Data Visualization with ggplot2 (Single Categorical Variable)

Learning Objectives

1. Summarize a categorical variable using tables
2. Summarize a categorical variable using bar charts
3. Customize title and labels of axes in bar charts
4. Create a bar chart using proportions instead of counts

The `ggplot2` package enables users to create various kinds of data visualizations, beyond the visualizations that can be made in base R. The `ggplot2` package is automatically loaded when we load the `tidyverse` package, although we can load `ggplot2` on its own.

We will use the dataset ClassDataPrevious.csv as a working example. Download the dataset from Collab and read it into R

```
library(tidyverse)
Data<-read.csv("ClassDataPrevious.csv", header=TRUE)
```

1. Summarize a categorical variable using tables

Note: Discrete variables are interesting since they can be used with tools meant for categorical variables, as well as tools meant for quantitative variables.

Frequency tables are a common tool to summarize categorical variables. The `table()` function creates frequency tables. Suppose we want to see the number of students in each year in our data

```
table(Data$Year)
```

```
##
##  First Fourth Second  Third
##     83     30    139     46
```

Notice the order of the years could be rearranged to make more sense

```
Data$Year<-factor(Data$Year, levels=c("First","Second","Third","Fourth"))
levels(Data$Year)
```

```
## [1] "First"  "Second" "Third"  "Fourth"
```

```
mytab<-table(Data$Year)
mytab
```

```
##
##  First Second  Third Fourth
##     83    139     46     30
```

So we have 83 first years, 139 second years, 46 third years, and 30 fourth years in our dataset.

We can easily create a table involving proportions using `prop.table()`

```
prop.table(mytab)
```

```
##
##     First    Second     Third    Fourth
## 0.2785235 0.4664430 0.1543624 0.1006711
```

or percentages

```
prop.table(mytab) * 100
```

```
##
##    First   Second    Third   Fourth
## 27.85235 46.64430 15.43624 10.06711
```

To round the percentages to two decimal places, use the `round()` function

```
round(prop.table(mytab) * 100, 2)
```

```
##
##  First Second  Third Fourth
##  27.85  46.64  15.44  10.07
```

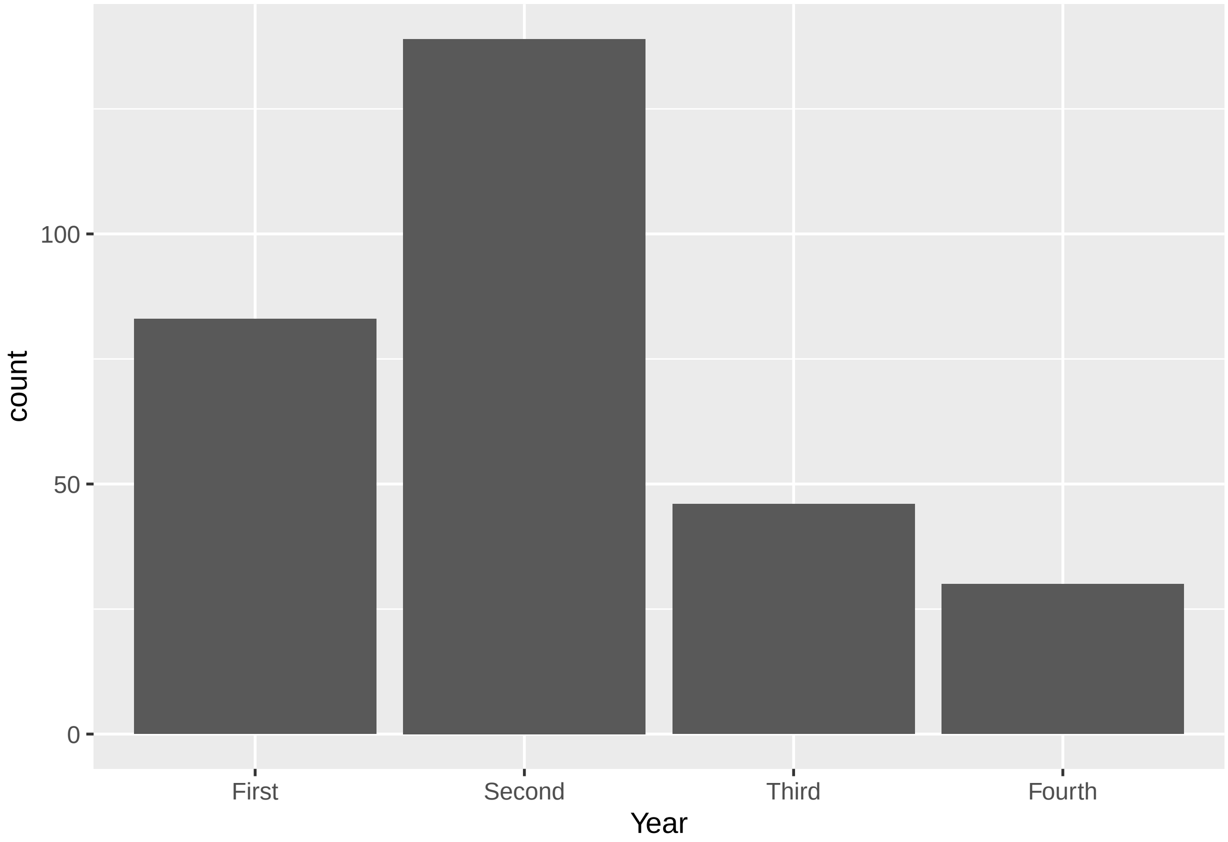
Next, we will create graphical summaries of a categorical variable.

