



Linguagem R

Aula 6

- Objetivo da aula:
- Condicional e looping;
- Exploração dos dados;
- Tratamento e limpeza de dados.

Condicional IF:

IF:

Exemplo:

```
a <- 33
```

```
b <- 200
```

```
if (b > a) {  
  print("b é maior que a")  
}
```

LOOP WHILE:

LOOP WHILE:

Exemplo:

```
i <- 1  
while (i < 6) {  
  print(i)  
  i <- i + 1  
}
```

LOOP FOR:

LOOP FOR:

Exemplo:

```
for (x in 1:10) {  
  print(x)  
}
```

Exploração de Dados:

A linguagem R é amplamente utilizada para explorar e manipular dados.

R facilita o carregamento de DataFrames, reconhece rapidamente suas dimensões, estrutura e propriedades estatísticas.

Explorando um DataFrame:

A seguir vamos realizar uma série de Scripts visando explorar os dados de um DataFrame.

No RStudio carregue o DataFrame vendas.csv

Muita atenção na digitação do 'path' do arquivo no RStudio.
O Windows vai mostrar o path da seguinte maneira:

C:\Users\arquivo

Mas no RStudio tem que ser digitado:

C:/Users/arquivo

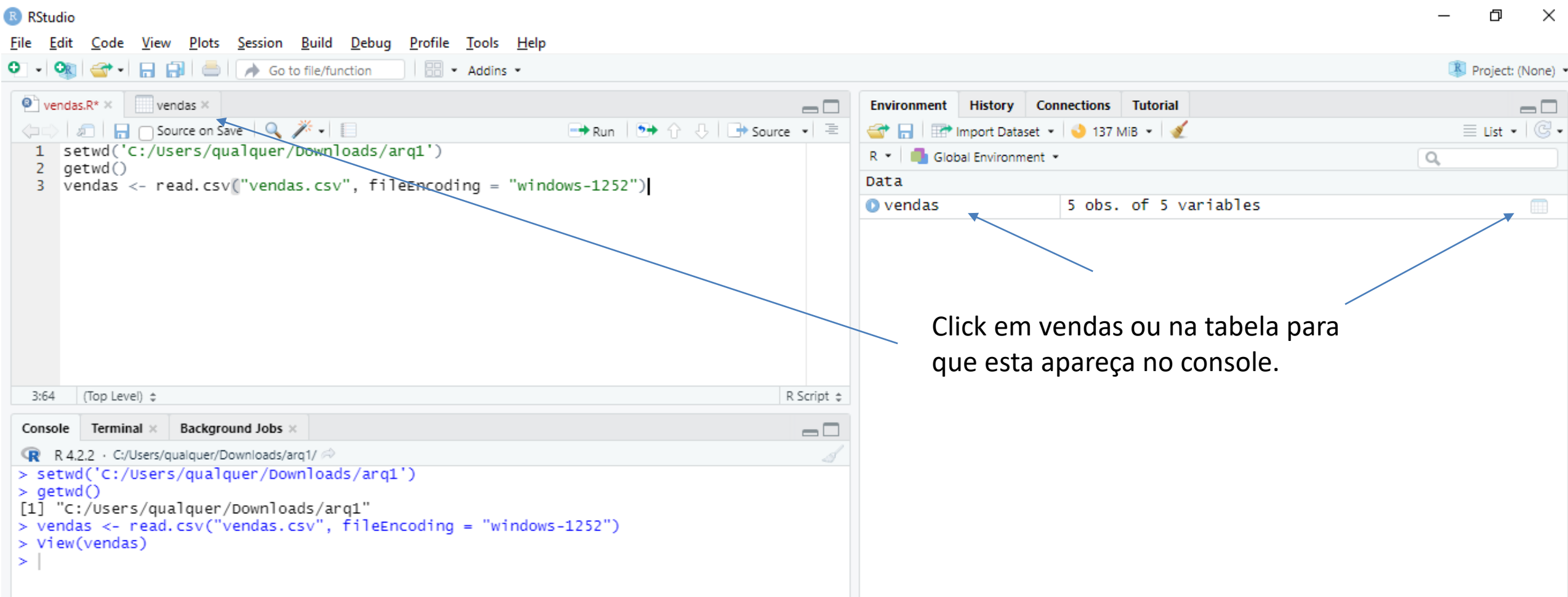
Explorando um DataFrame:

