

AULA 1 – PRIMEIROS PASSOS NO R E EXPLORAÇÃO DE DADOS

INTRODUÇÃO

Ao longo do nosso curso, trabalharemos majoritariamente com análise de dados em R. Em todo o material teremos **em vermelho** o código a ser inserido no R e logo abaixo, se necessário, uma breve explicação **sobre o código**.

Breve Introdução ao R e Suas Principais Funções:

O R é uma linguagem de programação e ambiente de software livre para análise estatística e gráficos. Ele é amplamente utilizado em estatística e ciência de dados pois é gratuito, aberto e possui uma série de bibliotecas importantes, além de claro, uma comunidade muito forte..

Algumas de suas principais funções:

- **read_csv():** Para importar dados de arquivos CSV.
- **head() e tail():** Para visualizar as primeiras ou últimas linhas de um dataframe.
- **summary():** Para obter um resumo estatístico das variáveis.

Na maioria das vezes, vamos trabalhar com Dataframes no R.

Sobre Dataframes

É uma estrutura de dados bidimensional que pode ser imaginada como uma tabela ou planilha em que os dados estão organizados em linhas e colunas.

Algumas Características

Estrutura Tabular:

Linhas e Colunas: Cada coluna representa uma variável e cada linha representa uma observação ou registro.

Tipos de Dados:

As colunas de um dataframe podem conter diferentes tipos de dados, como números, caracteres (strings), fatores, datas, etc.

Nomeação:

Colunas e Linhas: As colunas têm nomes (que são vetores de caracteres) e as linhas têm índices numéricos (embora os nomes das linhas possam ser definidos).

Manipulação:

Operações: Dataframes suportam diversas operações como filtragem, agregação, e transformação de dados através de funções como subset(), filter(), mutate(), e muitas outras

Vamos criar um dataframe simples

```
df <- data.frame(
  Nome = c("Ana", "João", "Pedro"),
  Idade = c(23, 35, 29),
  Cidade = c("São Paulo", "Rio de Janeiro", "Belo Horizonte"))
```

Visualizando o dataframe

```
print(df)
```

Tipos de dados mais comuns em um dataframe

numeric: Usado para armazenar valores numéricos com casas decimais.

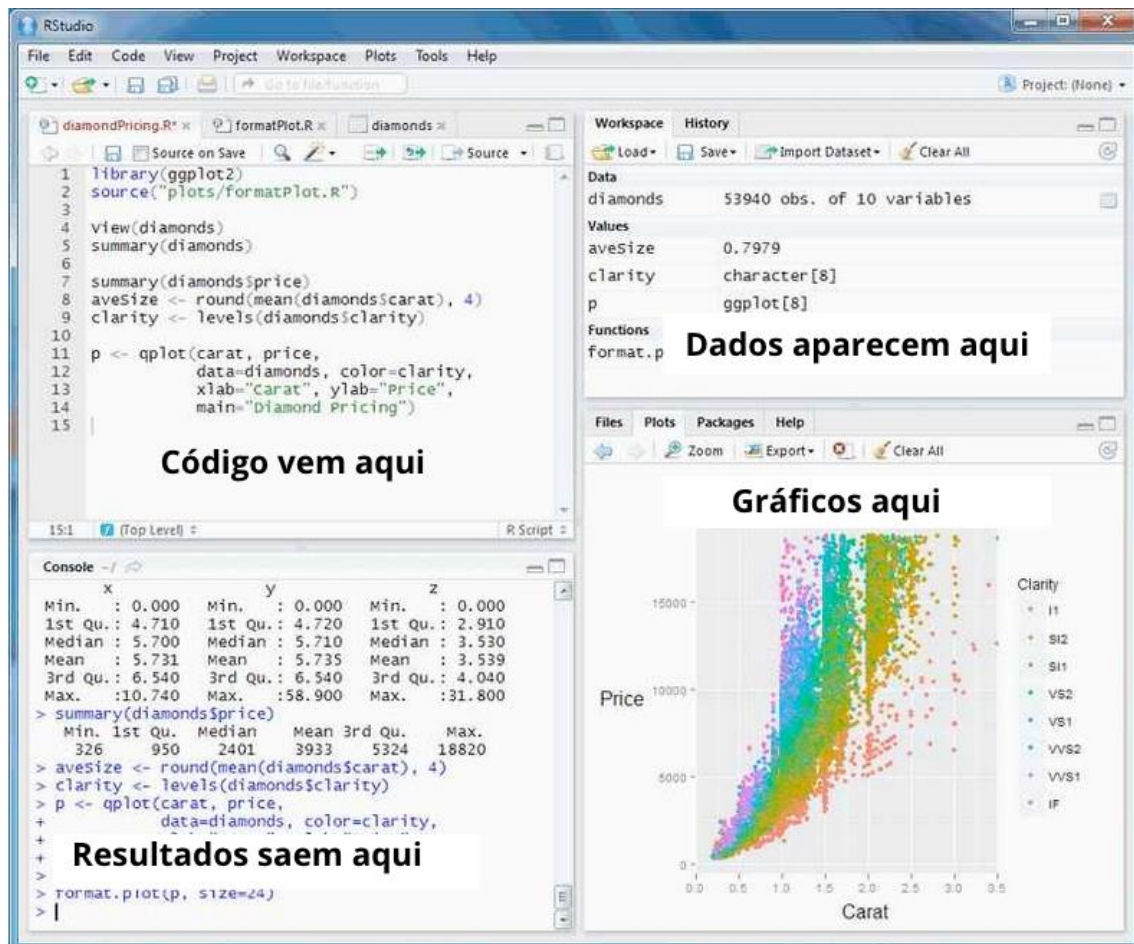
integer: Usado para armazenar números inteiros.

character: Usado para armazenar texto.

factor: Usado para representar variáveis categóricas, onde os valores são limitados a um conjunto específico de níveis

SOBRE O RSTUDIO

CONSOLE X SCRIPT



COMENTÁRIOS

Tudo o que você escrever na linha de código será lido como... código. Caso você queira fazer um comentário ou uma anotação, você terá que usar uma hashtag.

```
# fazer comentários assim que o R não lê.
```

Pacotes

O R é composto por diversos pacotes. Todo pacote que você precisará puxar, você o fará da seguinte forma

```
install.packages("tidyverse") # instala o pacote tidyverse  
library(tidyverse) # abre o pacote tidyverse
```

Sempre que você instalar um pacote, você precisará chamar ele no 'library'. Após um pacote instalado, basta você apenas chamar a library, não precisando instalar ele novamente.

