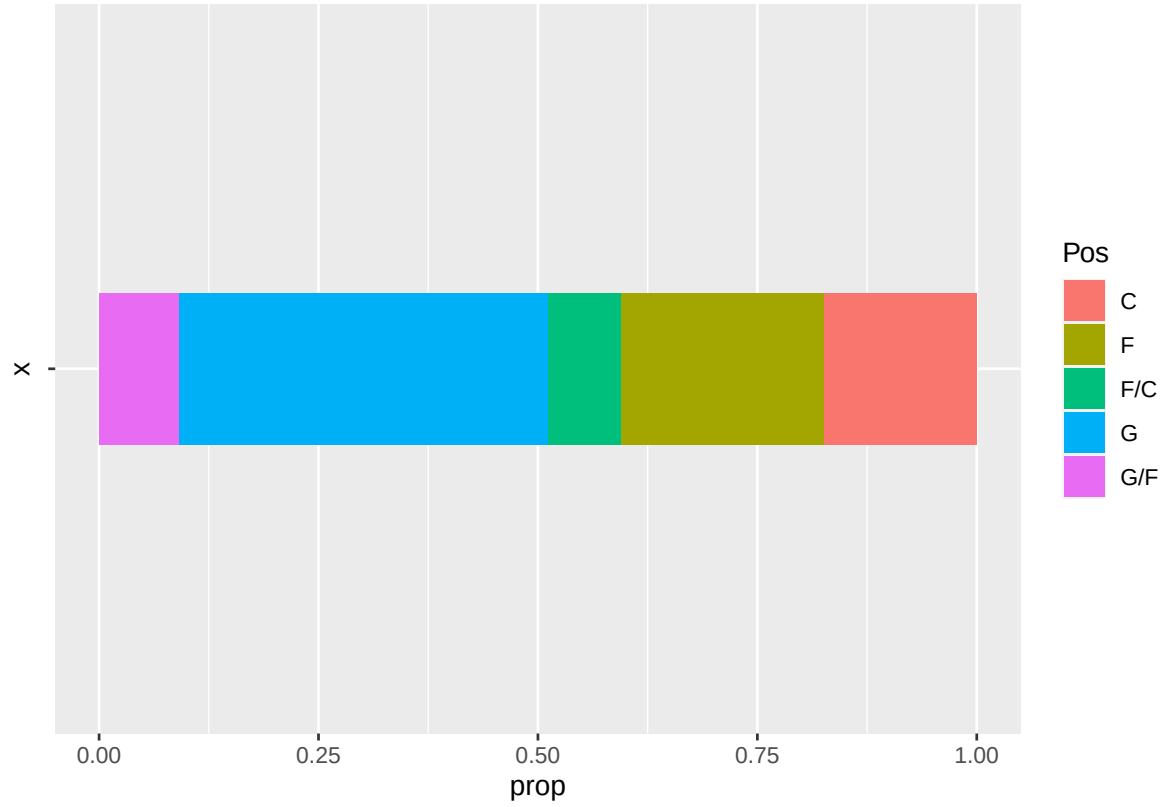
```
prop_pos <- wnba %>%
  group_by(Pos) %>%
  summarise(prop = n()/nrow(wnba))

prop_pos %>%
  ggplot(aes(x="",
             y=prop,
             fill=Pos))+
  geom_bar(stat="identity")
```



Dois pequenos detalhes são importantes, como não precisamos de várias barras, apenas uma barra com todas categorias, nós passamos a informação vazia “ ” ao parâmetro x. Como também já deixamos previamente calculado o que desejamos visualizar, é necessário informar o valor “identity” ao parâmetro *stat*.

Para melhorar a visualização do gráfico acima, podemos exibi-lo horizontalmente e afinar a grossura da barra.

```
prop_pos %>%
  ggplot(aes(x="",
             y=prop,
             fill=Pos))+
  geom_bar(stat="identity", width=0.25)+
  coord_flip()
```



Ainda assim é possível simplificar e melhorar a visualização, abaixo mais um exemplo:

```
ggplot(data = prop_pos,
               aes(x = "", y = prop, fill = Pos)) +
  geom_bar(stat = "identity", width = 0.25) +
  coord_flip() +
  geom_text(aes(label = str_c(round(prop * 100), "%")),
            position = position_stack(vjust = 0.5)) +
  labs(x = NULL,
       y = NULL,
       fill = NULL,
       title = "Player Distribution by Position") +
  theme_classic() +
  theme(axis.line = element_blank(),
          axis.text = element_blank(),
          axis.ticks = element_blank())
```

