**QUESTION:**

Load the in-build dataset from R and draw various basic plot in R using grid (Horizontal bar plot, Vertical bar plot, box plot, multiple box plot, plot with point an lines etc.,)

**CODE:**

library(datasets)

head("mtcars")

summary("mtcars")

boxplot(mpg~gear, data = mtcars, col = "purple")

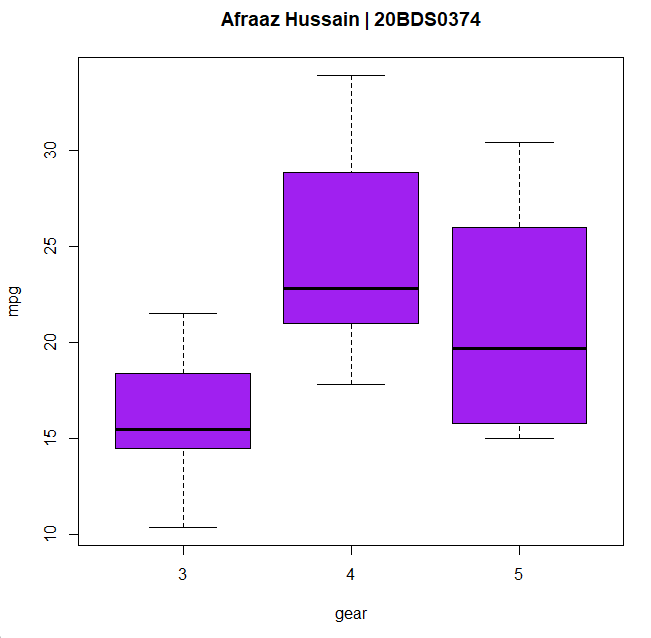
hist(mtcars$mpg, col = "purple", breaks = 50)

barplot(table(mtcars$carb), col = "purple")

with(mtcars, plot(mpg, qsec))

**OUTPUT:**

* Boxplot:



* Bar plot:

Chart, histogram

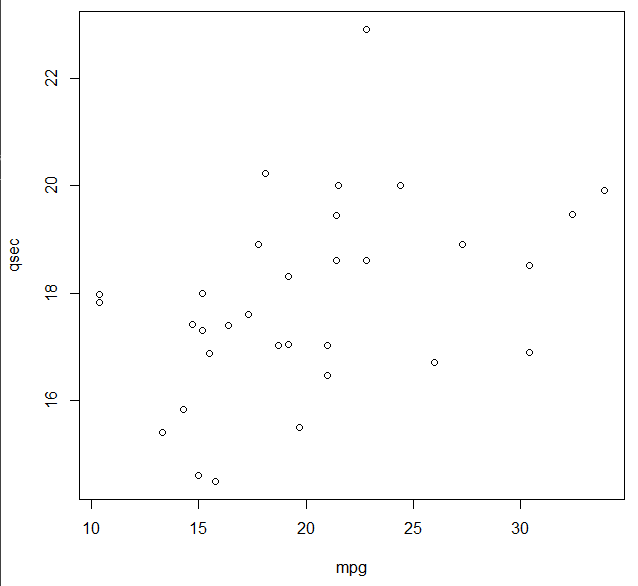
Description automatically generated

* Histogram:

Chart, histogram

Description automatically generated

* Scatter plot:



**QUESTION:**

Load in-build dataset mtcars and visualize data using visualization library ggplot.

**CODE:**

library(tidyverse)

ggplot(data = mpg) + geom\_point(mapping = aes(x = displ, y = hwy, color = class))

**OUTPUT:**

Chart, scatter chart

Description automatically generated

**QUESTION:**

Load the gapminder dataset and perform statistical analysis using tidyverse and dplyr libraries.

**CODE:**

library(tidyverse)

library(dplyr)

library(gapminder)

gapminder%>% filter(year == 1952)%>% arrange(desc(gdpPercap))

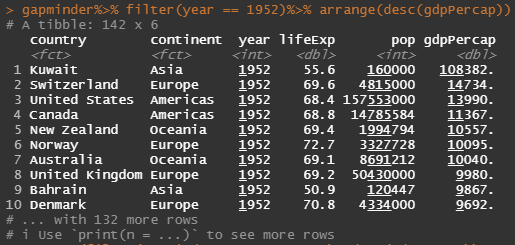
arrange(filter(gapminder, year == 1952), desc(gdpPercap))

gapminder %>% filter(year == 1992, continent == 'Europe')%>% arrange(desc(pop))

gapminder %>% mutate(pop = pop / 1000000)

**OUTPUT:**

* The year is 1952, and the data is arranged in descending order:



* Using filter, arrange the data with year as 1952 in descending order:

Graphical user interface, text

Description automatically generated

* Year is 1992 and continent is Europe:

Graphical user interface, text

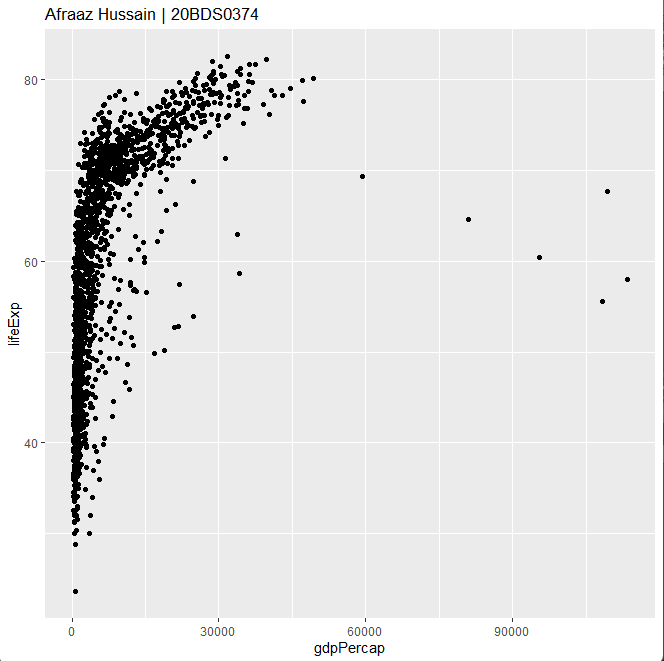
Description automatically generated

* Mutate the population:

A picture containing table

Description automatically generated

* Plot the dataset using ggplot:



**QUESTION:**

Using RClolorBrewer visualize mpg data.

**CODE:**

#Color visualisation using 'RColorBrewer' library

library(RColorBrewer)

library(ggplot2)

library(viridis)

library(datasets)

#The 'View' function is used for viewing a dataset.

#We will now do color visualisation on 'mpg' dataset

View(mpg)

#Here is a density plot on the city

#Use the 'str' function to view the dataset in a structured form

ggplot(data = mpg, aes(x = cty)) + geom\_density()

str(mpg)

#Use the 'factor' argument for factoring in different types in a column

#Use the 'alpha' argument for transparency. This argument is not one of 'factor', but outside it

#The 'labs' function is used for assigning labels

#The 'scale\_fill\_brewer' function is for switching the color palettes of the density graph

p1 = p2 = p3 = p4 = ggplot(data = mpg, aes(x = cty)) +

geom\_density(aes(fill = factor(mpg$cyl), alpha = 0.5)) +

labs(title = "Afraaz Hussain | 20BDS0374", x = "City mileage", y = "Density", fill = "Cylinder", alpha = "Transparency") +

scale\_fill\_brewer(palette = "Purples")

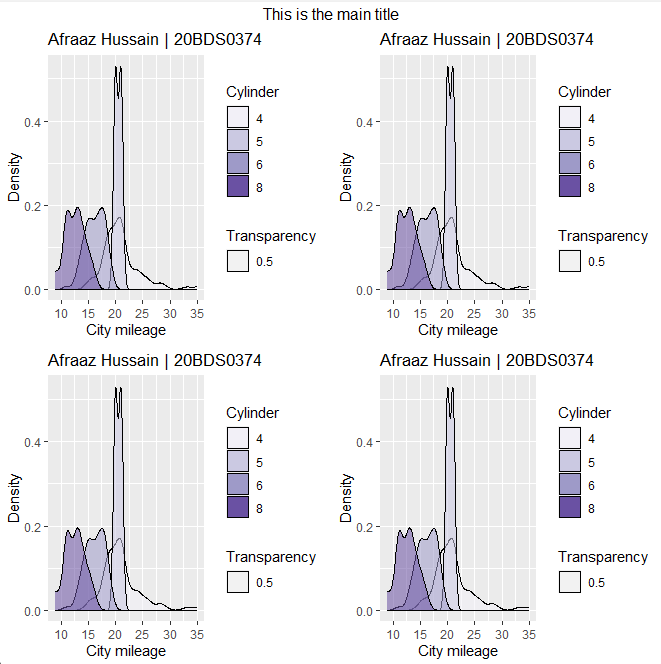
#This library is for a grid like structure

library(gridExtra)

#Now, you can show multiple plots in a single image. Here's how...

grid.arrange(p1, p2, p3, p4, nrow = 2, top = "This is the main title")

**OUTPUT:**



**QUESTION:**

Load USArrests in-build dataset and correlate in the maps with anyone fields. Display the maps using colormapping.

**CODE:**

#Now, we will import a map and color-map it

library(maps)

library(dplyr)

arrest <- USArrests

View(arrest)

#When viewed, it only has 4 columns. So, we will add a new column with the states in it

#We are converting it to lower-case words as we need to match it to the states in the map's state column

arrest$region = tolower(rownames(arrest))

View(arrest)

#The 'map\_data' function contains the longitude, latitude, etc to create a map of US

View(map\_data("state"))

#Now, we will join this data with the 'arrest' dataset

statesMap <- map\_data("state")

arrestMap <- left\_join(statesMap, arrest, by = "region")

View(arrestMap)

#Here, the 'color' argument is for the border of each state

#The 'scale\_fill\_virdis\_c' function is from the 'Viridis' library. It is for applying gradient according to the value

ggplot(data = arrestMap, aes(x = long, y = lat, group = group)) +

geom\_polygon(aes(fill = Assault),

color = "white") +

scale\_fill\_viridis\_c(option = "D", direction = 1)

**OUTPUT:**

