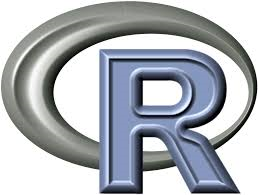
**Introduction to R Labs**



# Configuration Instructions

**Requirements:**

These labs were developed with Windows 10 but most of the content is platform independent.

**Setup and Preparation:**

**Part 1: Getting Started**

**Installing the R Programming Language**

|  |  |
| --- | --- |
| **Step-by-Step Instructions** | |
| **Click Steps** | **Screen Shots** |
| 1. Go to the Comprehensive R Archive Network to download R. Link is <https://cran.r-project.org/> |  |
| 1. Click on the link relevant to your operating system. In this example, we click on Windows. |  |
| 1. Then click on base to download the base R language install program. |  |
| 1. Double click on the downloaded program. If you get a prompt asking if you want to allow this app from an unknown publisher make changes to your computer, click Yes. Note: You need to have permissions to install software on your machine. |  |
| 1. Accept the all the defaults. The application should install successfully. |  |

**Using the R Console**

|  |  |
| --- | --- |
| **Step-by-Step Instructions** | |
|  |  |
| **Click Steps** | **Screen Shots** |
| 1. Start the R Console. In Figure 1, there are multiple versions of R installed. | Figure 1 |
| 1. You should see the screen in Figure 2. | Figure 2 |
| 1. Type in the text show in Figure 3 and press enter. The first line created a variable named a and assigned it a value of 5 times 2 which is 10. The second line causes the variable to display which, i.e. [1] 10. The [1] reminds us that all variables are arrays.   This confirms the R language is installed. | > a <- 5 \* 2  > a  [1] 10  > |

**Installing RStudio**

|  |  |
| --- | --- |
| **Step-by-Step Instructions** | |
|  |  |
| **Click Steps** | **Screen Shots** |
| 1. Go to the link <https://www.rstudio.com/products/rstudio/download/#download>   You see the screen in Figure 1. | Figure 1 |
| 3. Click on the link under Installers for Supported Platforms that applies to your operating system. This will download the RStudio installation program to your computer. | Figure 2 |
| 4. Locate the downloaded RStudio install program and execute it. If you get a prompt asking if you want to allow this app from an unknown publisher make changes to your computer, click Yes. Note: You need to have permissions to install software on your machine. | Figure 3 |
| 1. Accept all the defaults and RStudio should install successfully. |  |

# Using R Studio

|  |  |
| --- | --- |
| **Step-by-Step Instructions** | |
|  |  |
| **Click Steps** | **Screen Shots** |
| 1. Open the RStudio. | Figure 1 |
| 1. You should see the RStudio main screen as shown in Figure 2. | Figure 2 |
| 3. Create a new script file by clicking on the + sign in the upper right corner of the screen and selecting ‘R Script’. | Figure 3 |
| 1. In the script tab, enter the lines shown in Figure 4. |  |
| 1. You should see the program output appear in the console window. |  |

# Part 1 Continued: R Packages

Packages are function libraries that extend R with new functionality. In this lab, we will be installing a new package, loading it, and trying out some of the functionality it has to offer.

.

**Module Requirements**

1. Install the R Package ggplot2.
2. Load the package using the library function.
3. Use the package.

## Hints

1. Import this data set directly from the web

|  |  |
| --- | --- |
| **Step-by-Step Instructions** | |
|  |  |
| **Click Steps** | **Screen Shots** |
| 1. From within RStudio, bring up the Install Packages window. | Figure 1 |
| 1. Start typing the package name, i.e. ggplot2 and the selection list should locate it for you. | Figure 2 |
| 1. Select ggplot2 and click on Install. You should see messages displayed to the console similar to the ones in Figure 3. | Figure 3 |
| 1. Create a new R script by selecting New File from the File menu, and then selecting R Script. | Figure 4 |
| 1. Enter the script as shown in Figure 5. | data('mtcars')  # create factors with value labels  mtcars$gear <- factor(mtcars$gear,levels=c(3,4,5),  library(ggplot2)  labels=c("3gears","4gears","5gears"))  qplot(mpg, data=mtcars,  geom="density",  fill=gear, alpha=I(.5),  main="Distribution of Gas Milage",  xlab="Miles Per Gallon",  ylab="Density") |
|  |  |