Digite na IDE do RStudio:

```
setwd("path do arquivo")  
getwd()  
vendas <- read.csv("vendas.csv", fileEncoding = "windows-  
1252")
```

Use `head(nome do DataFrame)` para ver as primeiras 5 linhas do DataFrame e `tail(nome do DataFrame)` para ver as últimas 5 linhas.

Explorando um DataFrame:



The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains the R script for loading the data:

```
1 setwd('C:/Users/qualquer/Downloads/arq1')
2 getwd()
3 vendas <- read.csv("vendas.csv", fileEncoding = "windows-1252")
```
- Environment Pane:** Shows the variable `vendas` as a data frame with 5 observations and 5 variables.
- Console:** Shows the execution of the script, confirming the working directory and the successful loading of the `vendas` data frame.

Two blue arrows point from the text instruction to the `vendas` variable in the Environment pane and the data frame table icon.

Click em vendas ou na tabela para que esta apareça no console.

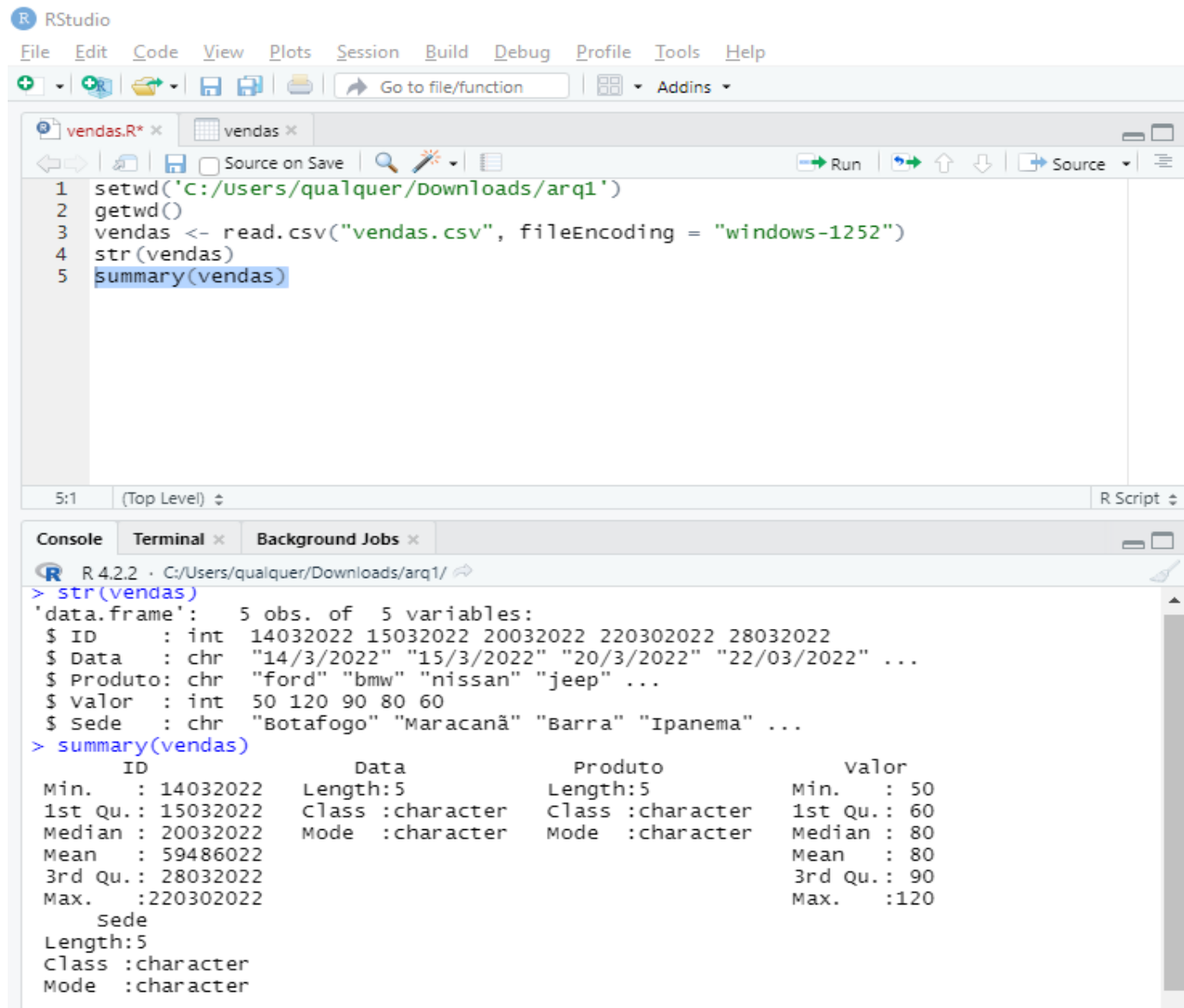
Explorando um DataFrame:

Usa-se a função `str`, para exibir a estrutura básica e os tipos de dados do DataFrame.

```
str(nome do DataFrame)
```

Usa-se a função `summary`, para exibir uma estatística básica dos dados do DataFrame.

```
summary(nome do dataframe)
```



The screenshot shows the RStudio environment. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with icons for file operations and a 'Go to file/function' search bar. The main editor window displays a script named 'vendas.R' with the following code:

```
1 setwd('C:/Users/qualquer/Downloads/arq1')
2 getwd()
3 vendas <- read.csv("vendas.csv", fileEncoding = "windows-1252")
4 str(vendas)
5 summary(vendas)
```

The status bar at the bottom of the editor indicates '5:1 (Top Level)' and 'R Script'.

Below the editor is the Console pane, which shows the output of the executed code:

```
> str(vendas)
'data.frame': 5 obs. of 5 variables:
 $ ID      : int  14032022 15032022 20032022 220302022 28032022
 $ Data    : chr   "14/3/2022" "15/3/2022" "20/3/2022" "22/03/2022" ...
 $ Produto : chr   "ford" "bmw" "nissan" "jeep" ...
 $ valor   : int    50 120 90 80 60
 $ sede    : chr   "Botafogo" "Maracanã" "Barra" "Ipanema" ...

> summary(vendas)
      ID      Data      Produto      valor
Min.   : 14032022 Length:5   Length:5   Min.    : 50
1st Qu.: 15032022 Class :character Class :character 1st Qu.: 60
Median : 20032022 Mode  :character Mode  :character Median : 80
Mean   : 59486022                      Mean   : 80
3rd Qu.: 28032022                      3rd Qu.: 90
Max.   :220302022                      Max.    :120

      sede
Length:5
Class :character
Mode  :character
```

Explorando um DataFrame:

Classificando os dados de uma coluna do DataFrame.

Para classificar um DataFrame em R, usa-se a função: `order()`.

Usa-se:

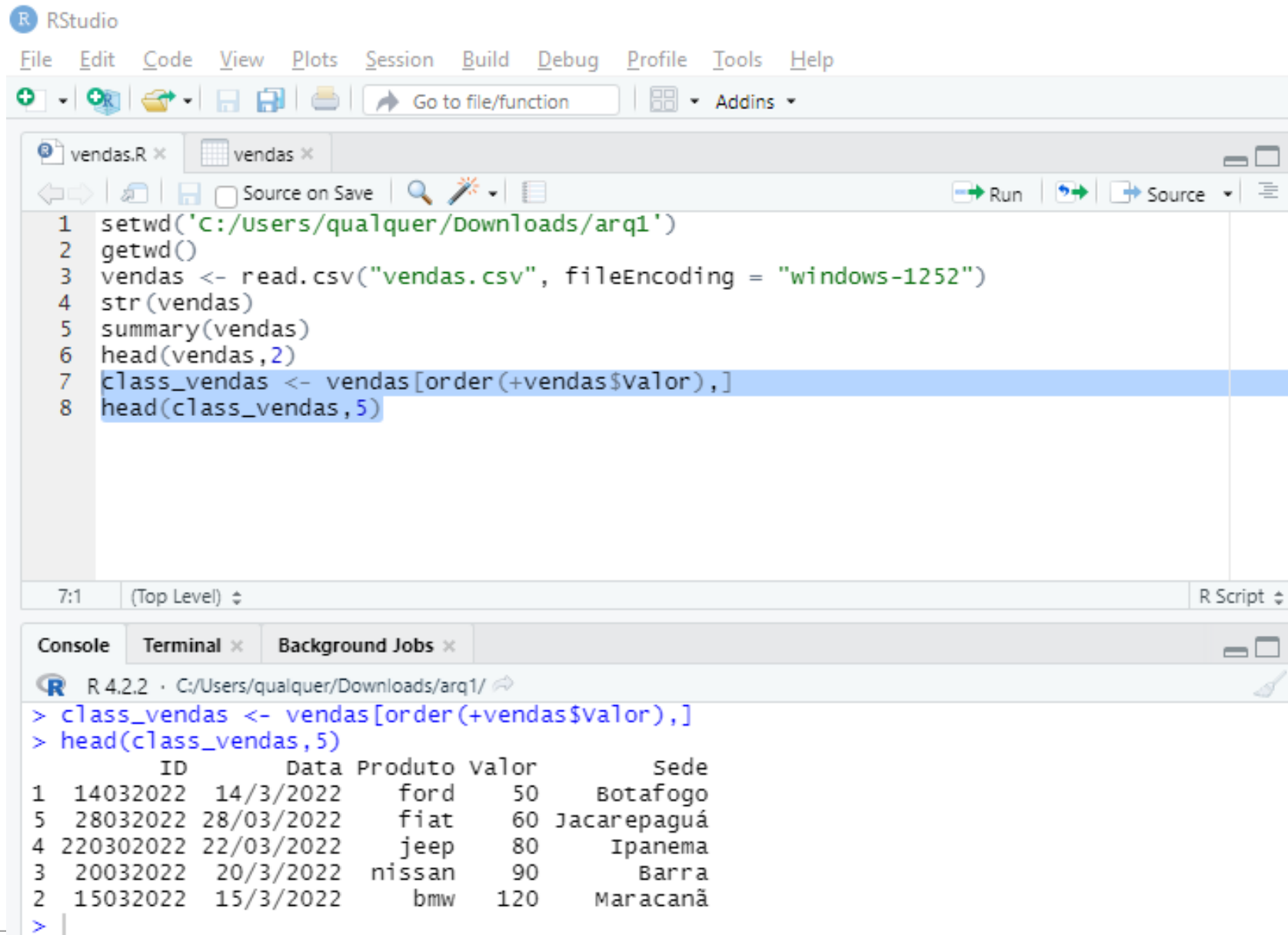
```
nome<- nome_DataFrame[order(- nome_DataFrame $coluna),]
```

```
head(nome,5)
```

para ver as 5 primeiras linhas em ordem decrescente.

Observação: O sinal de + após `order(+...`

irá fornecer uma ordem crescente dos dados da coluna.



Explorando um DataFrame:

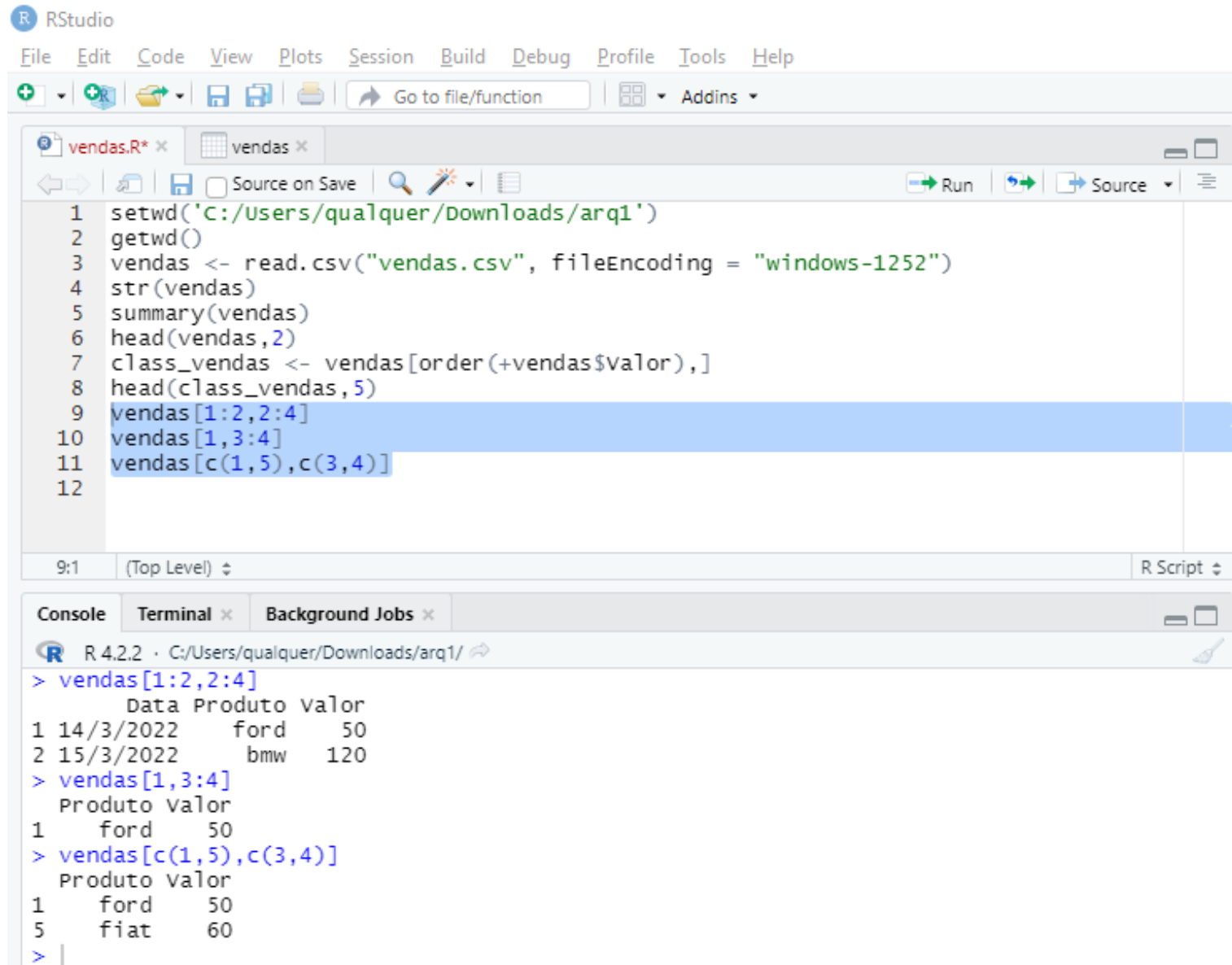
Explorar linhas e colunas individuais em um DataFrame.

Para isto usa-se:

`Nome_DataFrame[n0 da 1ª linha: n0 da última linha, n0 da 1ª coluna:
n0 da última coluna]`

Desejando-se selecionar algumas linhas e colunas usa-se:

`Nome_DataFrame[c(linha 1,linha2,...),c(coluna1,coluna2,...)]`



Explorando um DataFrame:

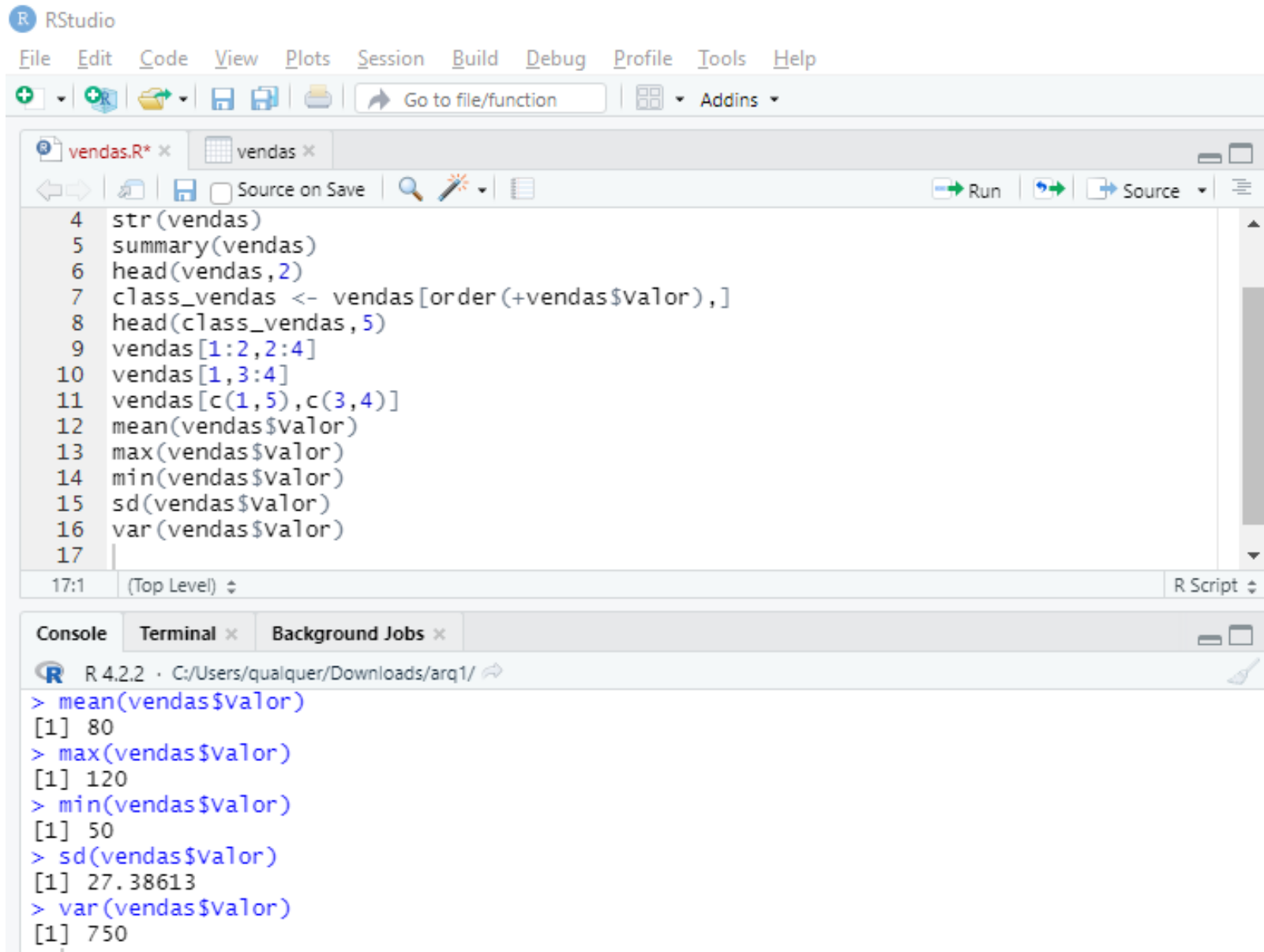
Determinando valores máximo, mínimo, valor médio, desvio padrão e variância de um conjunto de dados.

Estas funções são internas em R, sendo assim usa-se as seguintes funções: `max()`, `min()`, `mean()`, `sd()`, `var()`.