Nesta primeira aula vamos trabalhar a exploração de um banco de dados do AIRBNB. A ideia é que tenhamos um primeiro contato com o R e com as possibilidades do software. As variáveis do banco são:

Variável	Definição
room_id	Um número exclusivo que identifica um anúncio do Airbnb. O anúncio tem uma URL no site do Airbnb de http://airbnb.com/rooms/room_id
host_id	Um número exclusivo que identifica um anfitrião do Airbnb. A página do anfitrião tem uma URL no site do Airbnb de http://airbnb.com/users/show/host_id
room_type	Um de “Casa/apto inteiro”, “Quarto privado” ou “Quarto compartilhado”
borough	Uma sub-região da cidade ou área de busca para a qual a pesquisa é realizada. O distrito é tirado de um shapefile da cidade que é obtido independentemente do site do Airbnb.
neighborhood	Assim como borough: uma sub-região da cidade ou área de busca para a qual a pesquisa é realizada. Para cidades que têm ambos, um bairro é menor do que um borough. Para algumas cidades, não há informações sobre o bairro.
reviews	O número de avaliações que um anúncio recebeu. O Airbnb disse que 70% das visitas terminam com uma avaliação, então o número de avaliações pode ser usado para estimar o número de visitas. Observe que tal estimativa não será confiável para um anúncio individual (especialmente porque as avaliações ocasionalmente desaparecem do site), mas para uma cidade como um todo, deve ser uma métrica útil de tráfego.
overall_satisfaction	A classificação média (de cinco) que o anúncio recebeu dos visitantes que deixaram uma avaliação.
accommodates	O número de hóspedes que um anúncio pode acomodar
bedrooms	O número de quartos que um anúncio oferece.
price	O preço (em \$US) para uma estadia noturna. Em pesquisas anteriores, pode haver alguns valores que foram registrados por mês.
minstay	Estadia mínima para uma visita, conforme publicada pelo anfitrião.

latitude and longitude	A latitude e a longitude do anúncio conforme publicadas no site do Airbnb: podem estar erradas em algumas centenas de metros*.
last_modified	A data e a hora em que os valores foram lidos no site do Airbnb.

IMPORTANDO DADOS PARA O R

CRIANDO PASTA DE TRABALHO

Criar pasta na área de trabalho chamada **marketing** e inserir o documento dentro

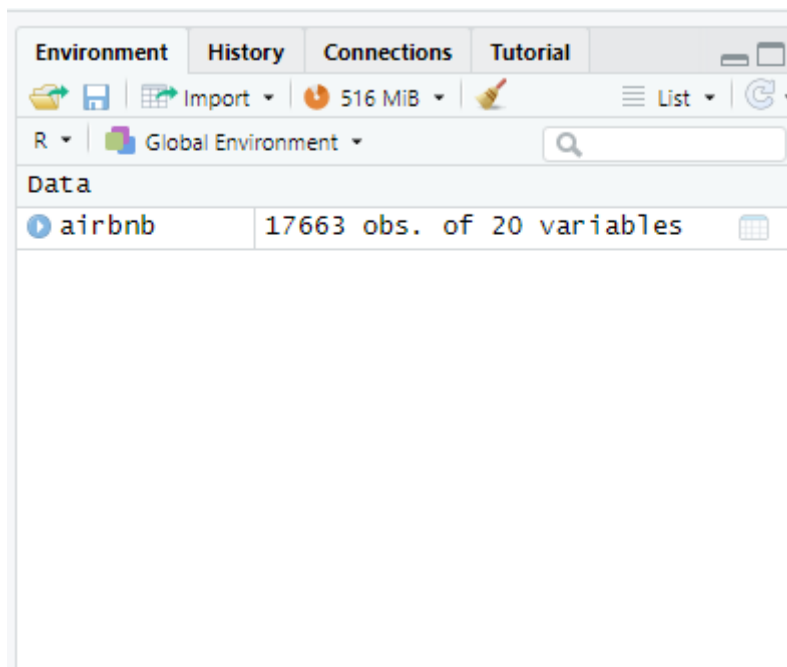
CONFIGURANDO SEU DIRETÓRIO

Session > Set Working Directory > Choose Directory > **Selecionar pasta marketing**

ATRIBUINDO DADOS A OBJETOS

Aqui nós vamos chamar o documento do diretório para dentro do R

```
airbnb <- read.csv("aula1.csv", sep = ";", Header = TRUE)
```



Sobre o código

aula1.csv – nosso documento

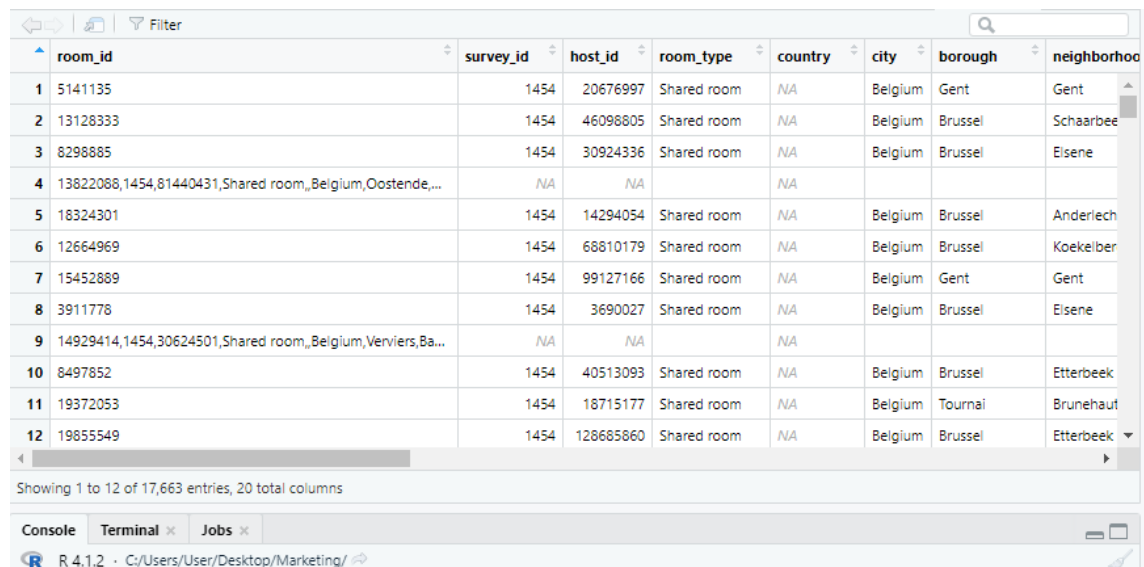
sep – Como é um arquivo csv, estamos dizendo que o separador das colunas é o “;”.