Player Distribution by Position

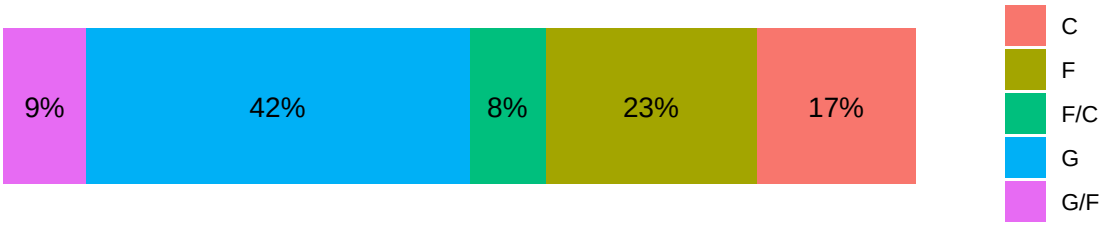


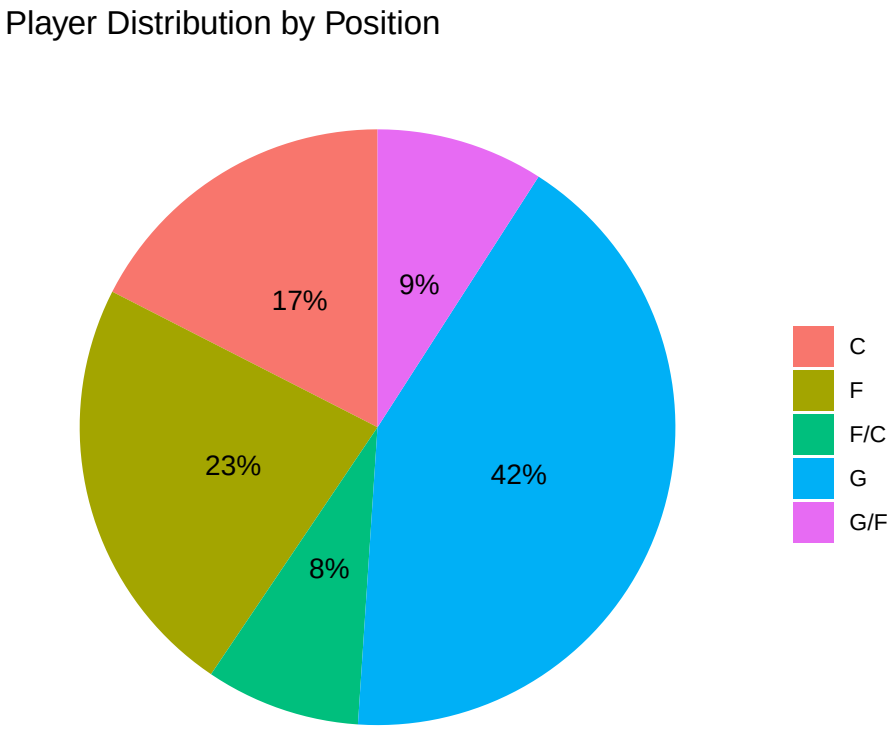
Gráfico de Pizza (Pie Chart)

É um tipo de gráfico com péssima reputação, mas há quem o defenda e ainda é bastante utilizado, então vamos aprender a cria-lo. Alguns defendem que quanto mais categorias mais difícil de entender as proporções, e se as categorias tem proporções muito próximas fica dificil de determinar qual é maior que qual.

Por outro lado pode ser um gráfico mais amigável para quem não está acostumado com gráficos mais complexos para ter a noção do todo e as partes que cada categoria representa.

```
ggplot(data = prop_pos,
               aes(x = "", y = prop, fill = Pos)) +
  geom_bar(stat = "identity", width = 0.25) +
  coord_polar(theta="y") +
  geom_text(aes(label = str_c(round(prop * 100), "%")),
            position = position_stack(vjust = 0.5)) +
  labs(x = NULL,
       y = NULL,
       fill = NULL,
```

```
      title = "Player Distribution by Position") +
  theme_classic() +
  theme(axis.line = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank())
```