# 4 Lab: Vectors

All variables in R are stored as arrays called vectors. In this lab, we’ll create and manipulate some vectors to see how they work.

.

## Module Requirements

1. Start RStudio and create a new script.
2. Create a vector of random numbers following a normal distribution. Hint: Use the rnorm function to generate a list of random normally distributed numbers. rnorm(10) returns 10 numbers on a normal distribution.
3. Display elements 1 and 3 of the vector. Hint: Use the c() function to pass multiple vector indexes, i.e. c(1,3)
4. Display the data type of the vector. Hint: The mode function will do this.
5. Create a second vector of ten random numbers in a uniform distribution and compare the values to the first vector. Hint: Use the runif function to generate a list of random normally distributed numbers. runif(10) returns 10 numbers on a normal distribution.
6. Extract a subset of the first vector using a conditional expression as a subscript. Example: x[x > 1]
7. Create a new vector that combines the two vectors you have created. Note: The c() function will combine vectors.

## Hints

## Solution

Note: Words are a # character are comments and are ignored. You do not need to code them.

# R Code

x <- rnorm(10) # Creates vector x containing 10 number in normal distribution

mode(x) # Displays the data type

x[c(1,3)] # Select items 1 and 3 from x

y <- runif(10) # Random numbers uniform distribution

x > y # Which elements in vector x are > y, return TRUE/FALSE

x[y > .7] # Return elements in x where the same element subscript in y is > .7

z <- c(x,y) # the c function combines vectors

# Lab: Matrices

A matrix is just a two-dimensional array. In this lab, we’ll create and manipulate some matrices to how they work.

.

## Module Requirements

1. Start RStudio and create a new script.
2. Create a matrix variable named m1 with 4 rows and 5 columns consisting of sequential numbers from 1 to 20.
3. Create a second matrix variable called m2 consisting of sequential numbers starting at 20 and decreasing to the value 1. Note: The : character generates integers in a range such as 1:5.
4. Compare the values in m1 to m2 where m1 is greater than m2 so you get a TRUE when the value is greater and FALSE when it is not.
5. Now display the element values in m1 where it is greater than the same element in m2. The comparison is the same what you did in step 4 but now you are displaying the element value instead of TRUE and FALSE.
6. Display row 3, column 1 of m1. Note: Matrix subscripts are accessed in row, column order so m1[2,3] means row 2, column 3.
7. Display all columns of row 3 in m1. Note: If you include the comma between subscripts but do not supply a subscript value, all elements are returned, i.e. m1[,4] returns column 4 for all rows.
8. Display rows 2 through 3 of m1. Note: You can use the : operator to generate a vector as a subscript list.
9. Remove the value in m1 row 3, column 4 by assigning it a value of NA.
10. Display the matrix m1 to confirm the NA value was assigned.

## Hints

## Solution

Note: Words are a # character are comments and are ignored. You do not need to code them.

m1 <- matrix(1:20, nrow=4)

m2 <- matrix(20:1, nrow=4)

m1 > m2

m1[m1 > m2]

m1[3,1] # Row 3, Column 1

m1[3,] # Row 3

m1[2:3,] # Rows 2 through 3

m1[3,4] <- NA

m1

# Lab: Arrays

Arrays have multiple dimensions. In this lab, we’ll create and manipulate an array to how they work.

.

