# Lista 1 - Gráficos

**Questão 1 (*Gráfico de dispersão*)** *(2 pontos)*

Carregue o pacote do ggplot

```
library(ggplot2)
```

Utilizando o banco de dados Iris, Crie um gráfico de dispersão das variáveis "Sepal.Length", e "Sepal.Width". De acordo com as seguintes propriedades:

1. A variável "Sepal.Length" deve ser colocada no eixo x;
2. A variável "Sepal.Width" deve ser colocada no eixo y;
3. Cada ponto deve ser colorido baseado na variável "Species"
4. O titulo do gráfico deve ser "Sepal Length vs Sepal Width"
5. O subtitulo do "Iris Database"
6. O caption do gráfico deve ser "Fonte: Iris database (R-Studio)"
7. O nome do eixo x deve ser "Sepal Length"
8. O nome do eixo y deve ser "Sepal Width"

**Questão 2 (*Gráfico de barras*)** *(2 pontos)*

Utilizando o banco de dados airquality, Crie um gráfico de barras da variável "Temp". De acordo com as seguintes propriedades:

1. A cor da borda das barras deve ser preta (colour="black")
2. A cor da barras deve ser azul (fill="blue")
3. A barra deve ter transparência de 50% (alpha=0.5)
4. O titulo do gráfico deve ser "New York Air Quality Measurements"
5. O subtitulo deve ser nulo (subtitle = NULL)
6. O caption do gráfico deve ser "Fonte: airquality database (R-Studio)"
7. O nome do eixo x deve ser "Temperatura [F]"
8. O nome do eixo y deve ser "Qtd"

**Questão 3 (*Histograma*)** *(2 pontos)*

Utilizando o banco de dados diamonds (para isso carregue a biblioteca ggplot2), crie um histograma da variável "Carat". De acordo com as seguintes propriedades:

1. A cor do histograma deve ser azul (fill="blue")
2. O histograma deve ter transparência de 50% (alpha=0.5)
3. A variável y deve mostrar a densidade (y=..density..)
4. O histograma deve ter 50 *bins*
5. O histograma deve ter facetas (*Facets*) por "cut"
6. O titulo do gráfico deve ser "Carat of round cut diamonds"
7. O subtitulo deve ser "Facet by cut"

**Questão 4 (*Gráfico de linha (time series)*)** *(2 pontos)*

Primeiramente vamos converter o banco de dados "AirPassengers" para um data.frame, o código é mostrado abaixo.

1. O nome do banco convertido será "data"
2. Vamos nomear a coluna do banco "data" de "Passengers"
3. Vamos criar uma nova coluna chamado "date" com as datas associadas.
4. Vamos converter os dados da coluna "Passengers" para o tipo "numeric"

```r
# O nome do banco convertido será "data"
data <- as.data.frame(AirPassengers)

# Vamos nomear a coluna do banco "data" de "Passengers"
colnames(data) <- "Passengers"

# Vamos criar uma nova coluna chamado "date" com as datas associadas.
data$date <- seq(from=as.Date("1949-01-01"), to=as.Date("1960-12-01"), by="month")

# Vamos converter os dados da coluna "Passengers" para o tipo "numeric"
data$Passengers <-  as.numeric(data$Passengers)
```

Utilizando o banco de dados convertido, crie um gráfico de linha de acordo com as seguintes propriedades:

1. O eixo x deve conter a variável "date"
2. O eixo y deve conter a variável "Passengers"
3. Adicione uma linha, de cor vermelha, com intercept = 760, slope = 0.087
4. Adicione os seguintes elementos gráficos:
   (a) title = "Monthly Airline Passenger Numbers 1949-1960";
   (b) eixo y = "Passengers";
   (c) eixo x = "Data"

**Questão 5 (*Gráfico de dispersão com linha*)** *(2 pontos)*

Utilizando o banco de dados mtcars, crie um gráfico de dispersão colocando a variável "hp" no eixo x e a variável "mpg" no eixo y. De acordo com as seguintes propriedades:

1. O titulo do gráfico deve ser "Scatter plot"
2. O nome do eixo x deve ser "Weight (1000 lbs)"
3. O nome do eixo y deve ser "Miles/(US) gallon"
4. Deve conter uma linha de regressão linear que aproxima os dados (para isso utilize o geoma: "geom_smooth(method=lm)") tracejada (linetype="dashed") com preenchimento em azul (fill="blue")
5. O tema aplicado deve retirar os *grids* ("theme(panel.background = element_blank())")

# Lista 2 - Funções básicas

**Questão 1 (*Tipos de variáveis*)** *(2 pontos)*

Crie um data frame com 3 colunas aonde:

1. coluna1: números inteiros de 1 a 3
2. coluna2: números (float) de 1 a 3
3. coluna3: caracteres 1 a 3

(a) Faça a soma da coluna1 com a coluna1. Qual o tipo de resultado? (Inteiro, Float ou caractere)

(b) Faça a soma da coluna1 com a coluna2. Qual o tipo de resultado? (Inteiro, Float ou caractere)

(c) Faça a soma da coluna2 com a coluna3. Qual o tipo de resultado? (Inteiro, Float ou caractere)

**Questão 2 (*Looping*)** *(2 pontos)*

uma forma de gerar uma sequência é a função **"seq"**. Seus primeiros dois argumentos são "from" e "to", seguidos por um terceiro, que é "by". Crie uma sequencia do numero 1 ao numero 100, de maneira que a sequencia tenha incrementos de 7 números.

```
minha_sequencia = seq(from=1, to=100, by=7)
```

crie um looping que faz a soma de todos os elementos do vetor "minha_sequencia". Aonde a cada interação ele deve imprimir no console qual a soma atual, qual o número a ser somado, e o resultado da soma.

**Questão 3 (*Navegando no dataframe*)** *(2 pontos)*

Primeiramente vamos converter o banco de dados "AirPassengers" para um data.frame.

1. O nome do banco convertido será "data"
2. Vamos nomear a coluna do banco "data" de "Passengers"
3. Vamos criar uma nova coluna chamado "date" com as datas associadas.
4. Vamos converter os dados da coluna "Passengers" para o tipo "numeric"

```
data <- as.data.frame(AirPassengers)
colnames(data) <- "Passengers"
data$date <- seq(from=as.Date("1949-01-01"), to=as.Date("1960-12-01"), by="month")
data$Passengers <-  as.numeric(data$Passengers)
```

Utilizando o banco de dados convertido:

(a) Crie uma coluna chamada "UpDown", iniciada com o valor numérico NA (as.numeric(NA)).

(b) crie um *looping* que em cada linha verifica se o numero de passageiros foi maior ou menor do que o numero de passageiros no período anterior (linha anterior). Se for maior ele deve colocar o valor de +1 na coluna "UpDown", Se for menor ele deve colocar o valor de -1 na coluna "UpDown", Se for igual ele deve colocar o valor de 0 na coluna "UpDown".

(c) Faça um gráfico de barras da variável "UpDown"

**Questão 4 (*Funções básicas*)** *(2 pontos)*

Utilizando o banco de dados iris, crie uma cópia do banco chamada de data. No banco de dados "data" crie 4 novas variáveis:

(a) variável 1 (ln_Sepal.Length): deve ser o logaritimo neperiano da variável "Sepal.Length"

(b) variável 2 (exp_Sepal.Length): deve ser o exponencial neperiano da variável "Sepal.Length"

(c) variável 3 (std_Sepal.Length): deve seguir a formula:

$$std\_Sepal.Length_i = \frac{Sepal.Length_i - mean(data\$Sepal.Length)}{sd(data\$Sepal.Length)}$$

(d) variável 4 (iris_idx): deve ser a média aritmética das variáveis Sepal Length, Sepal Width, Petal Length, Petal Width;

**Questão 5 (*break*)** *(2 pontos)*

Faça um looping de 10.000 interações com as seguintes intruções de execução:

1. sorteie um número aleatorio entre zero e um utilizando a função runif(1);
2. verifique se o número sorteado é maior que 0.8, em caso afirmativo ele deve parar no meio utilizando o comando break; caso contrário deve continuar o processo.

