To create a named value *x* and assign it a value of 3.

x <- 3

A **numeric vector** is an ordered list of numbers.

c() to create a numeric vector:

numlist <- c(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)

R also supports a number of more advanced data structures:

* A **matrix** - a vector of vectors, where each value in the matrix is the same data type.
* A **data frame** is very similar to a Pandas DataFrame where each column can be a different data type.
* A **tibble** is a recent data object introduced by the tidyverse package in R and is an optimized data frame with extra metadata and features. The most current libraries and packages in R use data frames or tibbles.

Graphical user interface, text, application

Description automatically generated

* **Function**

function\_name <- function(arg1, arg2=T, …){

<BODY OF FUNCTION>

return <RETURN VALUE>

}

library(package) to import the library.

library(jsonlite) library to read in JSON data structures and convert them to an R data frame.

fromJSON() function to read in a JSON file into R.

R's index starts at 1. So, the third element would be index = 3.

A screenshot of a computer

Description automatically generated with medium confidence

* **To Select**

To select the third row of the Year column using bracket notation:

demo\_table[3,"Year"]

or

demo\_table[3,3]

To select the vector of vehicle classes from demo\_table:

demo\_table$"Vehicle\_Class"

To select single value:

demo\_table$"Vehicle\_Class"[2]

* **To Filter:**

To filter car data demo\_table2 with vehicle price is greater than $10,000:

filter\_table <- demo\_table2[demo\_table2$price > 10000,]

(The comma is necessary to subset by rows. Adding column(s) after the comma specifies the columns to select.)

Graphical user interface, text, application, email

Description automatically generated

Filter a dataset from used car data demo\_table2 where price > 10000, drive == 4wd, and "clean" %in% title\_status:

filter\_table2 <- subset(demo\_table2, price > 10000 & drive == "4wd" & "clean" %in% title\_status)

* **Sample Data**

Graphical user interface, text, application, email

Description automatically generated

1. Capture the number of rows in demo\_table in a variable. The nrow() function counts the number of rows in a dataframe.

num\_rows <- 1:nrow(demo\_table)

2. Sample 3 of those rows:

sample\_rows <- sample(num\_rows, 3)

3. Retrieve the requested data within the demo\_table:

demo\_table[sample\_rows,]

If combine:

demo\_table[sample(1:nrow(demo\_table), 3),]

* **Dplyr**

%>% To chain together functions in a single statement.

library(tidyverse)

# add 2 columns to original data frame.

demo\_table <- demo\_table %>% mutate(Mileage\_per\_Year=Total\_Miles/(2020-Year),IsActive=TRUE)

Graphical user interface

Description automatically generated

* **Groupby**

summarize\_demo <- demo\_table2 %>% group\_by(condition) %>% summarize(Mean\_Mileage=mean(odometer), .groups = 'keep')

Graphical user interface

Description automatically generated

summarize\_demo <- demo\_table2 %>% group\_by(condition) %>% summarize(Mean\_Mileage=mean(odometer),Maximum\_Price=max(price),Num\_Vehicles=n(), .groups = 'keep')

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated

* **Gather()**

**Graphical user interface, text, application, email

Description automatically generated**

pivot\_longer() lengthens the data by increasing the number rows and decreasing the number of columns.

pivot\_wider() will perform an inverse transformation.

long\_table <- gather(demo\_table3,key="Metric",value="Score",buying\_price:popularity)

A picture containing graphical user interface

Description automatically generated

* **Spread()**

**Graphical user interface, text, application, email

Description automatically generated**

wide\_table <- long\_table %>% spread(key="Metric",value="Score")

all.equal() compare two data frames to be equal.

order() and colnames() functions and bracket notation:

sorting the columns of both data frames

> table <-demo\_table3[,order(colnames(wide\_table))]

Or

> table <- demo\_table3[,(colnames(wide\_table))]

(The comma in the bracket indicates that we're selecting all rows.)

* **Ggplot2**

1. **ggplot function**—tells ggplot2 what variables to use
2. **geom function**—tells ggplot2 what plots to generate
3. **formatting or theme functions**—tells ggplot2 how to customize the plot

Graphical user interface, text, application, email

Description automatically generated

* **Bar Plot**

head(mpg) # Dataset: mpg.

plt <- ggplot(mpg,aes(x=class)) # import dataset into ggplot2.

plt + geom\_bar() # plot a bar plot.