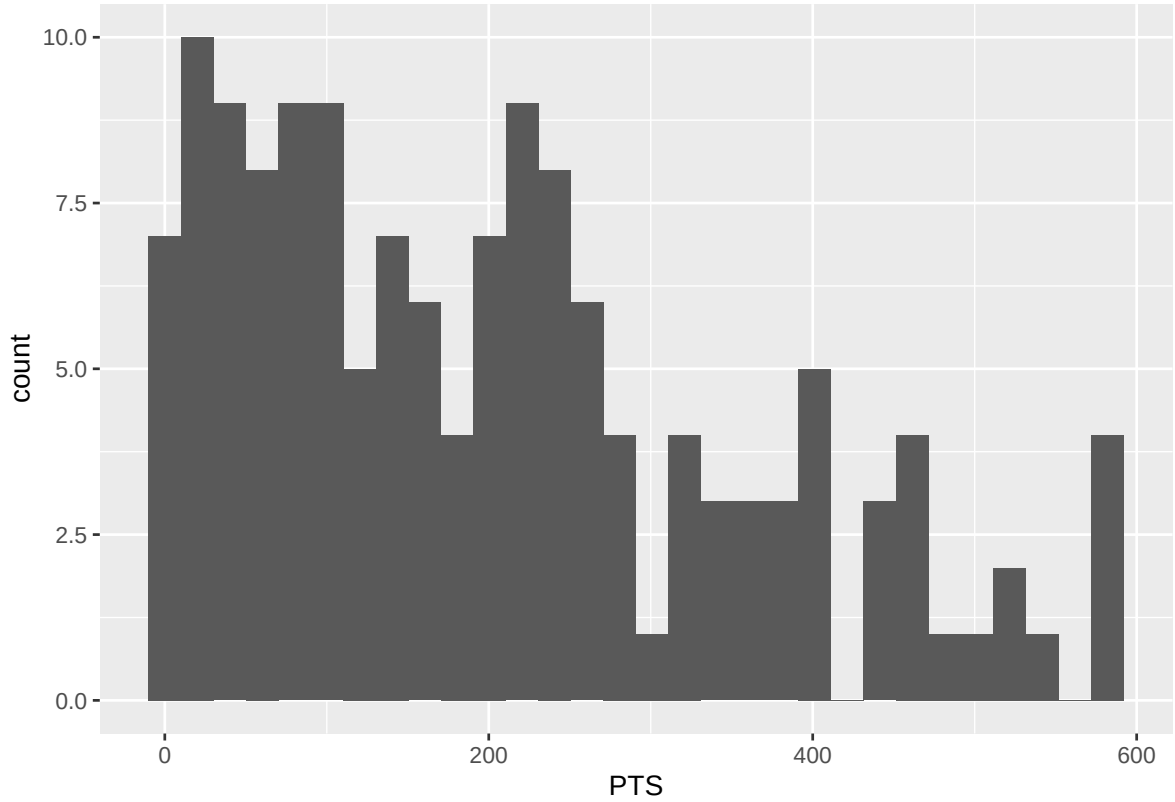


Histograma

Quando temos uma informação contínua com muitas categorias, uma gráfico que ajuda a entender melhor a distribuição é o histograma. Diferentemente do gráfico de barra, aqui as barras estão “grudadas” umas nas outras pra passar a ideia de continuidade.

```
wnba %>%
  ggplot(aes(x=PTS)) +
  geom_histogram()
```

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

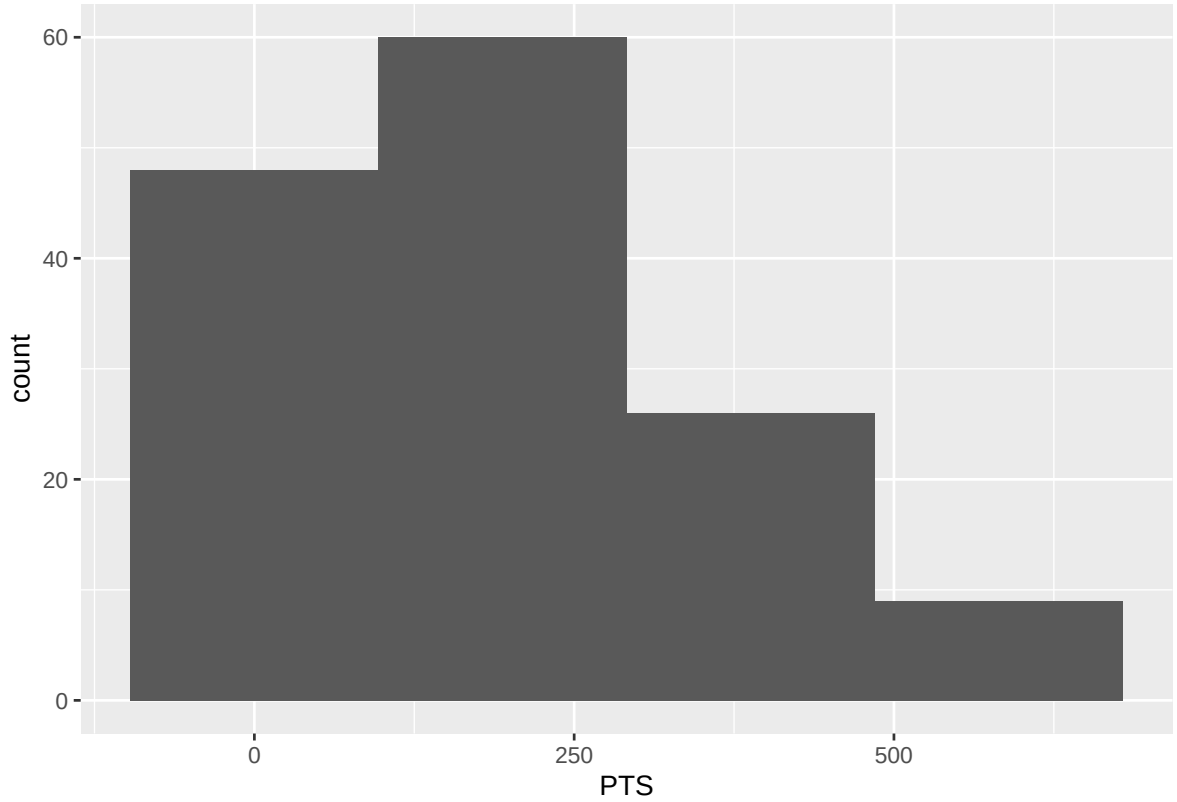


As barras são chamadas de bins, que é a quantidade de grupos que definem os intervalos assim como na Tabela Frequência. No entanto na tabela frequência quanto mais bins, ou quanto mais grupos/intervalos, mais difícil fica de compreender os dados. Já no caso do histrograma não temos tanto essa limitação, usar mais bins pode melhorar a visualização.

Exemplo com granularidade muito baixa

Fica difícil encontrar padrões, entender qual a disposição da base, em qual valor os dados se acumulam mais ou se acumulam menos.