# Lista 3 - Banco de dados

**Questão 1 (*Importação de base*)** *(2 pontos)*

Acesse o site do *Federal Reserve Bank of St. Louis* (https://fred.stlouisfed.org/).

Faça download das series:

1. Production of Total Industry in Germany (DEUPROINDMISMEI) em formato XLS;
2. Inflation, consumer prices for Germany (FPCPITOTLZGDEU) em formato CSV;

(a) Utilize a biblioteca **readxl** para fazer a importação do da base de dados da Produção Industrial da Alemanha. Informe o range de dados (ex: "A11:B741"), bem como o tipo de dado de cada coluna observation_date: date e DEUPROINDMISMEI: numeric). Chame a base de GER_PI.

(b) Troque o nome das colunas da base GER_PI para data e prodInd

```
colnames(GER_PI) <- c("data", "prodInd")
```

(c) Utilize a biblioteca readr para fazer a importação do da base de dados da Inflação e preços do consumidor da Alemanha. Informe o tipo de dado de cada coluna DATE: date e FPCPITOTLZGDEU: double). Chame a base de GER_Price.

(d) Troque o nome das colunas da base GER_Price para data e price

```
colnames(GER_Price) <- c("data", "price")
```

**Questão 2 (*Agrupamento de base*)** *(2 pontos)*

Note que a base GER_PI, carregada no exercício anterior, está em periodicidade mensal, enquanto a base GER_Price, carregada no exercício anterior, está em periodicidade anual.

Vamos transformar a base de periodicidade mensal para periodicidade anual.

1. Crie uma coluna com a informação do "ano" de cada registro (dica: para isso utilize a biblioteca lubridate)
2. Crie um data frame com a informação de agrupamento.
3. Agrupe os dados da base GER_PI por ano, tirando a média da variável "prodInd" (dica: para isso utilize o summarise do pacote dplyr).
4. Salve os resultados do agrupamento em um data frame com nome "GER_PI.Anual"

**Questão 3 (*Agrupamento de base*)** *(2 pontos)*

Uma transformação comum é criar uma variável nova que é o logaritmo de uma outra variável.

(a) Utilizando a base GER_PI.Anual, crie uma nova variável que é o logaritmo neperiano da variável "ProdIndAnual" (chame a variável de "ln_prod").

(b) Utilizando a base GER_Price, tente criar uma nova variável que é o logaritmo neperiano da variável "price". É mostrado alguma mensagem? Caso afirmativo, porque a mensagem aparece?

(c) Utilizando a base GER_Price, mostre as "Estatisticas descritivas" das colunas, utilizando a função summary. Existe valores negativos?

(d) Utilizando a base GER_Price, filtre as linhas que tem valores de preço abaixo de zero.

**Questão 4 (*Inner join*)** *(2 pontos)*

Vamos juntar as duas bases de dados para fazer uma análise dos dados. Primeiramente note que precisamos de uma coluna somente com o ano na base GER_Price.

(a) Crie uma coluna com a informação do "ano" de cada registro da base GER_Price (dica: para isso utilize a biblioteca lubridate)

(b) Utilizando a base GER_Price e GER_PI.Anual junte as duas bases utilizando o *inner join*.

(c) Com isso salve os dados em um data frame chamado "FullData"

(d) Faça um gráfico de dispersão do das variáveis **"ln_prod"** e **"price"** juntamente com uma linha de regressão (comando "geom_smooth").

**Questão 5 (*Pivot longer e exportação*)** *(2 pontos)*

Utilizando o banco de dados "FullData"

(a) Selecione apenas as colunas ano, price, ln_prod.