## Module Requirements

1. Start RStudio and create a new script.
2. Create an array variable named a1 with the dimensions 4,5,2 using sequential numbers from 1:40
3. Display the elements in a1 that ae greater than 10.
4. Display the data type of a1. Note: the mode() function displays the data type of a vector.

## Hints

## Solution

Note: Words are a # character are comments and are ignored. You do not need to code them.

a1 <- array(1:40,dim = c(4,5,2))

a1[a1 > 10]

mode(a1)

# Lab: Lists

A list is a collection of objects. Unlike vectors, the objects a list can contains can be of any type.

.

## Module Requirements

1. Start RStudio and create a new script.
2. Create a list variable named l1 with the elements ‘Tom’, ‘Sue’, ‘Joe’, ‘Sally’. Note: the list() function will create a list of a vector passed into it.
3. Display the elements in l1.
4. Display the object class of l1. Note: The class() function will do this.
5. Display element 2 from l1.

## Hints

## Solution

Note: Words are a # character are comments and are ignored. You do not need to code them.

l1 <- list('Tom', 'Sue', 'Joe', 'Sally')

l1

class(l1)

l1[2]

# Lab: Data Frames

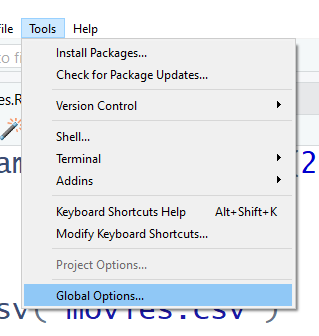
# 

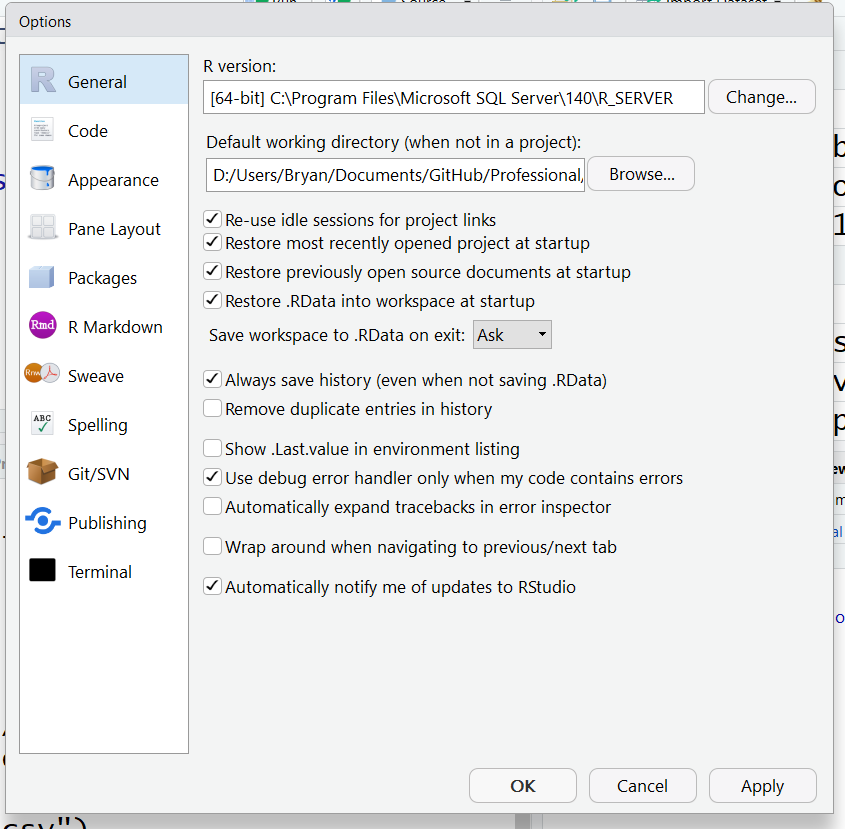
A data frame is an in-memory table that can be manipulated. This is the backbone of data analysis in R.

.