Header – Estamos dizendo que a primeira linha contém os nomes das colunas

INSPEÇÃO DA BASE DE DADOS

Abrindo o frame para visualizarmos

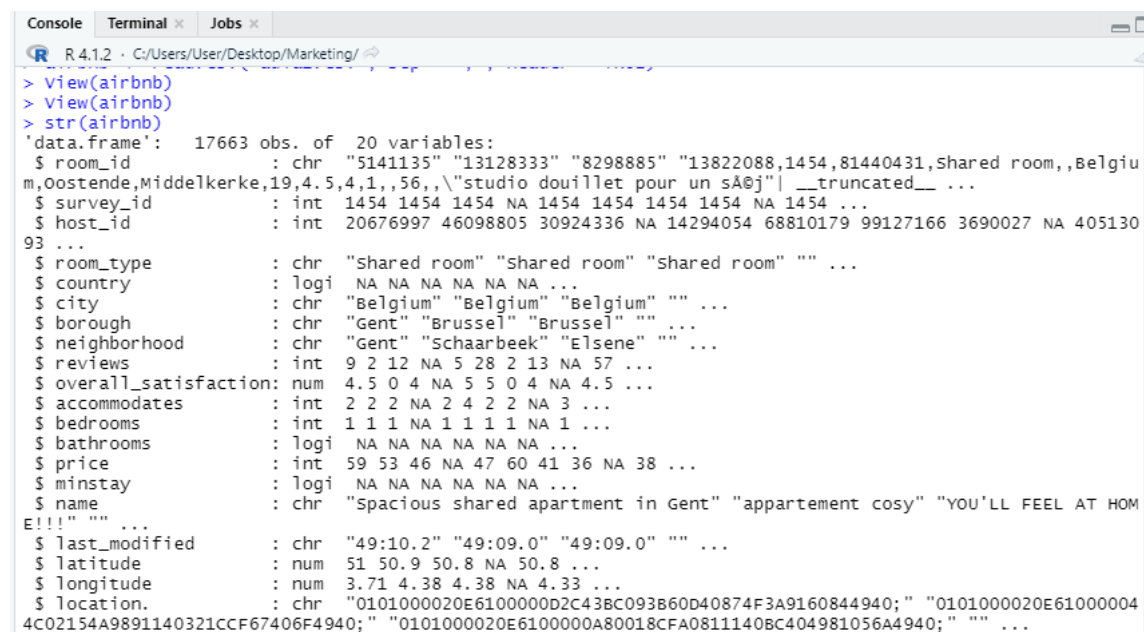
View(airbnb)



	room_id	survey_id	host_id	room_type	country	city	borough	neighborhood
1	5141135	1454	20676997	Shared room	NA	Belgium	Gent	Gent
2	13128333	1454	46098805	Shared room	NA	Belgium	Brussel	Schaarbee
3	8298885	1454	30924336	Shared room	NA	Belgium	Brussel	Elsene
4	13822088,1454,81440431,Shared room,Belgium,Oostende,...	NA	NA		NA			
5	18324301	1454	14294054	Shared room	NA	Belgium	Brussel	Anderlech
6	12664969	1454	68810179	Shared room	NA	Belgium	Brussel	Koekeiber
7	15452889	1454	99127166	Shared room	NA	Belgium	Gent	Gent
8	3911778	1454	3690027	Shared room	NA	Belgium	Brussel	Elsene
9	14929414,1454,30624501,Shared room,Belgium,Verviers,Ba...	NA	NA		NA			
10	8497852	1454	40513093	Shared room	NA	Belgium	Brussel	Etterbeek
11	19372053	1454	18715177	Shared room	NA	Belgium	Tournai	Brunehaut
12	19855549	1454	128685860	Shared room	NA	Belgium	Brussel	Etterbeek

VERIFICANDO ESTRUTURA DOS DADOS

str(airbnb)



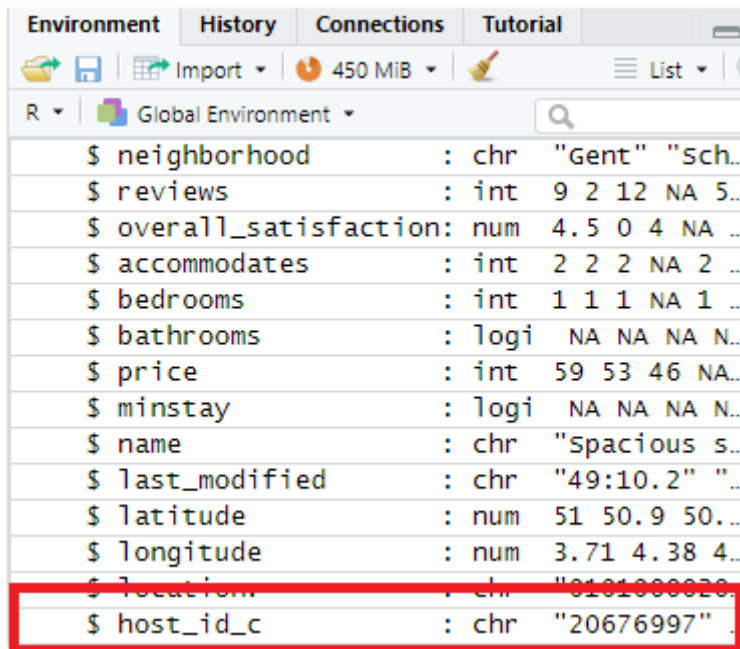
```
> view(airbnb)
> view(airbnb)
> str(airbnb)
'data.frame': 17663 obs. of 20 variables:
 $ room_id      : chr "5141135" "13128333" "8298885" "13822088,1454,81440431,Shared room,,Belgiu
m,Oostende,Middelkerke,19,4.5,4,1,,56,,\"studio douillet pour un sÅ@j\"| __truncated__ ...
 $ survey_id    : int  1454 1454 1454 NA 1454 1454 1454 1454 NA 1454 ...
 $ host_id      : int  20676997 46098805 30924336 NA 14294054 68810179 99127166 3690027 NA 405130
93 ...
 $ room_type    : chr  "shared room" "shared room" "shared room" "" ...
 $ country      : logi  NA NA NA NA NA NA ...
 $ city         : chr  "Belgium" "Belgium" "Belgium" "" ...
 $ borough      : chr  "Gent" "Brussel" "Brussel" "" ...
 $ neighborhood : chr  "Gent" "Schaarbeek" "Elsene" "" ...
 $ reviews     : int   9 2 12 NA 5 28 2 13 NA 57 ...
 $ overall_satisfaction: num  4.5 0 4 NA 5 5 0 4 NA 4.5 ...
 $ accommodates : int   2 2 2 NA 2 4 2 2 NA 3 ...
 $ bedrooms     : int   1 1 1 NA 1 1 1 1 NA 1 ...
 $ bathrooms    : logi  NA NA NA NA NA NA ...
 $ price        : int   59 53 46 NA 47 60 41 36 NA 38 ...
 $ minstay      : logi  NA NA NA NA NA NA ...
 $ name         : chr  "Spacious shared apartment in Gent" "appartement cosy" "YOU'LL FEEL AT HOM
E!!!" "" ...
 $ last_modified : chr  "49:10.2" "49:09.0" "49:09.0" "" ...
 $ latitude     : num   51 50.9 50.8 NA 50.8 ...
 $ longitude    : num   3.71 4.38 4.38 NA 4.33 ...
 $ location     : chr  "0101000020E6100000D2C438C093860D40874F3A9160844940;" "0101000020E61000004
4C02154A9891140321CCF67406F4940;" "0101000020E6100000A80018CFA0811140BC404981056A4940;" "" ...
```