(b) Utilizando a biblioteca tidyr, transforme a base de dados de maneira que ela tenha a seguinte estrutura: (coluna 1) Ano: ano da informação; (coluna 2) Série: nome da série; (coluna 3) valor: O valor da série.

| ano | serie | valor |
|---|---|---|
| 1960 | price | 1.54 |
| 1960 | ln_prod | 3.34 |
| 1961 | price | 2.29 |
| 1961 | ln_prod | 3.41 |
| 1962 | price | 2.84 |
| 1962 | ln_prod | 3.49 |

Table 1: Exemplo dos dados

(c) Com o banco de dados transformado, faça um gráfico de linha da variável **"valor"** com o **"ano"** no eixo x, colorido pela variável **"Série"**.

(d) Salve os dados transformados em um arquivo "CSV"

# Lista 4 - Funções

O objetivo desta lista e criar um mini-projeto, o qual contem todas as camadas (Camanda de dados, camada de logica, e camada do Usuário). O projeto consiste em fazer a analise dos dados da COVID-19.

**Questão 1 (*Criando um projeto*)** *(2 pontos)*

Crie um projeto no R-Studio com a seguinte estrutura:

1. Um diretório chamado: "Database"
2. Um diretório chamado: "Graficos"
3. Baixe a base de dados do site https://ourworldindata.org/coronavirus-source-data
4. Salve a base no diretório "Database" com o nome "owid-covid-data.xlsx"

(Uma versão antiga da base pode ser obtida no e-class)

Caso você não esteja utilizando o R-Studio apenas organize seu diretório de maneira a seguir a estrutura sugerida.

**Questão 2 (*Camada de dados(Data Loader)*)** *(2 pontos)*

Crie um script chamado "DataLoader.r", o script deve realizar as seguintes tarefas:

1. Abrir o banco de dados de nome "owid-covid-data.xlsx" que está no diretório "Database"
2. A coluna data deve ser convertida para o tipo data. (garantindo que a coluna é do tipo "Date")
3. O script deve filtrar a base e mostrar apenas ose seguintes paises: "Brazil", "United States", "Mexico", "Germany", "France", "United Kingdom"
4. O script deve selecionar apenas as variáveis: **location**, **date**, **total_cases**, **new_cases**

**Questão 3 (*Função*)** *(2 pontos)*

Crie uma função que recebe um vetor com datas e determina o dia da semana bem como faz uma contagem das semanas.

1. A função deve retornar um data.frame com as seguintes colunas: date, weekday, week. Exemplo:

| date | weekday | week |
|---|---|---|
| 2020-01-01 | 4 | 1 |
| 2020-01-02 | 5 | 1 |
| 2020-01-03 | 6 | 1 |
| 2020-01-04 | 7 | 1 |
| 2020-01-05 | 1 | 2 |
| 2020-01-06 | 2 | 2 |
| 2020-01-07 | 3 | 2 |
| 2020-01-08 | 4 | 2 |
| 2020-01-09 | 5 | 2 |
| 2020-01-10 | 6 | 2 |
| 2020-01-11 | 7 | 2 |
| 2020-01-12 | 1 | 3 |

**Questão 4 (*Camada lógica*)** *(2 pontos)*

Crie um script chamado "main.r", o script deve fazer as seguintes tarefas:

1. Carregar a função criada anteriormente.
2. Carregar o script DataLoader.r

3. Utilizar a função criada para determinar qual a semana associada a cada registro na base "covid_data". (dica utilizar *join*)
4. Agrupar os dados por semana para cada pais selecionado. (dica: utilize *group_by* e *summarise*)
5. excluir a informação da ultima semana pois essa pode estar incompleta.

**Questão 5 (*Camada do usuário*)** *(2 pontos)*

Crie uma rotina que cria 2 gráficos e salva eles no diretório "Graficos"

1. Gráfico 1: utilizando os dados agrupados, crie um gráfico de linha do total de mortes por covid-19, com uma serie para cada pais selecionado. (eixo x: Semana, eixo y: Total de mortes).
2. Gráfico 2: utilizando os dados agrupados, crie um gráfico de linha das novas de mortes por covid-19, com uma serie para cada pais selecionado. (eixo x: Semana, eixo y: Novas mortes)
3. Salve os dois gráficos no diretório "Graficos". (Configurações recomendadas: scale=1, units = "in", dpi = 300, width = 10.4, height = 5.85)