2. Summarize a categorical variable using bar charts

We will be using the `ggplot()` function to create data visualizations. The function comes from the `ggplot2` package which is loaded with the `tidyverse` package.

Bar charts are useful with categorical data. To create a basic bar chart of `Years`

```
ggplot(Data, aes(x=Year))+
  geom_bar()
```



From these two lines we code, we can see the basic structure of creating data visualizations with the `ggplot()` function:

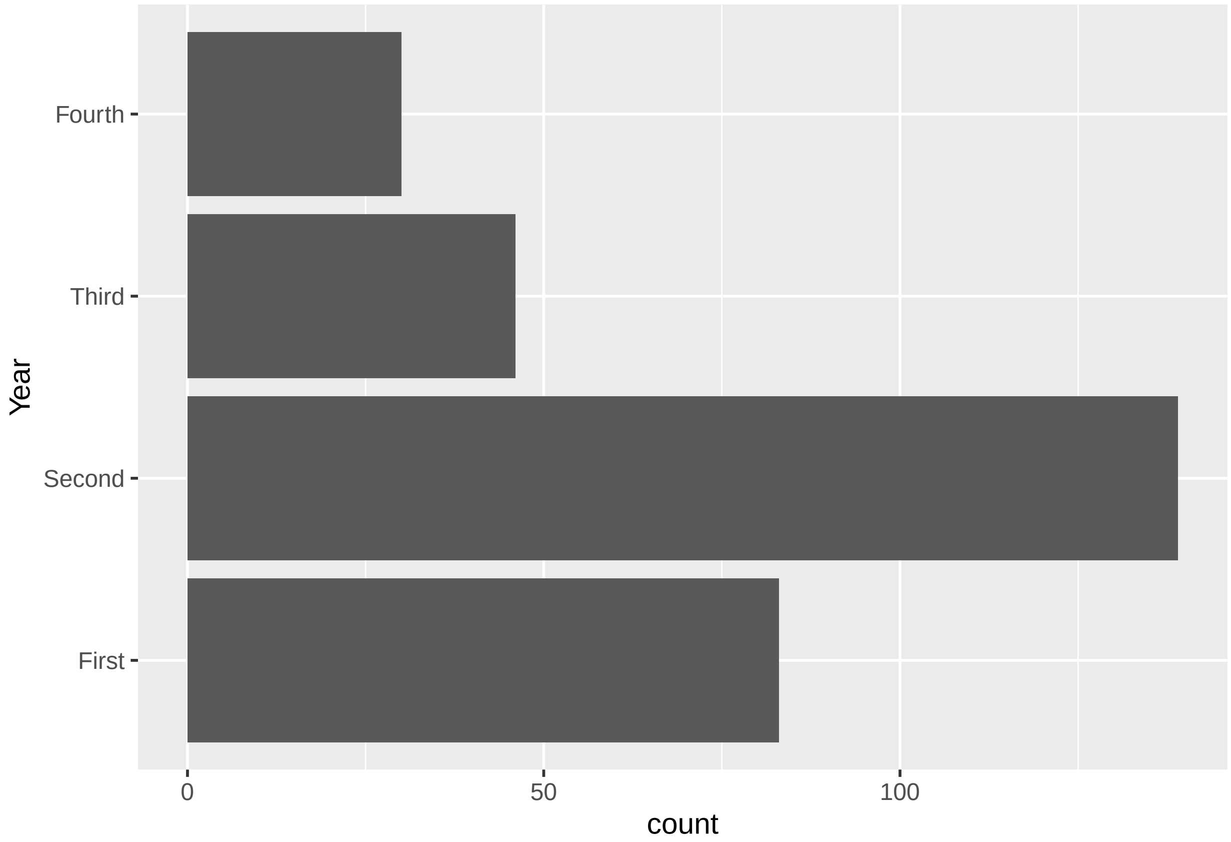
1. Use the `ggplot()` function, and supply the name of the data frame, and the x- and/or y- variables via the aes() function. End this line with a `+` operator, and then press enter.