## Module Requirements

1. Start RStudio and create a new script.
2. We want to load a CSV file called movies into a data frame. We can do this using the RStudio import feature or via code.
3. We need to set our working directory which we can do via RStudio’s tools menu as shown below. Set it to \R\_DataScience\_AtoZ\Script\Lab\ .





Set working folder

1. Write R code to load the movies.csv file into an R variable which is in your labs folder.
2. Use the view button on the Environment tab to view the data frame.
3. Get the description of the data frame using the str() function.
4. Use the summary() function on the data frame.
5. Get the average (mean) critics\_score.
6. Display just the movie titles.
7. List the first few rows of data using the head() function.
8. Display a histogram of the movie dvd\_rel\_year column using the hist() function.
9. List only movies where the genre is “Comedy”.
10. Add a new column to movies as movies$runtime/60 to get the run time in hours.

## Hints

* When filtering or sorting data frames, be sure to include the comma after the filter or sort column name respectively. Otherwise you will get an error.

## Solution

Note: Words after a # character are comments and are ignored. You do not need to code them. The solution shows how to programmatically set the working folder to the location of the script that is running.

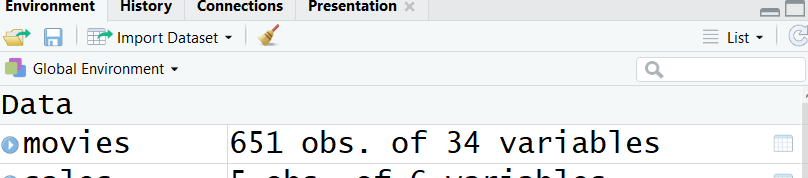
this.dir <- dirname(parent.frame(2)$ofile)

setwd(this.dir)

getwd()

movies <- read.csv("movies.csv")

To view the data frame…



View

Set working folder

Set working folder

str(movies)

summary(movies)

# average score...

mean(movies$critics\_score)

movies$title

head(movies)

hist(movies$dvd\_rel\_year)

movies[movies$genre == "Comedy",]

# Add a calculated column...

movies$runhours <- movies$runtime / 60

movies$runhours

# remove column…

movies$runhours <- NULL

# Sort…

movies[order(movies$title),]

Extra Credit Lab – Power BI with R

# Lab: Power BI with R

# 

This lab is to download and install Power BI desktop and use R scripts to augment what Power BI can do.

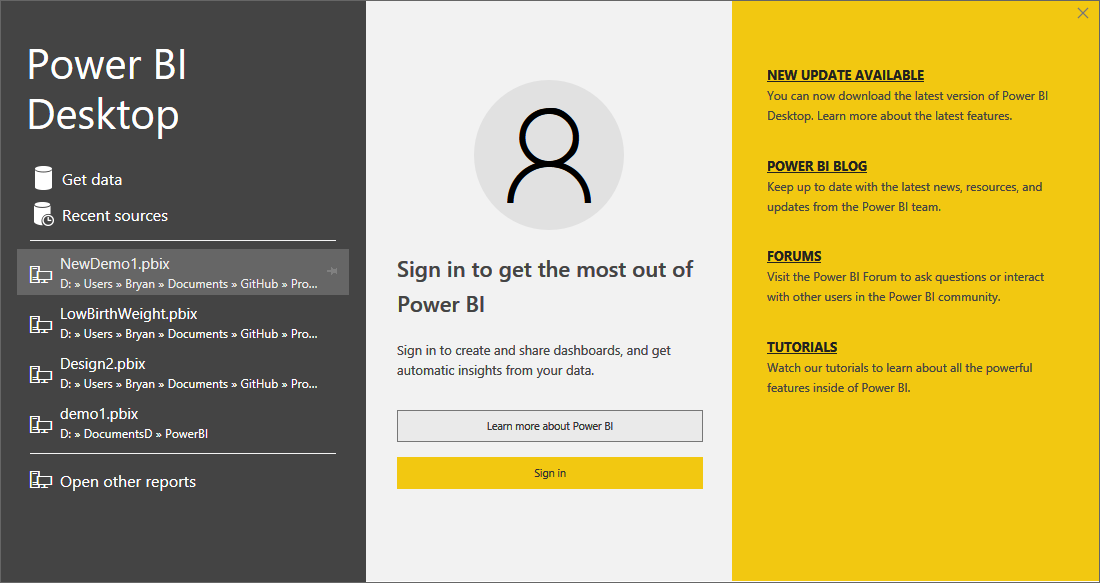
.