TRANSFORMANDO VARIÁVEIS

TRANSFORMANDO EM CARACTERES

O R interpreta a coluna `host_id` como números inteiros (*integer*). Isso pode causar uma confusão pois o software entende que essas colunas são passíveis de operações matemáticas – o que não faz sentido. Nesse sentido, vamos transformar eles em fatores.

```
airbnb$host_id_c <- as.character(airbnb$host_id)
```



\$ neighborhood	:	chr	"Gent"	"Sch...
\$ reviews	:	int	9 2 12	NA 5...
\$ overall_satisfaction	:	num	4.5 0 4	NA ...
\$ accommodates	:	int	2 2 2	NA 2 ...
\$ bedrooms	:	int	1 1 1	NA 1 ...
\$ bathrooms	:	logi	NA NA NA	N...
\$ price	:	int	59 53 46	NA...
\$ minstay	:	logi	NA NA NA	N...
\$ name	:	chr	"Spacious s...	
\$ last_modified	:	chr	"49:10.2"	"...
\$ latitude	:	num	51 50.9 50...	
\$ longitude	:	num	3.71 4.38 4...	
\$ location	:	chr	"8101000020"	
\$ host_id_c	:	chr	"20676997"	

Sobre o código

Para acessarmos uma coluna específica, utilizamos: o nome do banco + \$ + nome da coluna. Nesse caso estamos transformando a coluna `airbnb$host_id` em `airbnb$room_id_c` (*criando uma nova coluna*)

Você também pode transformar a mesma variável sem criar uma nova. Basta indicar ela e transformar ela mesma na frente. Exemplo:

```
airbnb$host_id <- as.character(airbnb$host_id)
```

TRANSFORMAÇÕES NUMÉRICAS

Em nosso banco de dados, a satisfação geral é uma nota de 0 a 5.

```
head(airbnb$overall_satisfaction)
```

```
> head(airbnb$overall_satisfaction)
[1] 4.5 0.0 4.0 NA 5.0 5.0
> |
```

Podemos transformar ela em uma nota de 0 até 100 por exemplo.

```
airbnb$overall_satisfaction_100 <- airbnb$overall_satisfaction * 20
head(airbnb$overall_satisfaction_100)
```

```
> head(airbnb$overall_satisfaction_100)
[1] 90 0 80 NA 100 100
> |
```

Sobre o código

Head – Visualiza algo. No caso, visualizamos a coluna `airbnb$overall_satisfaction` ou a coluna nova `airbnb$overall_satisfaction_100`

Multiplicamos a satisfação por 20 (`airbnb$overall_satisfaction*20`) e atribuímos a uma nova coluna `airbnb$overall_satisfaction_100`

TRANSFORMAÇÕES COM MUTATE

É possível fazer todas as operações anteriores de uma vez só com o Mutate. (é preciso estar com o pacote dply instalado e carregado)

```
airbnb <- mutate(airbnb,  
  host_id_c = as.character(host_id),  
  overall_satisfaction_100 = overall_satisfaction * 20)
```

INCLUIR OU EXCLUIR E RENOMEAR VARIÁVEIS (COLUNAS)

Problema: País está vazio, cidade está com nome de país e a sub-região está com nome da cidade.

country	city	borough
NA	Belgium	Gent
NA	Belgium	Brussel
NA	Belgium	Brussel
NA		
NA	Belgium	Brussel
NA	Belgium	Brussel
NA	Belgium	Gent
NA	Belgium	Brussel
NA		
NA	Belgium	Brussel
NA	Belgium	Tournai
NA	Belgium	Brussel

Muitas vezes precisamos alterar o nosso dataset por algum problema ou qualquer outra questão. Por vezes precisamos excluir alguma variável do nosso ou renomear alguma variável. Nesse caso, vamos excluir as variáveis `country` e `survey_id`. Além disso, vamos renomear algumas variáveis.

Excluir

```
airbnb <- select(airbnb, -country, -survey_id)
```

Renomear

```
airbnb <- rename(airbnb, country = city, city = borough)
```

room_id	host_id	room_type	country	city	neighborhood
5141135	20676997	Shared room	Belgium	Gent	Gent
13128333	46098805	Shared room	Belgium	Brussel	Schaarbeek
8298885	30924336	Shared room	Belgium	Brussel	Eisene
13822088,1454,81440431,Shared room,,Belgium,Oostende,...	NA				
18324301	14294054	Shared room	Belgium	Brussel	Anderlecht
12664969	68810179	Shared room	Belgium	Brussel	Koekelberg
15452889	99127166	Shared room	Belgium	Gent	Gent
3911778	3690027	Shared room	Belgium	Brussel	Eisene
14929414,1454,30624501,Shared room,,Belgium,Verviers,Ba...	NA				
8497852	40513093	Shared room	Belgium	Brussel	Etterbeek
19372053	18715177	Shared room	Belgium	Tournai	Brunehaut
19855549	128685860	Shared room	Belgium	Brussel	Etterbeek

INCLUIR E EXCLUIR OBSERVAÇÕES (LINHAS)

Criar um Vetor com C

Mais adiante, vamos criar um gráfico com os preços das 10 maiores cidades. Para isso, vamos criar um vetor com os nomes dessas 10 maiores cidades, para mais adiante incluirmos os dados apenas dessas cidades.