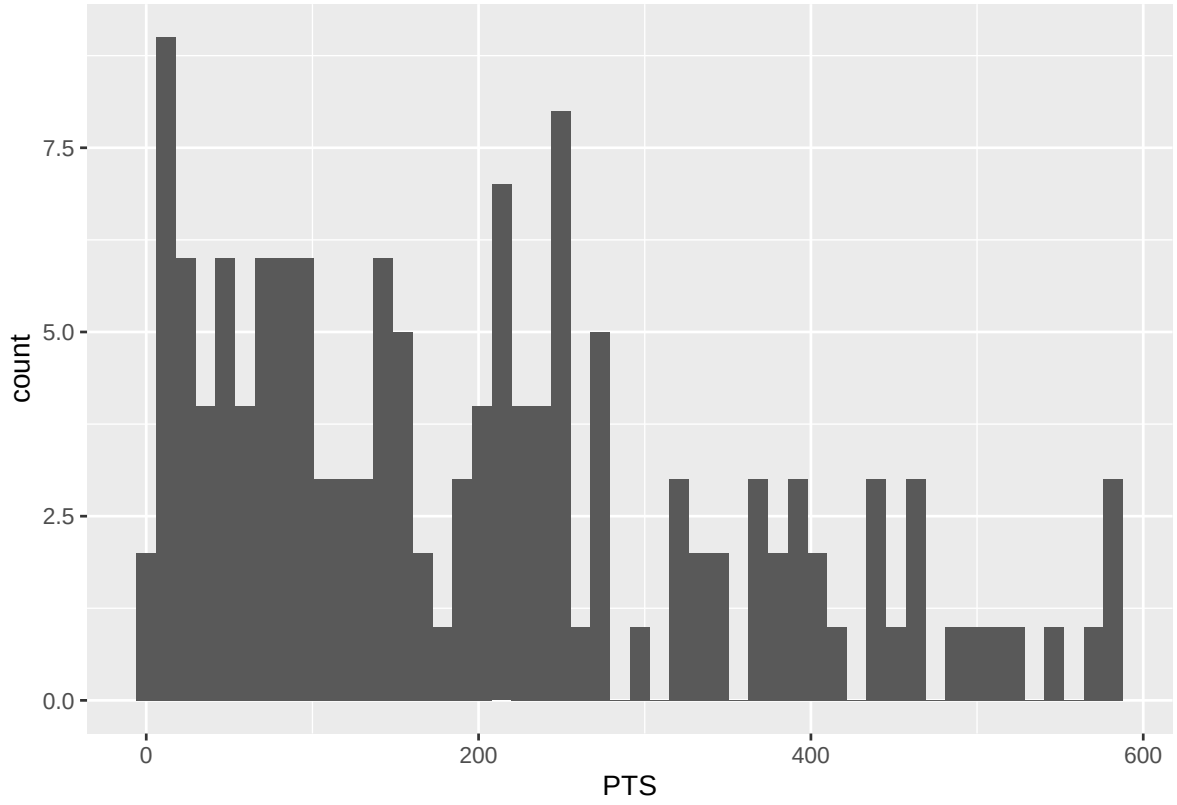
```
wnba %>%
  ggplot(aes(x=PTS)) +
  geom_histogram(bins = 4)
```



Exemplo com granularidade muito alta

A gráfico perde a ideia de continuidade, fica tão quebrado os intervalos que pode deixar de existir observações que se encaixem nas condições do intervalo, ficando então com “buracos”.

```
wnba %>%
  ggplot(aes(x=PTS)) +
  geom_histogram(bins = 50)
```



Comparando com Tabela Frequência

Se o histrograma agrupa em bins, em intervalos, o resultado fica igual ao da tabela frequência?

Histograma com 10 bins

```
wnba %>%
  ggplot(aes(x=PTS)) +
  geom_histogram(bins = 10)
```