## Module Requirements

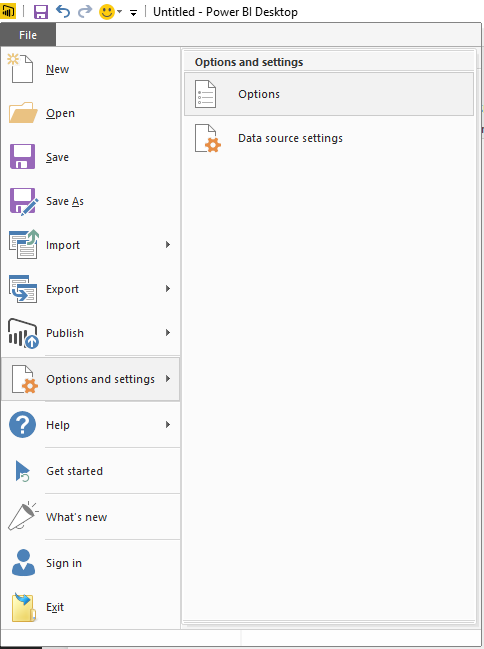
1. Install Power BI Desktop using the link below.

<https://powerbi.microsoft.com/en-us/desktop/>

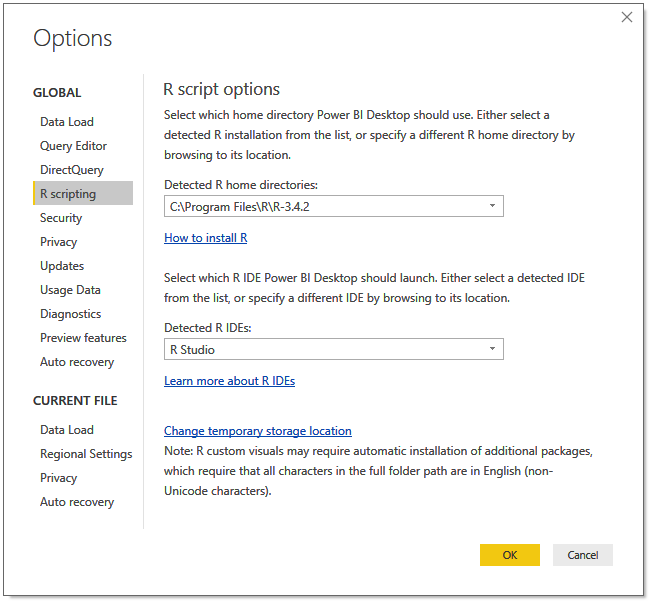
1. Run the downloaded install program and accept the defaults.
2. Find the Power BI program and start it. When the dialog box pops up, just close it.



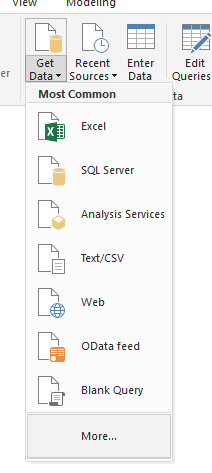
1. Select the File menu and then Options and Settings, then Options.