Vetor de palavras

```
airbnb_topten
c("Brussel", "Antwerpen", "Gent", "Charleroi", "Liege", "Brugge", "Namur", "Leuven", "Mons", "Aalst").
airbnb_topten
```

```

Console Terminal x Jobs x
R 4.1.2 · C:/Users/User/Desktop/Marketing/
> airbnb_topten
[1] "Brussel" "Antwerpen" "Gent" "Charleroi" "Liege" "Brugge" "Namur" "Leuven"
[9] "Mons" "Aalst"
>

```

Vetor de números

```
number_vector <- c(0,2,4,6)
number_vector
```

```

> number_vector <- c(0,2,4,6)
> number_vector
[1] 0 2 4 6

```

Note que para o vetor de palavras, é obrigatório inserir os “”. Quando é um vetor de números, não é necessário.

INCLUIR OU EXCLUIR OBSERVAÇÕES COM A FUNÇÃO *FILTER*

Instalar pacote Hmisc

```
install.packages("Hmisc")
library(Hmisc)
```

FILTER

Queremos tiltrar os dados para apenas as cidades que são as topten (Variável *airbnb_top10* que criamos anteriormente).

```
airbnb_top10 <- filter(airbnb, city %in% topten)
```

Como era. Note que as cidades Nivelles e Oostende estão no dataframe anterior.

67	7479719	37665177	Shared room	Belgium	Mechelen
68	5609860	29082746	Shared room	Belgium	Nivelles
69	17306887	45269039	Shared room	Belgium	Mons
70	8422305	44356910	Shared room	Belgium	Oostende

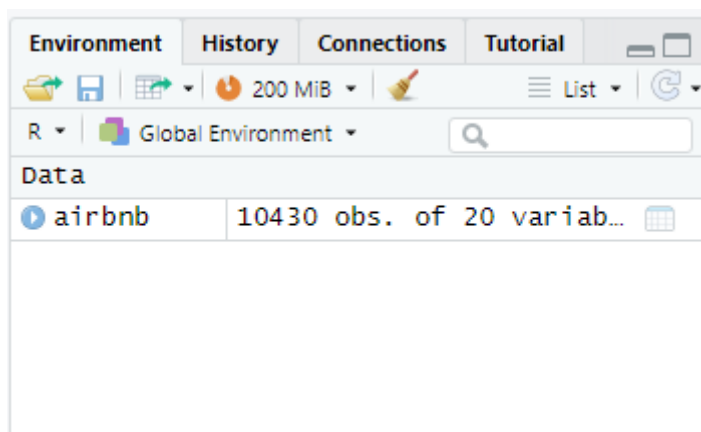
Como está

66	13838407	81635541	Shared room	Belgium	Brussel
67	15459583	64870905	Shared room	Belgium	Brussel
68	18303344	119596943	Shared room	Belgium	Brussel
69	13856670	81635541	Shared room	Belgium	Brussel

OPERADOR PIPE

Tudo o que fizemos até agora, foi feito linha por linha. Podemos fazer com que as linhas encadeiem as funções de maneira legível utilizando o operador pipe '*%>%*'. Automaticamente, o resultado de uma função vira entrada para a outra.

```
airbnb <- read.csv("aula1.csv") %>%  
  mutate(host_id_c = as.character(host_id), overall_satisfaction_100 = overall_satisfaction * 20)  
%>%  
  select(-country, -survey_id) %>%  
  rename(country = city, city = borough) %>%  
  filter(city %in%  
c("Brussel", "Antwerpen", "Gent", "Charleroi", "Liege", "Brugge", "Namur", "Leuven", "Mons", "Aalst"))
```



AGRUPAR E SUMARIZAR

Agrupar e Sumarizar por Cidade

Quantas observações existem por cidade? Vamos agrupar as cidades e contar o número de observações por cidade.

```
airbnb %>%  
group_by(city) %>%  
summarise(nr_per_city = n())
```

```
# A tibble: 9 x 2  
  city      nr_per_city  
  <chr>      <int>  
1 Aalst         67  
2 Antwerpen    1445  
3 Brugge       946  
4 Brussel     6071  
5 Charleroi     110  
6 Gent        1055  
7 Leuven       376  
8 Mons        107  
9 Namur       253  
> |
```

Sobre o código

Group_by – Agrupa as observações por cidade

Summarise – conta o numero de observações por cidade.

COLOCAR EM ORDEM (CRESCENTE)

```
airbnb %>%  
group_by(city) %>%  
summarise(nr_per_city = n()) %>%  
arrange(nr_per_city)
```

```
# A tibble: 9 x 2  
  city      nr_per_city  
  <chr>      <int>  
1 Aalst         67  
2 Mons        107  
3 Charleroi     110  
4 Namur       253  
5 Leuven       376  
6 Brugge       946  
7 Gent        1055  
8 Antwerpen    1445  
9 Brussel     6071
```

Sobre o código

Arrange coloca em ordem crescente por default

COLOCAR EM ORDEM (DESCRESCENTE)

```
airbnb %>%  
group_by(city) %>%  
summarise(nr_per_city = n()) %>%  
arrange(desc(nr_per_city))
```

	city	nr_per_city
	<chr>	<int>
1	Brussel	6071
2	Antwerpen	1445
3	Gent	1055
4	Brugge	946
5	Leuven	376
6	Namur	253
7	Charleroi	110
8	Mons	107
9	Aalst	67

Sobre o código

Inserir o *desc* transforma em ordem decrescente.

ESTATÍSTICAS DESCRITIVAS

O jeito mais rápido de você verificar as estatísticas descritivas de uma coluna é com o `summary`.

Vamos ver as estatísticas descritivas de preço geral

`Summary(airbnb$price)`

```
> summary(airbnb$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 10.00   47.00   70.00   84.62  100.00 3536.00
```

Vamos agora fazer análises mais aprofundadas.