Aplicando Estatística nos Valores de uma Coluna de Um DataFrame:

Para isto usa-se:

```
max(nome do Dataframe$nome da coluna)
min(nome do Dataframe$nome da coluna)
mean(nome do Dataframe$nome da coluna)
sd(nome do Dataframe$nome da coluna)
var(nome do Dataframe$nome da coluna)
```



The image shows the RStudio environment. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with icons for file operations and a search bar labeled 'Go to file/function'. The main editor window displays a script named 'vendas.R' with the following R code:

```
4 str(vendas)
5 summary(vendas)
6 head(vendas, 2)
7 class_vendas <- vendas[order(+vendas$valor),]
8 head(class_vendas, 5)
9 vendas[1:2, 2:4]
10 vendas[1, 3:4]
11 vendas[c(1, 5), c(3, 4)]
12 mean(vendas$valor)
13 max(vendas$valor)
14 min(vendas$valor)
15 sd(vendas$valor)
16 var(vendas$valor)
17
```

The status bar at the bottom of the editor indicates '17:1 (Top Level)' and 'R Script'. Below the editor is a console window with tabs for Console, Terminal, and Background Jobs. The console shows the output of the executed code:

```
> mean(vendas$valor)
[1] 80
> max(vendas$valor)
[1] 120
> min(vendas$valor)
[1] 50
> sd(vendas$valor)
[1] 27.38613
> var(vendas$valor)
[1] 750
```

Manipulando um DataFrame:

Criação de um dataframe para estudo.

Crie um DataFrame com as seguintes colunas e dados:

```
data_frame <- data.frame(  
  colA = c('102E32','204B25','340Z22','901K23'),  
  colB = c('ford','fiat','nissan','cherry'),  
  colC = c(20,30,40,50))
```

A seguir imprima o DataFrame.

```
print (data_frame)
```

Manipulando um DataFrame:

1. Selecionando uma coluna específica do DataFrame.

Para isto usa-se: `(data_frame$coluna desejada)`

Execute o comando selecionando a coluna de marcas do DataFrame.

2. Inserindo uma nova coluna ao DataFrame.

Para isto usa-se: `data_frame["nome nova coluna"]<-c(vetor contendo a mesma quantidade de dados colunares)`

Execute o comando criando a coluna **motor** com os dados EV,EP,IV,IE da coluna do DataFrame.

Resultado:

```
70
71 data_frame <- data.frame(colA = c('102E32','204B25','340Z22','901K23'),
72                             colB = c('ford','fiat','nissan','cherry'),
73                             colC = c(20,30,40,50))
74 print (data_frame)
75
76 col_sel<-(data_frame$colB)
77 print(col_sel)
78
79 data_frame['motor']<- c('EV','EP','IV','IE')
80 print(data_frame)
81
82
```

71:1 (Top Level) ↕

Console

Terminal x

Background Jobs x



R 4.2.2 · ~/

```
+                                     colB = c( ford , fiat , nissan , cherry ),
+                                     colC = c(20,30,40,50))
> print (data_frame)
  colA  colB colC
1 102E32  ford   20
2 204B25  fiat   30
3 340Z22 nissan   40
4 901K23 cherry  50
>
> col_sel<-(data_frame$colB)
> print(col_sel)
[1] "ford"  "fiat"  "nissan" "cherry"
>
> data_frame['motor']<- c('EV','EP','IV','IE')
> print(data_frame)
  colA  colB colC motor
1 102E32  ford   20   EV
2 204B25  fiat   30   EP
3 340Z22 nissan   40   IV
4 901K23 cherry  50   IE
```

Manipulando um DataFrame:

3. Realizando operações com duas colunas do DataFrame.

Insira mais uma coluna no dataframe.

Para isto use o seguinte:

```
data_frame["colD"]<-c( 0.02,0.05,0.02,0.03)
```

A seguir crie uma nova coluna chama **valor** que seja o resultado da multiplicação da coluna C pela coluna D.