* **Compare**

mpg\_summary <- mpg %>% group\_by(manufacturer) %>% summarize(Vehicle\_Count=n(), .groups = 'keep') # create summary table.

plt <- ggplot(mpg\_summary,aes(x=manufacturer,y=Vehicle\_Count)) # import dataset into ggplot2.

plt + geom\_col() + xlab("Manufacturing Company") + ylab("Number of Vehicles in Dataset") #plot a boxplot with labels.

+ theme(axis.text.x=element\_text(angle=45,hjust=1)) #rotate the x-axis label 45 degrees.

* **Line Plot**

To compare the differences in average highway fuel economy (hwy) of Toyota vehicles as a function of the different cylinder sizes (cyl):

mpg\_summary <- subset(mpg,manufacturer=="toyota") %>% group\_by(cyl) %>% summarize(Mean\_Hwy=mean(hwy), .groups = 'keep') #create summary table.

plt <- ggplot(mpg\_summary,aes(x=cyl,y=Mean\_Hwy)) #import dataset.

plt + geom\_line() + scale\_x\_discrete(limits=c(4,6,8)) + scale\_y\_continuous(breaks = c(15:30)) #add line plot with labels.

scale\_x\_discrete() generate x-axis ticks for each value in a list.

scale\_y\_continuous() rescale the y-axis based on a defined range, from 15 through 30 using breaks = c(15:30).

* **Scatter Plot**

plt <- ggplot(mpg,aes(x=displ, y=cty, color=class, shape=drv)) #import dataset into ggplot2

plt + geom\_point() + labs(x="Engine Size (L)", y="City Fuel-Efficiency (MPG)", color="Vehicle Class", shape="Type of Drive") #add scatter plot with multiple aesthetics

Customizing aesthetics add to aes() function to change scatter plot data points:

* **alpha** changes the transparency of each data point
* **color** changes the color of each data point
* **shape** changes the shape of each data point
* **size** changes the size of each data point
* **Box Plot**

plt <- ggplot(mpg,aes(x=manufacturer,y=hwy)) #import dataset into ggplot2

plt + geom\_boxplot() + theme(axis. text.x=element\_text(angle=45,hjust=1)) #add boxplot and rotate x-axis labels 45 degrees

* **Heatmap**

mpg\_summary <- mpg %>% group\_by(model, year) %>% summarize(Mean\_Hwy=mean(hwy), .groups = 'keep') #create summary table

plt <- ggplot(mpg\_summary, aes(x=model, y=factor(year),fill=Mean\_Hwy)) #import dataset into ggplot2

plt + geom\_tile() + labs(x="Model", y="Vehicle Year", fill="Mean Highway (MPG)") #add heatmap with labels

+ theme(axis.text.x = element\_text(angle=90,hjust=1,vjust=.5)) #rotate x-axis labels 90 degrees

* **Add Layer**

plt <- ggplot(mpg,aes(x=manufacturer,y=hwy)) #import dataset into ggplot2

plt + geom\_boxplot() #add boxplot

+ theme(axis.text.x=element\_text(angle=45,hjust=1)) #rotate x-axis labels 45 degrees

+ geom\_point() #overlay scatter plot on top

mpg\_summary <- mpg %>% group\_by(class) %>% summarize(Mean\_Engine=mean(displ),SD\_Engine=sd(displ), .groups = 'keep')

plt <- ggplot(mpg\_summary,aes(x=class,y=Mean\_Engine)) #import dataset into ggplot2

plt + geom\_point(size=4) + labs(x="Vehicle Class",y="Mean Engine Size") #add scatter plot with labels

+ geom\_errorbar(aes(ymin=Mean\_Engine-SD\_Engine,ymax=Mean\_Engine+SD\_Engine)) #overlay with error bars

plt <-ggplot(mpg\_long,aes(x=manufacturer, y=Rating, color=MPG\_Type)) #import dataset into ggplot2

plt + geom\_boxplot() + facet\_wrap(vars(MPG\_Type)) #create multiple boxplots, one for each MPG type

+ theme(axis.text.x=element\_text(angle=45,hjust=1), legend.position = "none") + xlab("Manufacturer") #rotate x-axis labels

**Chart, scatter chart

Description automatically generated**