Qual o preço médio por cidade?

```
airbnb.summary <- airbnb %>%
  group_by(city) %>%
  summarise(nr_per_city = n(), average_price = mean(price)) %>%
  arrange(desc(average_price))
print(airbnb.summary, n = Inf)
```

```
# A tibble: 9 x 3
  city      nr_per_city average_price
  <chr>      <int>      <dbl>
1 Brugge      946      126.
2 Namur       253      107.
3 Aalst        67       96.6
4 Antwerpen  1445       94.7
5 Gent       1055       91.0
6 Mons        107       81.8
7 Charleroi   110       76.8
8 Leuven       376       74.8
9 Brussel   6071       74.4
```

Sobre o código

`Group_by` – Agrupa por cidade

`nr_per_city = n()` – Cria uma variável que conta o número de registros por cidade

`average_price = mean(price)` – Cria uma variável que calcula o preço médio por cidade

`arrange(desc(average_price))` – Ordena as cidades pelos preços médios em ordem decrescente

E se quisermos calcular a Mediana e o preço Máximo...

```
airbnb %>%
  group_by(city) %>%
  summarise(nr_per_city = n(),
            average_price = mean(price),
            median_price = median(price),
            max_price = max(price)) %>%
  arrange(desc(median_price),
           desc(max_price))
```

```
# A tibble: 9 x 5
  city          nr_per_city average_price median_price max_price
  <chr>          <int>         <dbl>         <dbl>         <int>
1 Brugge           946          126.           105           884
2 Gent            1055           91.0            76           708
3 Antwerpen        1445           94.7            74          1473
4 Aalst             67           96.6            65           754
5 Mons             107           81.8            65           448
6 Namur            253          107.            64          2239
7 Brussel          6071           74.4            59          3536
8 Leuven           376           74.8            58          1178
9 Charleroi        110           76.8            57           464
```

Sobre o Código

O mesmo que o anterior mais:

median_price = *median(price)* – Mediana do Preço

max_price = *max(price)* – Preço Máximo

arrange(desc(median_price), desc(max_price)) – Ordena em ordem descendente primeiro a mediana e depois o máximo.

GRÁFICOS

Instalar pacote Hmisc

```
install.packages("Hmisc")
install.packages("ggplot2")
library(ggplot2)
library(Hmisc)
```

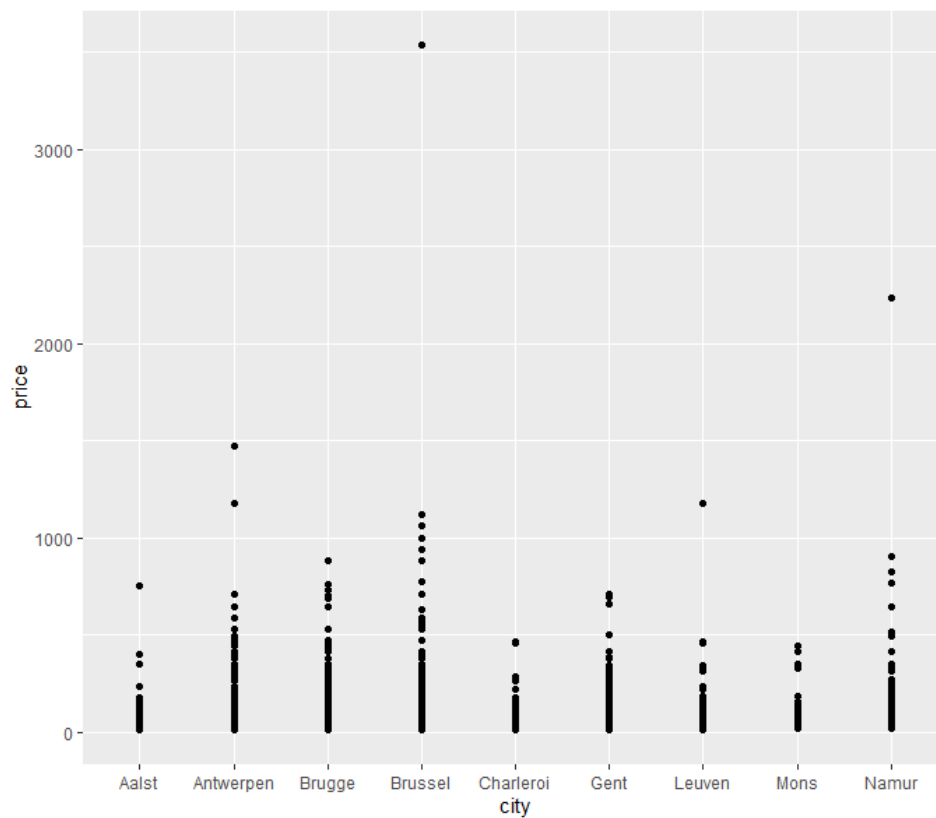
Selecionando as 10 cidades Principais

```
airbnb_top10 <- airbnb %>%
  filter(city %in%
c("Brussel", "Antwerpen", "Gent", "Charleroi", "Liege", "Brugge", "Namur", "Leuven", "Mons", "Aalst"))
```