Para isto usa-se:

```
data_frame["valor"]<- (valor$colC)*(valor$colD)
```

Resultado:

```
80 print(data_frame)
81
82 data_frame["colD"]<-c( 0.02,0.05,0.02,0.03)
83 print(data_frame)
84
85 data_frame["valor"]<- ((data_frame$colC)*(data_frame$colD))
86 print(data_frame)
87
88
89
```

82:1

(Top Level) ↕

Console

Terminal ×

Background Jobs ×

R 4.2.2 · ~/

```
> data_frame["colD"]<-c( 0.02,0.05,0.02,0.03)
> print(data_frame)
  colA  colB colC motor colD valor
1 102E32  ford  20   EV 0.02  0.4
2 204B25  fiat  30   EP 0.05  1.5
3 340Z22 nissan  40   IV 0.02  0.8
4 901K23 cherry 50   IE 0.03  1.5
>
> data_frame["valor"]<- ((data_frame$colC)*(data_frame$colD))
> print(data_frame)
  colA  colB colC motor colD valor
1 102E32  ford  20   EV 0.02  0.4
2 204B25  fiat  30   EP 0.05  1.5
3 340Z22 nissan  40   IV 0.02  0.8
4 901K23 cherry 50   IE 0.03  1.5
> |
```

Manipulando um DataFrame:

4. Exibindo os registros (linhas) que contenham um valor específico.

Para isto usa-se:

```
data_frame[grep(pattern = "atributo", x = data_frame$coluna),]
```

Por exemplo: Para selecionar a linha com o atributo ford use:

```
data_frame[grep(pattern = "ford", x = data_frame$colB),]
```


Manipulando um DataFrame:

5. Aplicando um condicional numa linha:

Por exemplo: Deseja-se acessar as marcas que tem valor maior ou igual que 1.

Para isto usa-se:

```
data_frame[(data_frame$valor) >= 1, c("colB","valor")]
```

Manipulando um DataFrame:

6. Alterando as legendas das colunas.

Para isto usa-se:

```
colnames(data_frame) <- c("new colA", "new colB",...)
```

Por exemplo: Altere o nome da coluna A para ID e o nome da coluna B para Marcas.

Observação: As colunas que não tiveram o nome alterado devem ser digitadas novamente.

```
colnames(data_frame) <-  
c("ID", "Marcas", "colC", "motor", "colD", "valor")  
print(data_frame)
```

Manipulando um DataFrame:

7. Adicionando linhas com `rbind()`.

Para isto define-se uma nova linha:

```
new_row = c("objeto", "objeto", valor, ...)
```

E usa-se a função `rbind()`:

```
data_frame <- rbind(data_frame, new_row)
```

Por exemplo: Adicione a linha com os seguintes dados:

"203E21", "hyundai", 20, "EP", 0.05, 1.0 ao DataFrame.

Resultado:

```
97
98 new_row = c("203E21","hyundai",20,"EP",0.05,1.0) |
99 data_frame <- rbind(data_frame,new_row)
100 print(data_frame)
101
102
```

98:50 (Top Level) ↕

Console

Terminal ×

Background Jobs ×

R 4.2.2 · ~/

```
> print(data_frame)
      ID Marcas colC motor colD valor
1 102E32  ford   20    EV 0.02   0.4
2 204B25  fiat   30    EP 0.05   1.5
3 340Z22 nissan   40    IV 0.02   0.8
4 901K23 cherry  50    IE 0.03   1.5
> new_row = c("203E21","hyundai",20,"EP",0.05,1.0)
> data_frame <- rbind(data_frame,new_row)
> print(data_frame)
      ID Marcas colC motor colD valor
1 102E32  ford   20    EV 0.02   0.4
2 204B25  fiat   30    EP 0.05   1.5
3 340Z22 nissan   40    IV 0.02   0.8
4 901K23 cherry  50    IE 0.03   1.5
5 203E21 hyundai  20    EP 0.05    1
> |
```

Manipulando um DataFrame:

8. Verificando se existem valores ausentes do DataFrame.

Para isto usa-se:

```
data_frame <- data_frame[is.na(data_frame$coluna),]  
print(data_frame)
```