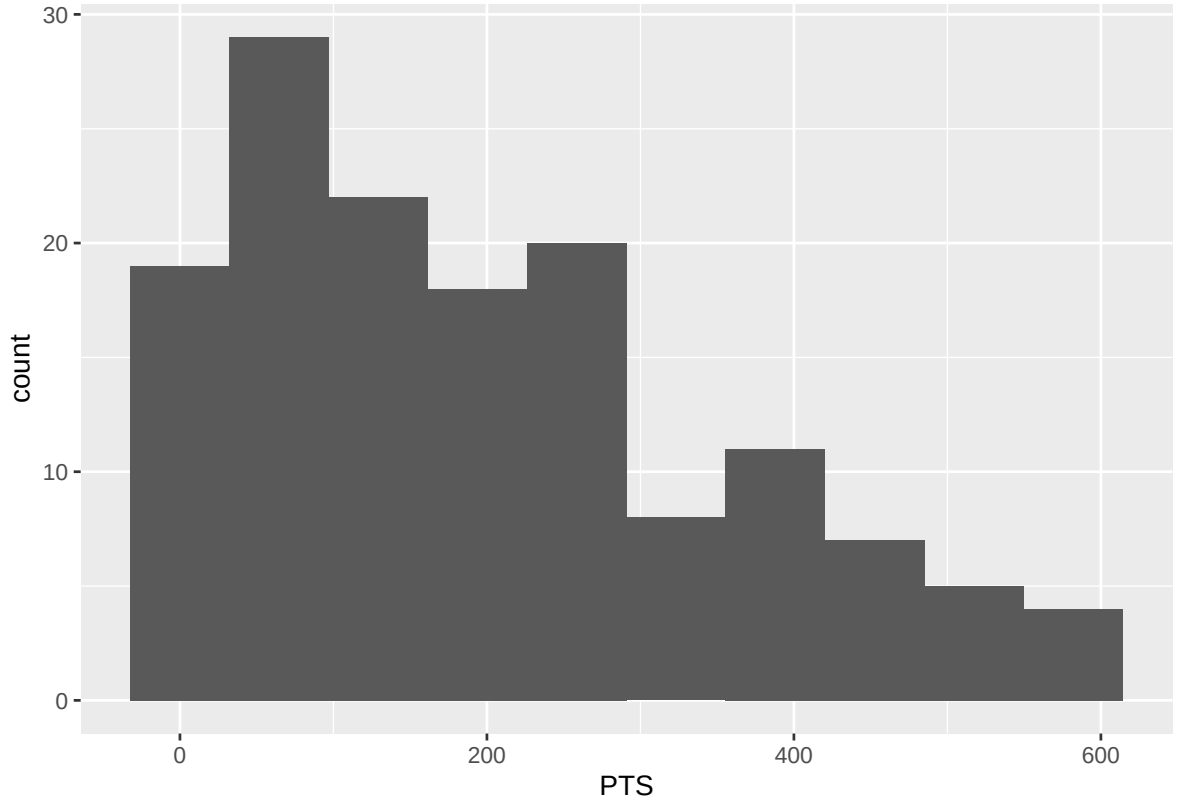


Tabela frequência com 10 grupos

```
wnba <- wnba %>%
  mutate(points_categories = cut(PTS, breaks = 10, dig.lab = 4))

wnba %>%
  group_by(points_categories) %>%
  summarize(Freq = n())
```

```
## # A tibble: 10 x 2
##    points_categories  Freq
##    <fct>             <int>
##  1 (1.418,60.2]         30
##  2 (60.2,118.4]         24
##  3 (118.4,176.6]        17
##  4 (176.6,234.8]        20
##  5 (234.8,293]          17
##  6 (293,351.2]           8
##  7 (351.2,409.4]        10
##  8 (409.4,467.6]         8
##  9 (467.6,525.8]         4
```

```
## 10 (525.8,584.6]         5
```

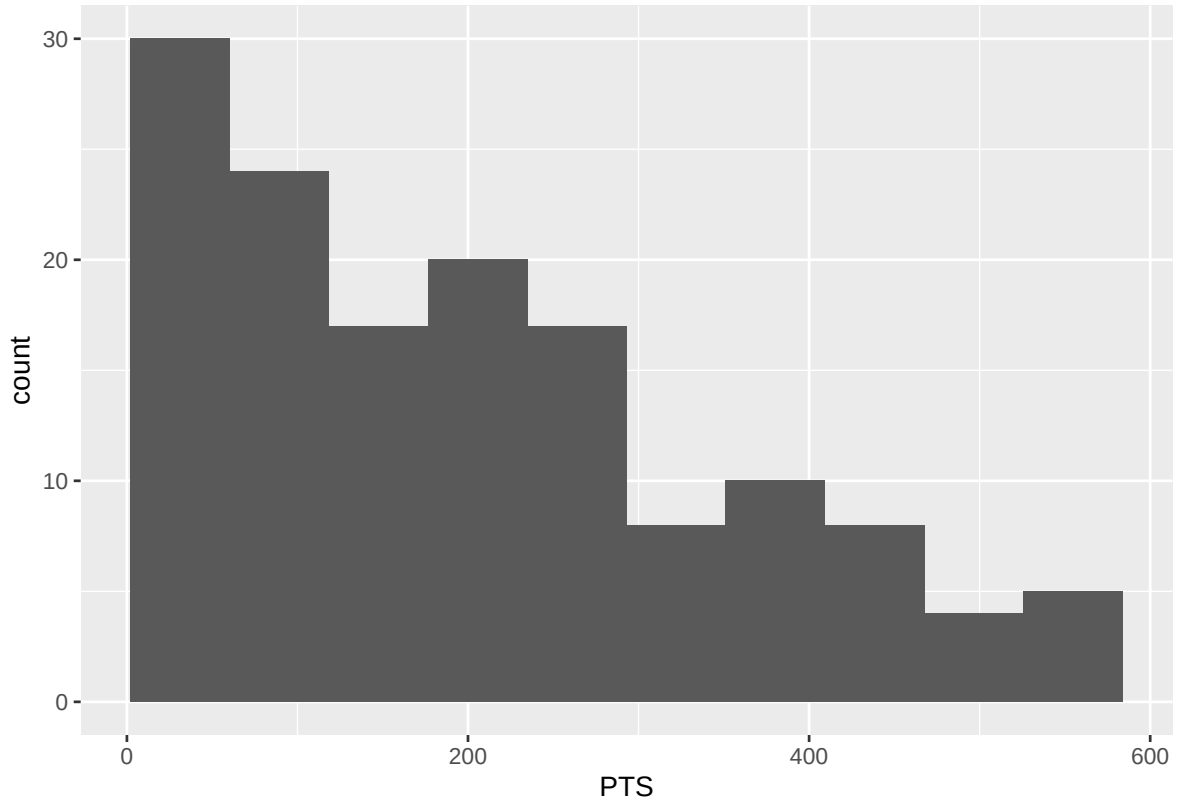
Não, não fica igual, o primeiro grupo na tabela fica com 30 observações, enquanto que a primeira barra do histograma fica abaixo dos 20.