SCATTERPLOT

As 10 cidades pelo preço

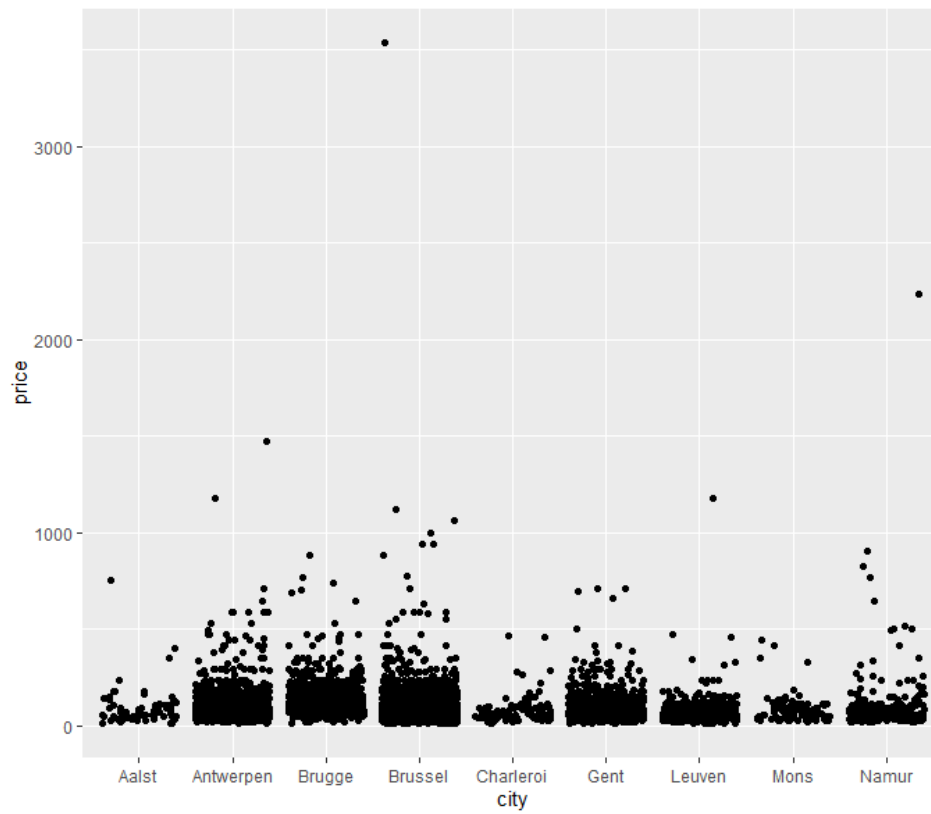
```
ggplot(data = airbnb_top10, mapping = aes(x = city, y = price)) +
  geom_point()
```



JITTER

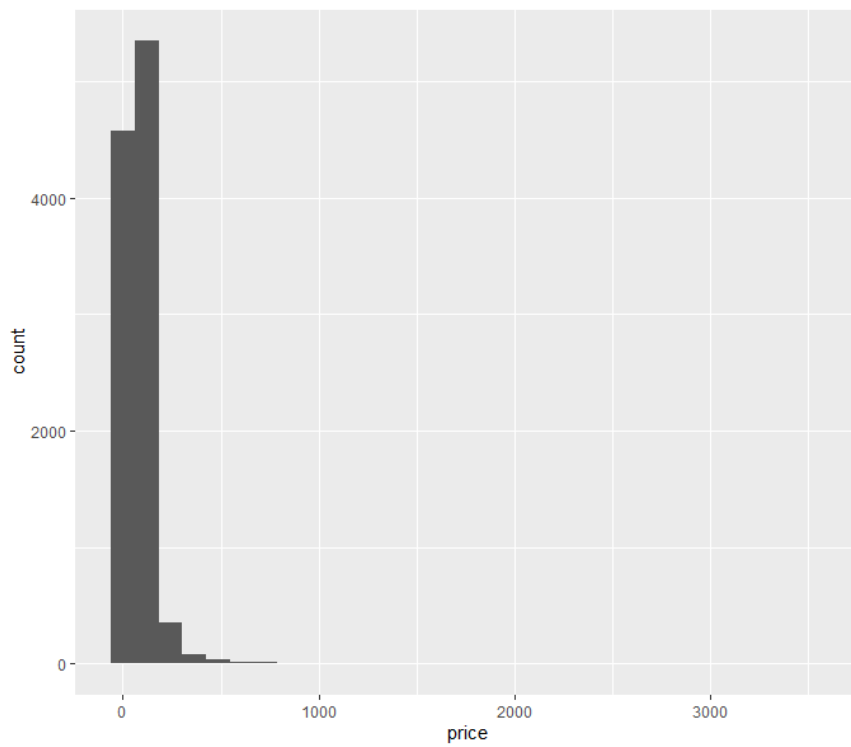
Para não se sobrepor, colocamos os dados 'desagregados'

```
ggplot(data = airbnb_topten, mapping = aes(x = city, y = price)) +  
geom_jitter()
```



HISTOGRAMMA

```
ggplot(data = airbnb_topten, mapping = aes(x = price)) +  
geom_histogram()
```