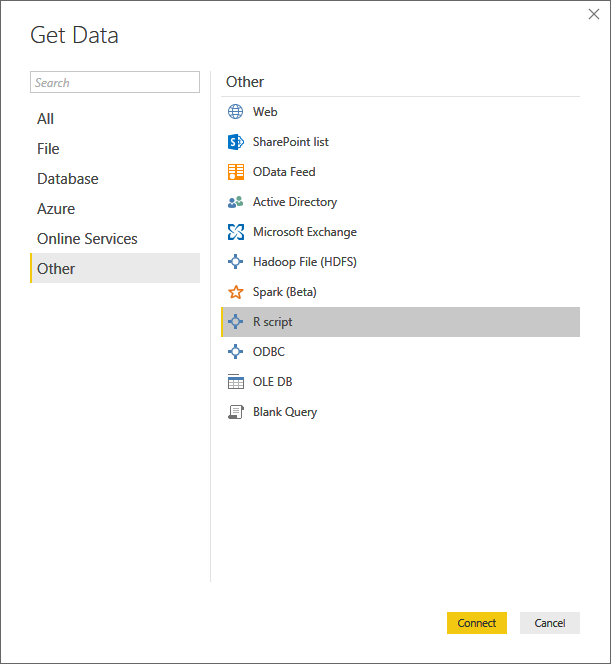
Then use the drop down to select the R installation to use and the R IDE. Then click Ok.



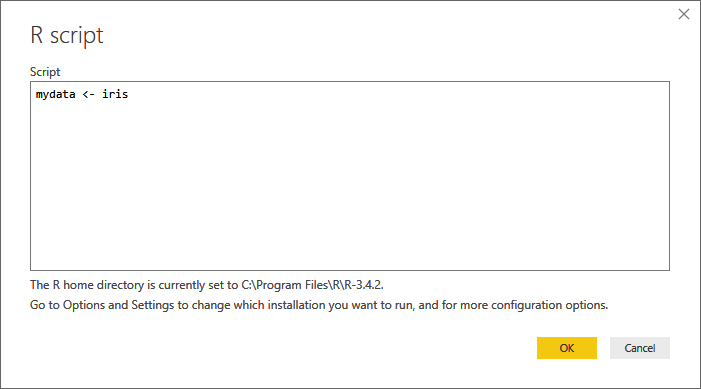
1. Click on Get Data. Then click on More…



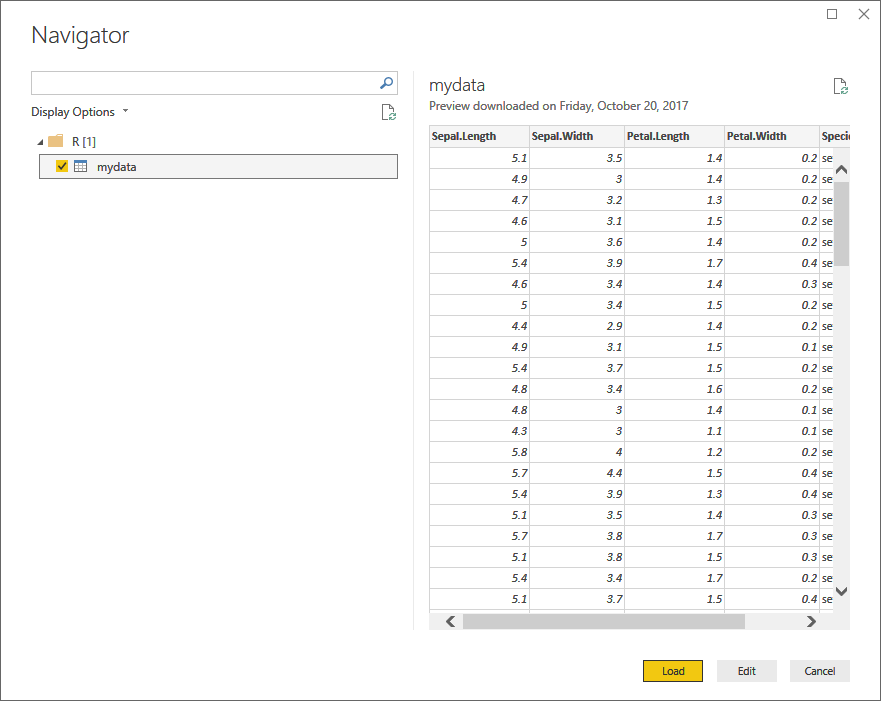
1. Select Other and R Script. Then click on Connect.