Isso se dá por diferença na lógica de cada algoritmo, os limites mínimos e máximos de cada abordagem são diferentes. Então as observações que na tabela ficam no primeiro grupo são colocados no segundo grupo no histograma.

Para isso podemos usar 2 parâmetros:

- boundary: o menor valor da nossa base é 2 para PTS, mas o histograma começa em menor que zero, então podemos forçar ele respeitar o valor mínimo começando dele.
- binwidth: No histograma o bindwidth mantem o mesmo tamanho para todos grupos, nós queremos 10 grupos, então pegamos a diferença entre o menor e maior valor dividido por 10.

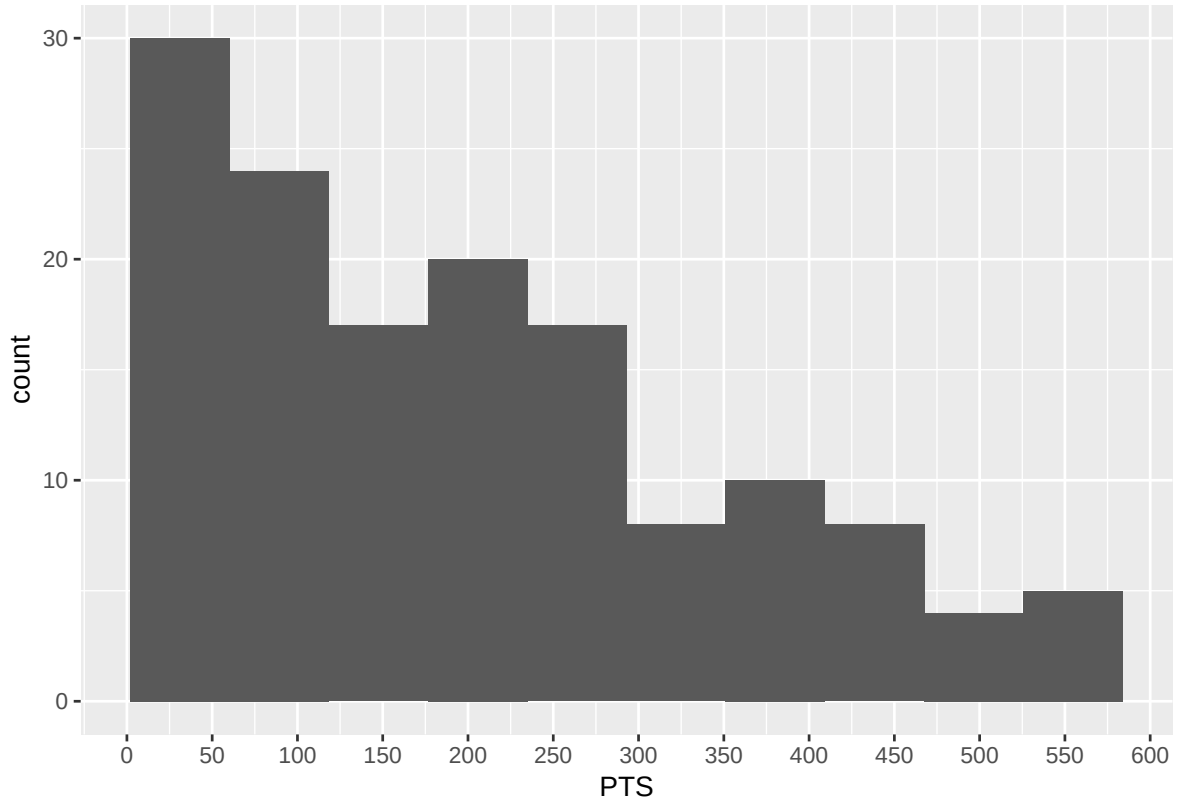
```
wnba %>%
  ggplot(aes(x=PTS)) +
  geom_histogram(boundary = min(wnba$PTS),
                 binwidth = ((max(wnba$PTS) - min(wnba$PTS)) /10) )
```



Podemos melhorar o gráfico acima mexendo nos valores exibidos no eixo x que estão pulando de 200 em 200 números dificultando as análises.

Podemos pular de 50 em 50 por exemplo, e fica mais fácil até de comparar com a tabela frequência ou com a função summary.

```
wnba %>%
  ggplot(aes(x=PTS)) +
  geom_histogram(boundary = min(wnba$PTS),
                 binwidth = ((max(wnba$PTS) - min(wnba$PTS)) /10) )+
  scale_x_continuous(breaks = seq(0,600, by = 50))
```




```
summary(wnba$PTS)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     2.0    75.0   177.0   201.8   277.5   584.0
```