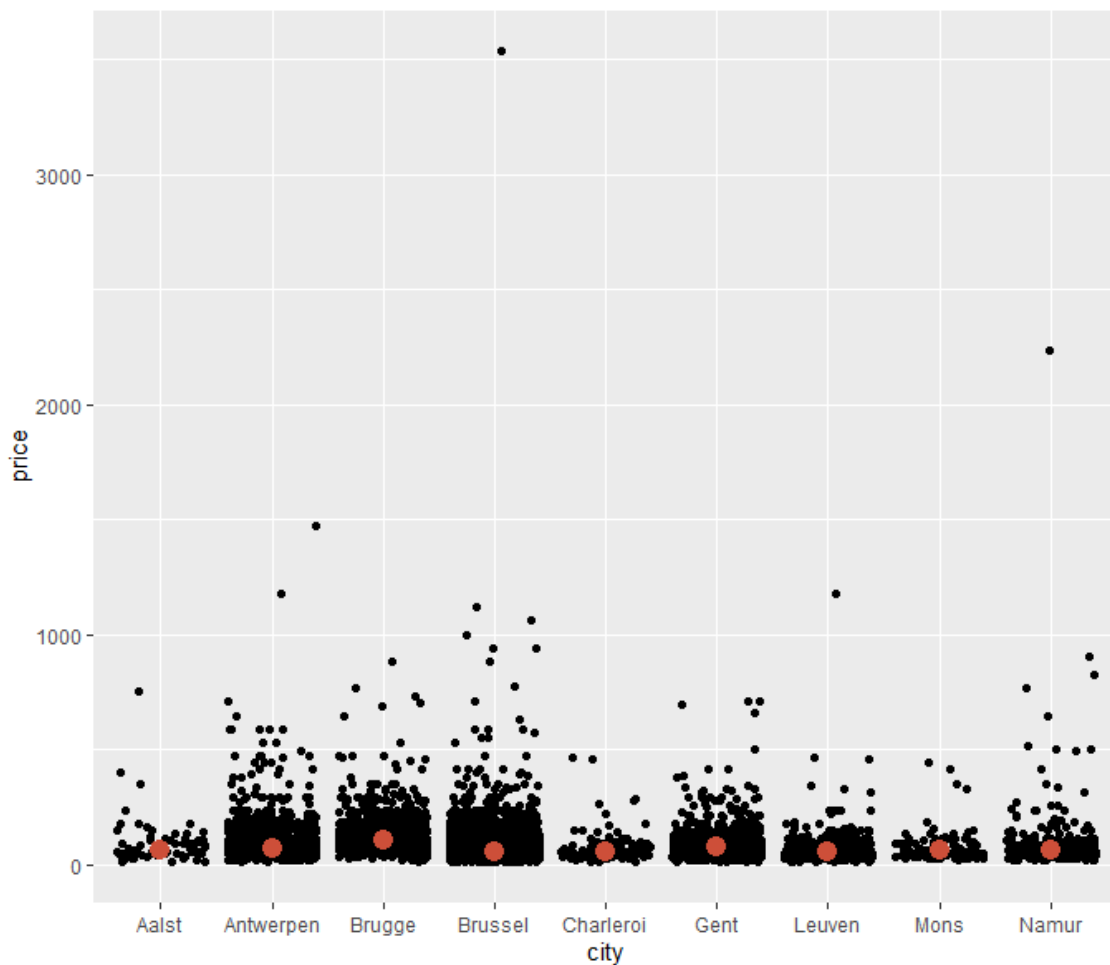
Sobre o código

ggplot(data = airbnb.topten, mapping = aes(x = price)): Cria um gráfico com base no dataframe `airbnb.topten`, mapeando a variável `price` para o eixo X.

geom_histogram(): Adiciona um histograma ao gráfico, onde o eixo X representa os preços e o eixo Y representa a frequência desses preços.

PLOTAR A MÉDIA

```
ggplot(data = airbnb.topten, mapping = aes(x = city, y = price)) +  
  geom_jitter() +  
  stat_summary(fun.y=median, colour="tomato3", size = 4, geom="point")
```



EXPORTAR

Para exportar em csv.

```
library(readr)  
write_excel_csv(airbnb, "airbnb.csv")
```

```

# CONFIGURANDO SEU DIRETÓRIO
# Criar pasta na área de trabalho chamada marketing e inserir o documento dentro
# Session > Set Working Directory > Choose Directory > Selecionar pasta marketing

# ATRIBUINDO DADOS A OBJETOS
airbnb <- read.csv("aula1.csv", sep = ",", header = TRUE)
View(airbnb)

#VERIFICAR ESTRUTURA DOS DADOS
str(airbnb)

# TRANSFORMANDO EM FATORES
airbnb$host_id_c <- as.character(airbnb$host_id)

# TRANSFORMAÇÕES NUMÉRICAS
head(airbnb$overall_satisfaction)
airbnb$overall_satisfaction_100 <- airbnb$overall_satisfaction * 20
head(airbnb$overall_satisfaction_100)

# TRANSFORMAÇÕES COM MUTATE
airbnb <- mutate(airbnb,
  host_id_c = as.character(host_id),
  overall_satisfaction_100 = overall_satisfaction * 20)

#INSTALANDO DPLYR
install.packages("dplyr")
library(dplyr)

# INCLUIR OU EXCLUIR E RENOMEAR VARIÁVEIS (COLUNAS)
# Excluir
airbnb <- select(airbnb, -country, -survey_id)

# Renomear
airbnb <- rename(airbnb, country = city, city = borough)

View(airbnb)

# INCLUIR E EXCLUIR OBSERVAÇÕES (LINHAS)
# Vetor de palavras
airbnb_topten <-
c("Brussel", "Antwerpen", "Gent", "Charleroi", "Liege", "Brugge", "Namur", "Leuven", "Mons", "Aalst")
airbnb_topten

# Vetor de números
number_vector <- c(0,2,4,6)
number_vector

# INCLUIR OU EXCLUIR OBSERVAÇÕES COM A FUNÇÃO FILTER
# Instalar pacote Hmisc
install.packages("Hmisc")
library(Hmisc)

# FILTER
airbnb.topten <- filter(airbnb, city %in% airbnb_topten)
View(airbnb.topten)

# OPERADOR PIPE
airbnb <- read.csv("aula1.csv") %>%
  mutate (host_id_c = as.character(host_id), overall_satisfaction_100 = overall_satisfaction * 20)
%>%

```



```

select(-country, -survey_id) %>%
rename(country = city, city = borough) %>%
filter(city %in%
c("Brussel", "Antwerpen", "Gent", "Charleroi", "Liege", "Brugge", "Namur", "Leuven", "Mons", "Aalst"))

# AGRUPAR E SUMARIZAR
# Agrupar e Sumarizar por Cidade
airbnb %>%
  group_by(city) %>%
  summarise(nr_per_city = n())

# COLOCAR EM ORDEM (CRESCENTE)
airbnb %>%
  group_by(city) %>%
  summarise(nr_per_city = n()) %>%
  arrange(nr_per_city)

# COLOCAR EM ORDEM (DESCRESCENTE)
airbnb %>%
  group_by(city) %>%
  summarise(nr_per_city = n()) %>%
  arrange(desc(nr_per_city))

# ESTÁTISTICAS DESCRITIVAS
# Verificar as estatísticas descritivas de preço geral
summary(airbnb$price)

# Preço médio por cidade
airbnb.summary <- airbnb %>%
  group_by(city) %>%
  summarise(nr_per_city = n(), average_price = mean(price)) %>%
  arrange(desc(average_price))
print(airbnb.summary, n = Inf)

# Mediana e preço Máximo
airbnb %>%
  group_by(city) %>%
  summarise(nr_per_city = n(),
            average_price = mean(price),
            median_price = median(price),
            max_price = max(price)) %>%
  arrange(desc(median_price),
            desc(max_price))

# GRÁFICOS
# Instalar pacote Hmisc e ggplot
install.packages("Hmisc")
install.packages("ggplot2")
library(ggplot2)
library(Hmisc)

# Seleccionando as 10 cidades Principais
airbnb_topen <- airbnb %>%
  filter(city %in%
c("Brussel", "Antwerpen", "Gent", "Charleroi", "Liege", "Brugge", "Namur", "Leuven", "Mons", "Aalst"))

# SCATTERPLOT
ggplot(data = airbnb_topen, mapping = aes(x = city, y = price)) +
  geom_point()

```

```
# JITTER
ggplot(data = airbnb_topten, mapping = aes(x = city, y = price)) +
  geom_jitter()

# HISTOGRAMA
ggplot(data = airbnb_topten, mapping = aes(x = price)) +
  geom_histogram()

# PLOTAR A MÉDIA
ggplot(data = airbnb_topten, mapping = aes(x = city, y = price)) +
  geom_jitter() +
  stat_summary(fun.y=median, colour="tomato3", size = 4, geom="point")

# EXPORTAR
# Para exportar em csv
library(readr)
write_excel_csv(airbnb, "airbnb.csv")
```