2. In the next line, specify the type of graph we want to create (called `geoms`). For a bar chart, type `geom_bar()`.

Some describe these lines of code as two layers of code. These two layers must be supplied for all data visualizations with `ggplot()`.

Additional optional layers can be added (these usually deal with the details of the visuals). Suppose we want to change the orientation of this bar chart, we can add an optional line, or layer

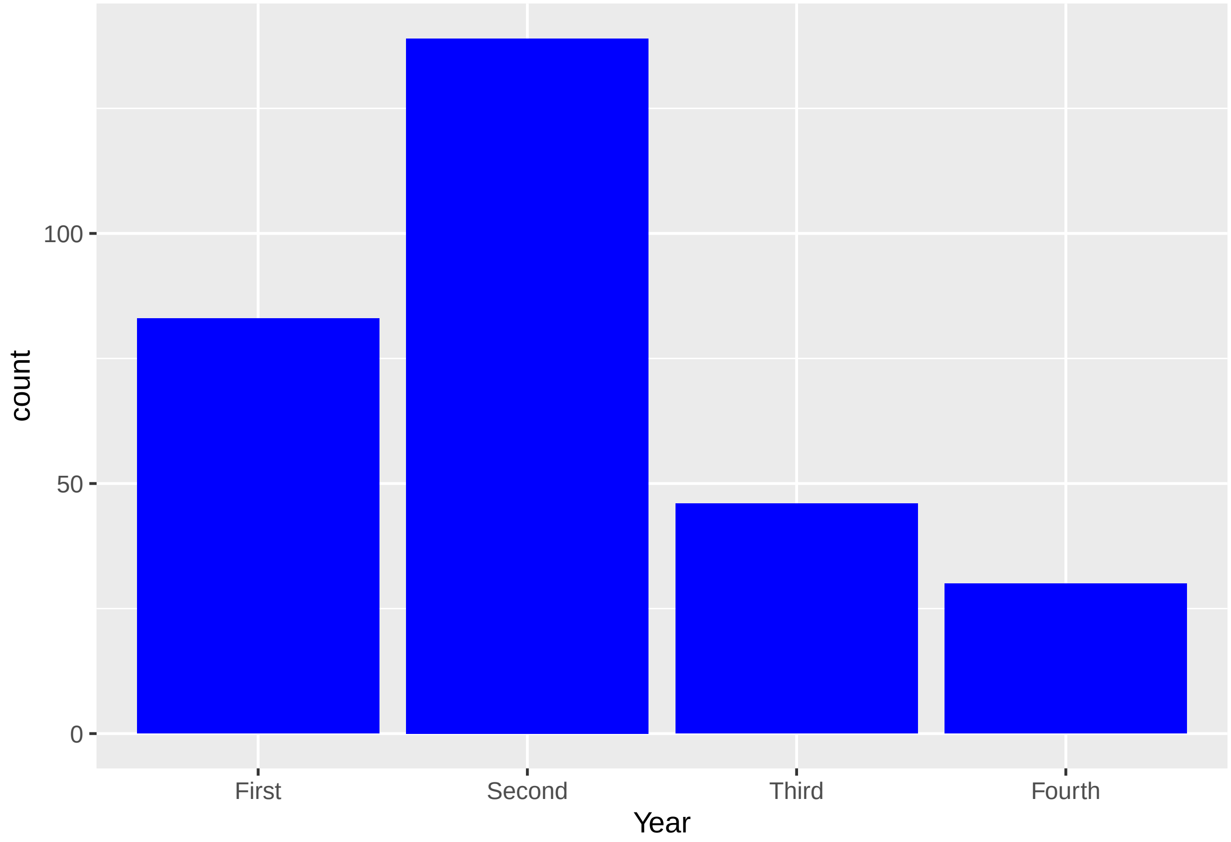
```
ggplot(Data, aes(x=Year))+
  geom_bar()+
  coord_flip()
```