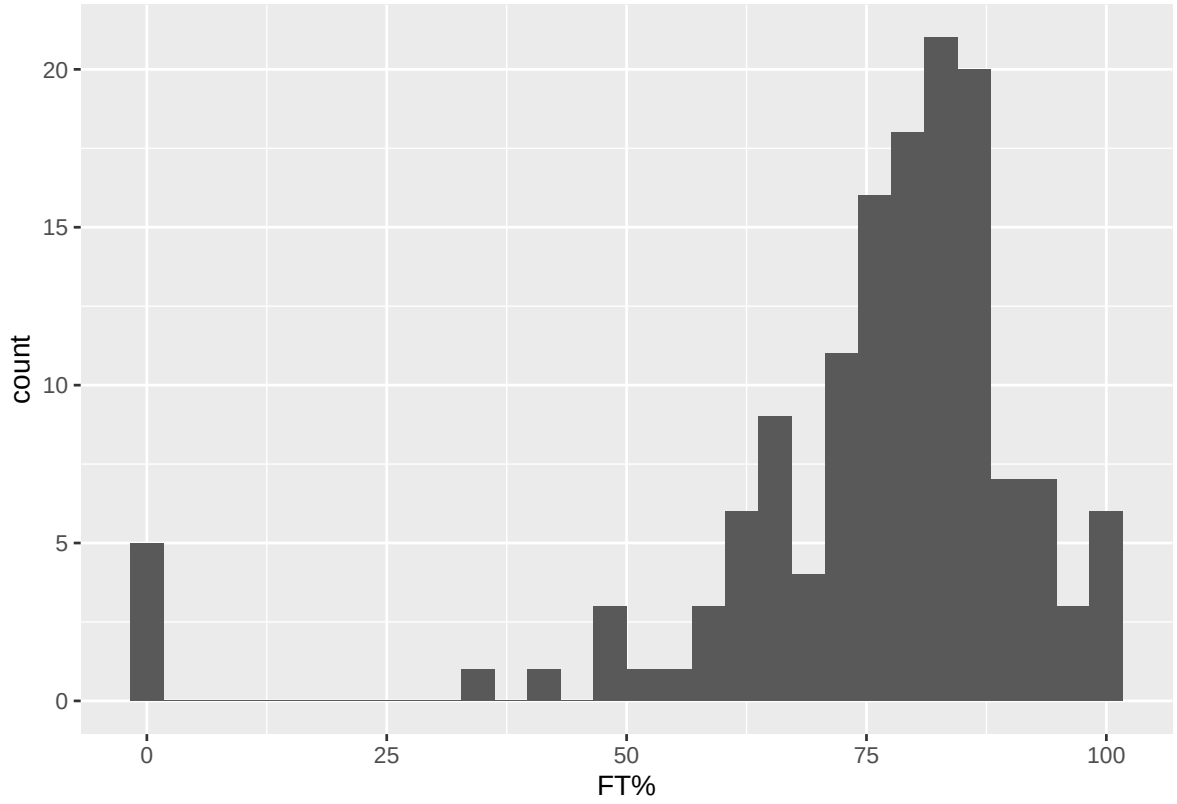
Skewed Distributions (Curva Assimétrica)

São distribuições onde os dados se acumulam numa ponta e vão dimnuindo drasticamente formando uma “cauda” até não haver mais valores no intervalo da ponta oposta.

Quando esse acumulo está na ponta da direita com cauda seguindo para o lado esquerdo, por estar na direção dos números negativos se chama **curva assimétrica negativa**. Caso contrário, se os valores se acumularem do lado esquerdo com uma cauda seguindo para o lado direito, então é uma **curva assimétrica positiva**.

```
wnba %>%
  ggplot(aes(x=`FT%`)) +
  geom_histogram()
```