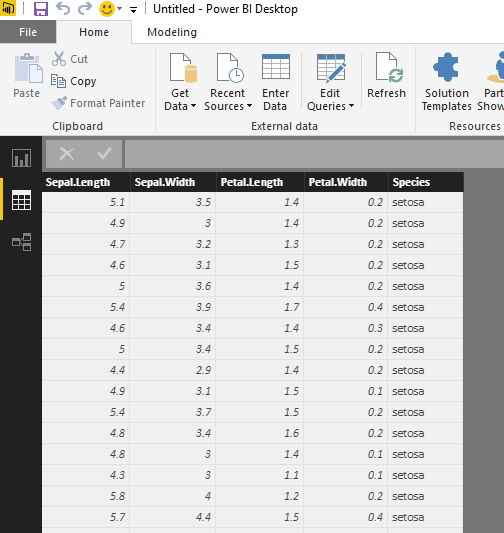
1. Enter the script into the R Script windows as shown below.



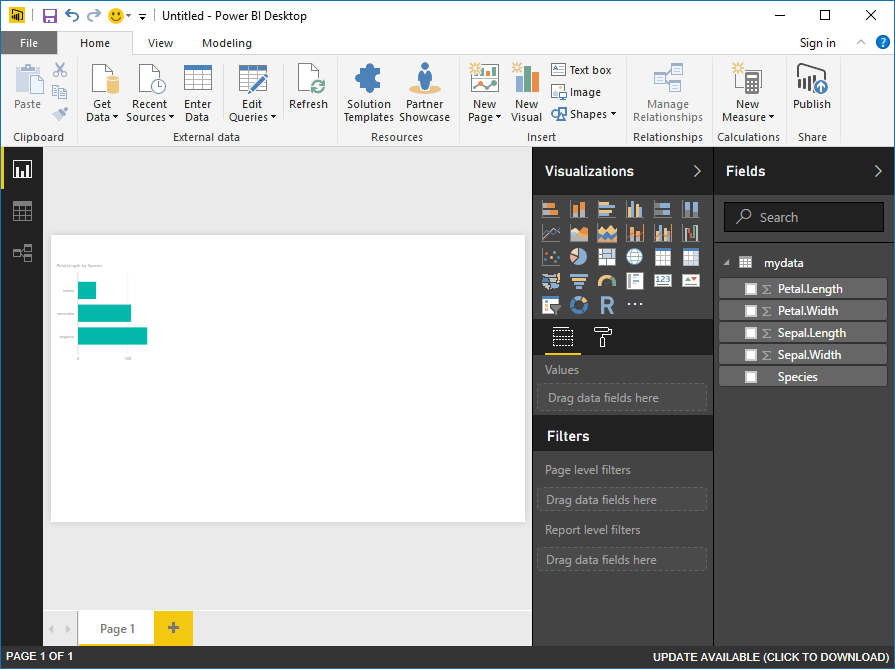
1. Select the checkbox and click on Load.



1. Wait while the data is being loaded.
2. Then click on the middle icon on the left menu bar that looks like a grid and you should see the data.



1. Click on the top side bar menu icon that looks like a bar chart. Then click on the third icon  in the top row under visualizations. Then just check off Species and Petal.Length under the Fields list of the data set. You should see the visualization below.



For more fun exploring, see the slides from the Data Science workshop to do transformations and visualizations in Power BI using R scripts.