It is recommended that each layer is typed on a line below the previous layer. A + sign is used at the end of each layer to add another layer below.

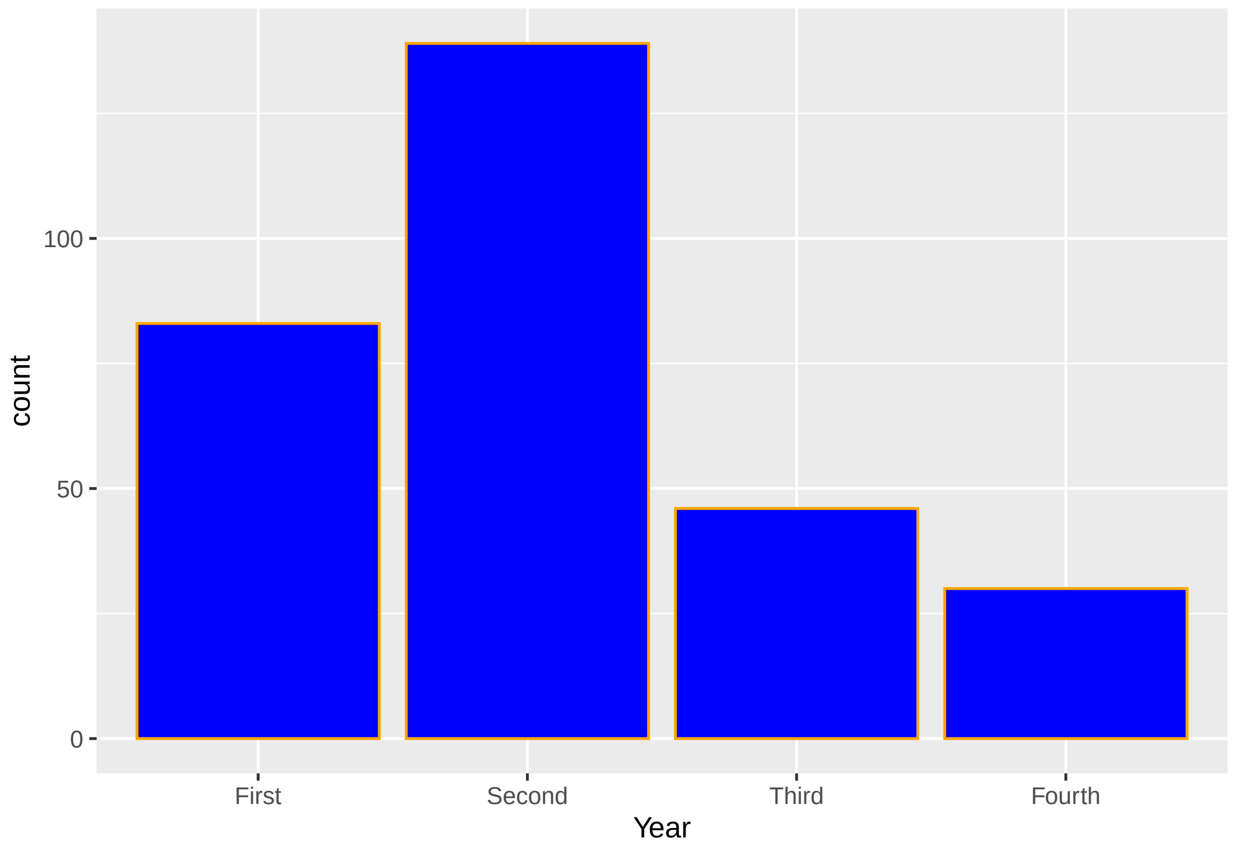
To change the color of the bars

```
ggplot(Data, aes(x=Year))+
  geom_bar(fill="blue")
```