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

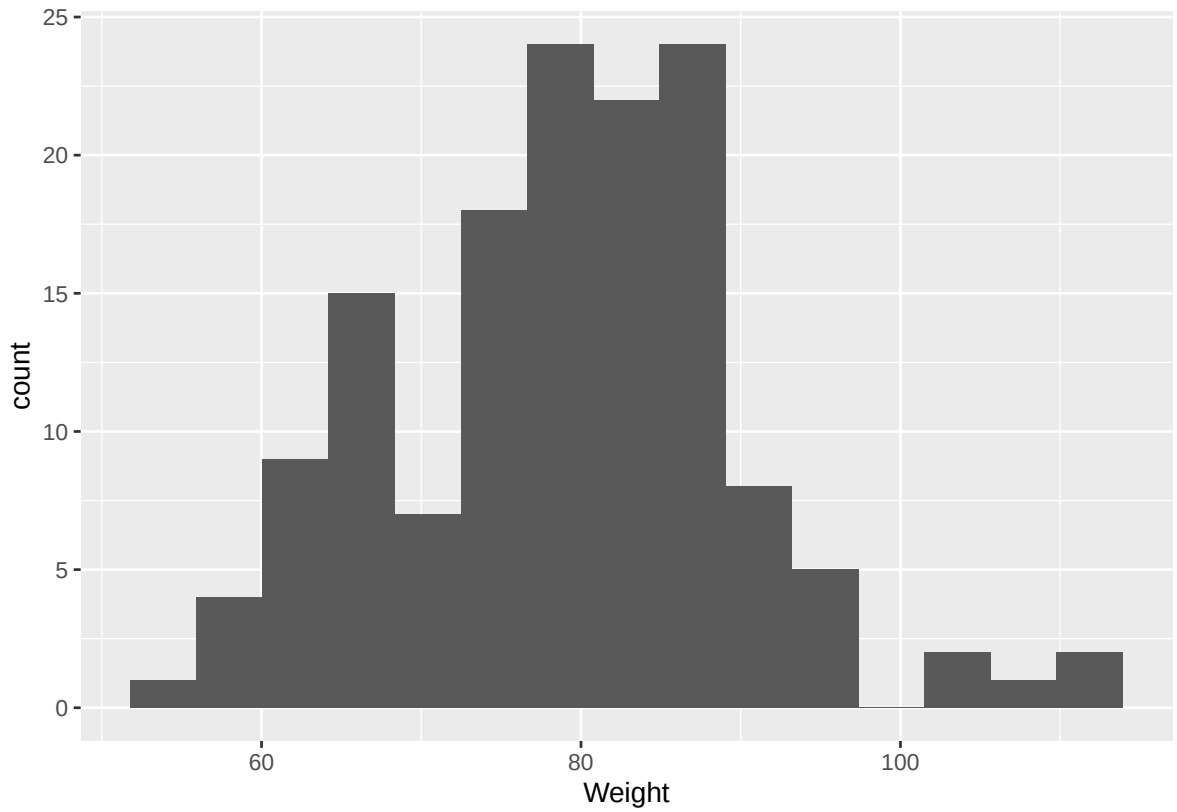


Curvas Simétricas

Quando a curva tem formato simétrico (se ao dividimos no meio e cada lado parece ser o “espelho” do outro), e além disso os dados se acumulam no meio e ambas as pontas vão diminuindo, chamamos de **curva normal**.

```
ggplot(data = wnba,
       aes(x = Weight)) +
  geom_histogram(bins = 15)
```

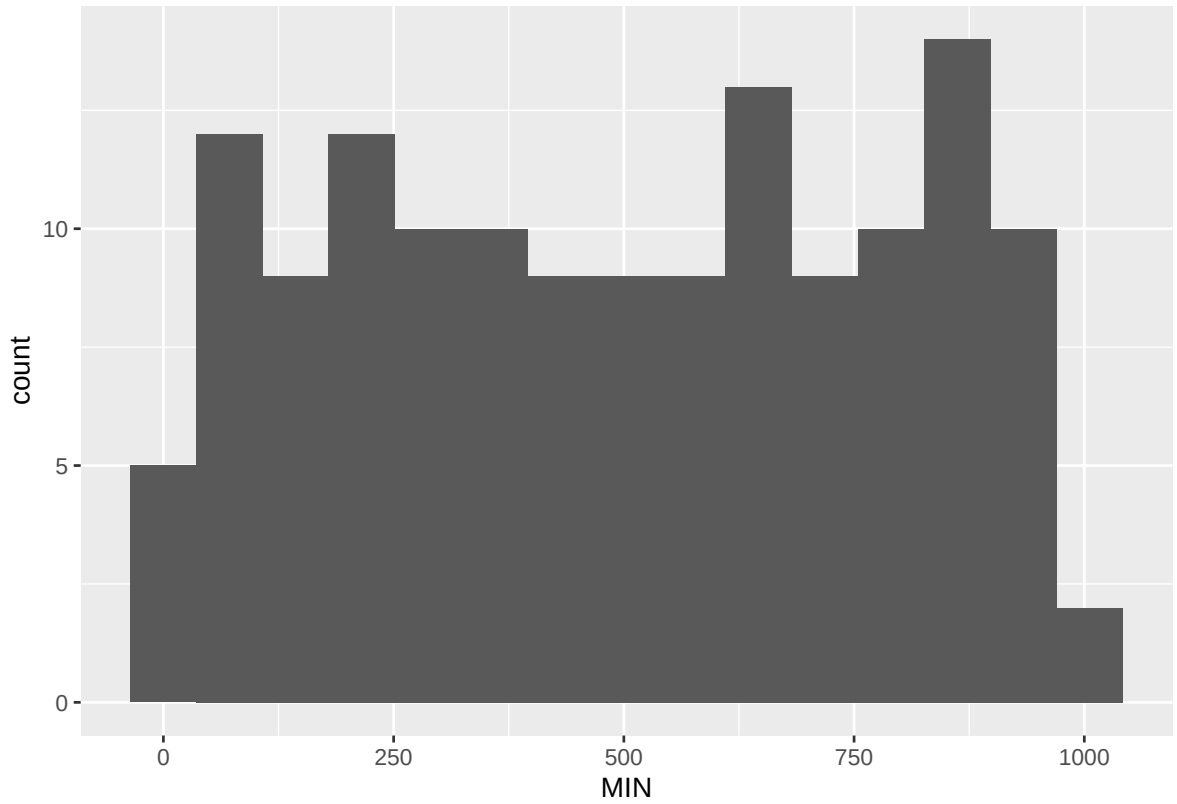
```
## Warning: Removed 1 rows containing non-finite values (`stat_bin()`).
```



Caso os dados se acumulem como uma reta ao invés de curva, ainda de forma simétrica, chamamos de **distribuição uniforme**.

Ainda assim é raro ver curvas perfeitamente simétricas, mas ainda assim usamos esse termo para descrever curvas que se aproximam desse formato.

```
ggplot(data = wnba,
       aes(x = MIN)) +
  geom_histogram(bins = 15)
```


count
10
5
0
0
250
500
750
1000
MIN