To have a different color to outline the bars

```
ggplot(Data, aes(x=Year))+
  geom_bar(fill="blue",color="orange")
```

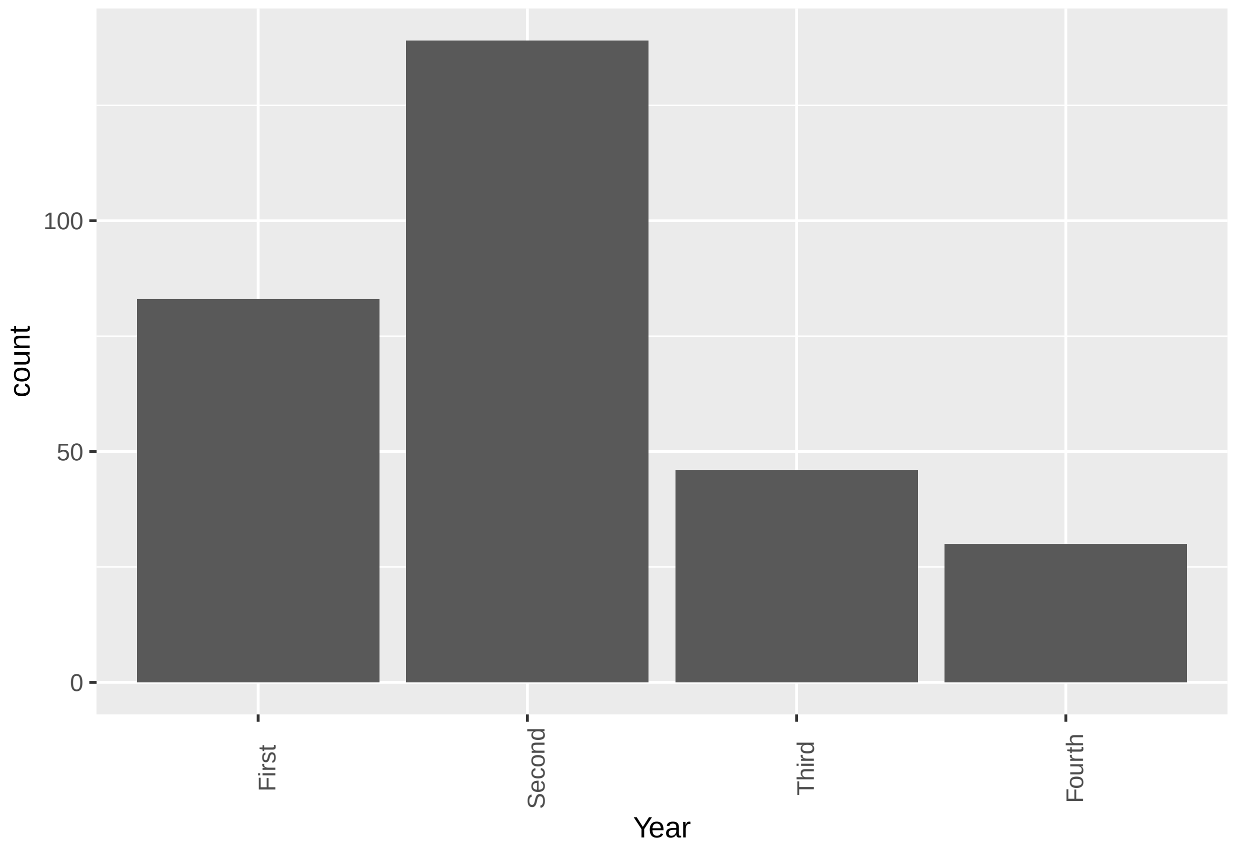


3. Customize title and labels of axes in bar charts

To change the orientation of the labels on the horizontal axis, we add an extra layer called `theme`. This will be useful when we have many classes and/or labels with long names.

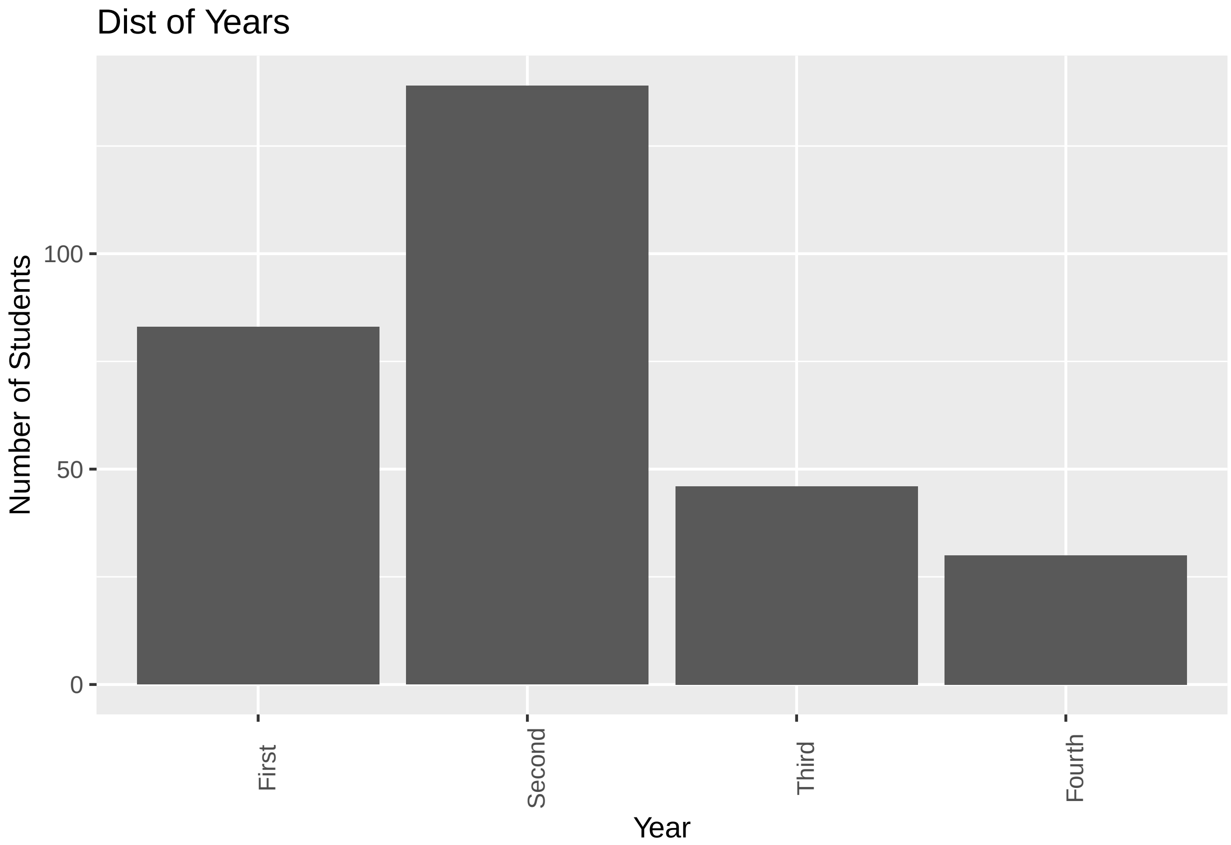
To rotate the labels on the horizontal by 90 degrees

```
ggplot(Data, aes(x=Year))+
  geom_bar()+
  theme(axis.text.x = element_text(angle = 90))
```



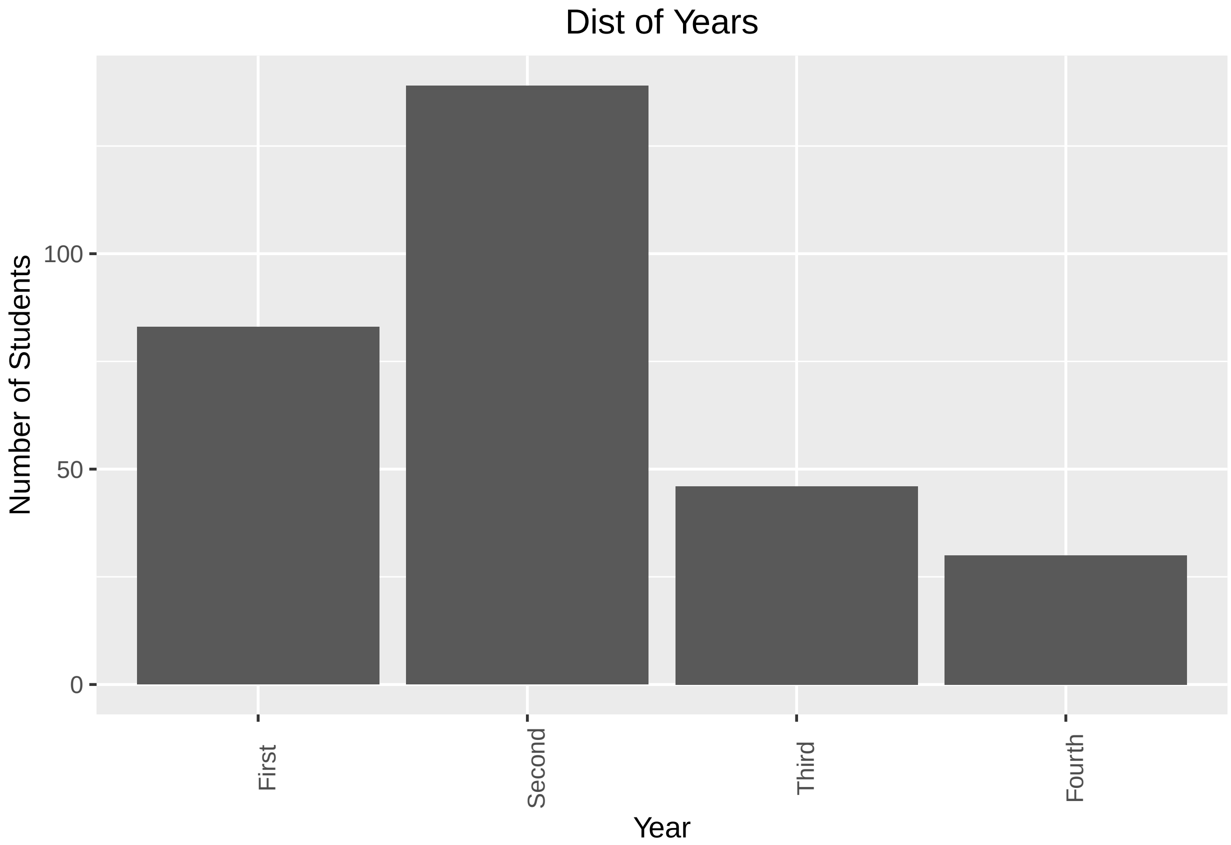
We can also change the labels of the x- and y- axes, as well as add a title for the bar chart by adding another layer called `labs`.

```
ggplot(Data, aes(x=Year))+
  geom_bar()+
  theme(axis.text.x = element_text(angle = 90))+
  labs(x="Year", y="Number of Students", title="Dist of Years")
```



We can also adjust the position of the title, for example, center-justify it via `theme`

```
ggplot(Data, aes(x=Year))+
  geom_bar()+
  theme(axis.text.x = element_text(angle = 90),
        plot.title = element_text(hjust = 0.5))+
  labs(x="Year", y="Number of Students", title="Dist of Years")
```



4. Create a bar chart using proportions instead of counts

Suppose we want to create a bar chart where the y-axis displays the proportions, rather than the counts of each level. First, we create a new data frame, where each row represents a year, and we add the proportion of each year into a new column

```
newData<-Data%>%
  group_by(Year)%>%
  summarize(Counts=n())%>%
  mutate(Percent=Counts/nrow(Data))
```

The code above does the following:

1. Creates a new data frame called `newData` by taking the data frame called `Data`,
2. and then groups the observations by `Year`,
3. and then counts the number of observations in each `Year` and storing these values in a vector called `Counts`,
4. and then creates a new vector called `Percent` by using the mathematical operations as specified in `mutate()`. `Percent` is added to `newData`.

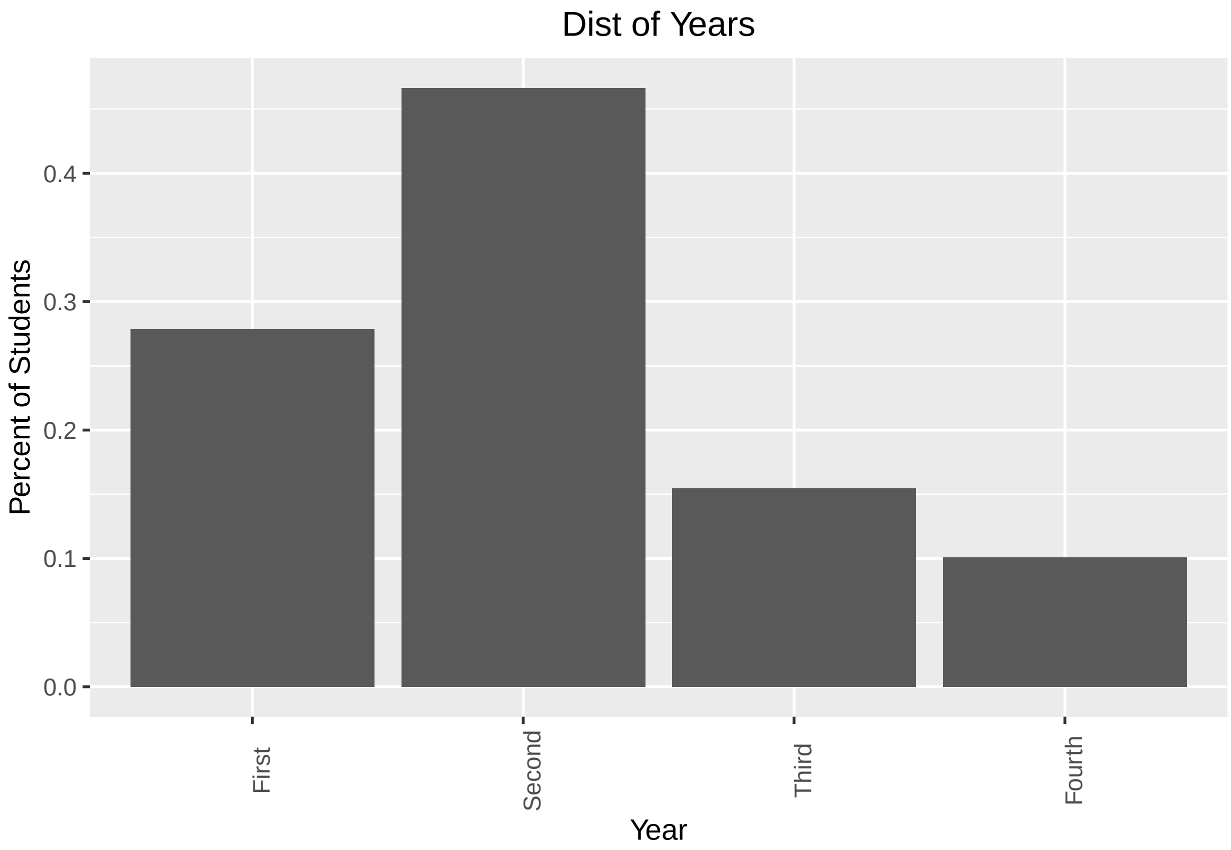
We can take a look at the contents of `newData`

```
newData
```

```
## # A tibble: 4 x 3
##   Year   Counts Percent
##   <fct>   <int>   <dbl>
## 1 First      83   0.279
## 2 Second    139   0.466
## 3 Third      46   0.154
## 4 Fourth     30   0.101
```

To create a bar chart using proportions

```
ggplot(newData, aes(x=Year, y=Percent))+
  geom_bar(stat="identity")+
  theme(axis.text.x = element_text(angle = 90),
        plot.title = element_text(hjust = 0.5))+
  labs(x="Year", y="Percent of Students", title="Dist of Years")
```



Note the following:

1. In the first layer, we use `newData` instead of the old data frame. In `aes()`, we specified a y-variable, which we want to be `Percent`.
2. In the second layer, we specified `stat="identity"` inside `geom_bar()`.