# Desafio 7

## 2025-09-18

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```r
library(RSQLite)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----------------------- tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ----------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
db <- dbConnect(SQLite(),
"//smb/ra254672/Downloads/discocopy.db")
```

## Including Plots

You can also embed plots, for example:

```r
dbListTables(db)
```

```
##  [1] "albums"          "artists"         "customers"       "employees"
##  [5] "genres"          "inst"            "instr"           "instrument"
##  [9] "instrumentos"    "instruments"     "invoice_items"   "invoices"
## [13] "media_types"     "mtcars"          "playlist_track"  "playlists"
## [17] "sqlite_sequence" "sqlite_stat1"    "tracks"
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```r
dbExecute(db,
"CREATE TABLE instru
(AlbumId INTEGER,
TrackId INTEGER,
ElectricGuitar INTEGER,
Singer INTEGER,
Trumpet INTEGER)", overwrite = TRUE)
```

```
## [1] 0
```

```
"dbExecute(db,
DROP TABLE i"
```

```
## [1] "dbExecute(db,\nDROP TABLE i"
```

```
dbListTables(db)
```

```
##  [1] "albums"          "artists"         "customers"       "employees"
##  [5] "genres"          "inst"            "instr"           "instru"
##  [9] "instrument"      "instrumentos"    "instruments"     "invoice_items"
## [13] "invoices"        "media_types"     "mtcars"          "playlist_track"
## [17] "playlists"       "sqlite_sequence" "sqlite_stat1"    "tracks"
```

```
aname = "Gilberto Gil"
sql = paste0("SELECT ArtistId FROM artists "
,
"WHERE Name = '"
, aname,
"'")
aId = dbGetQuery(db, sql)
sql = paste('SELECT Title FROM albums'
,
'WHERE ArtistId ='
, aId)
dbGetQuery(db, sql)
```

```
##                                  Title
## 1             As Canções de Eu Tu Eles
## 2            Quanta Gente Veio Ver (Live)
## 3 Quanta Gente Veio ver--Bônus De Carnaval
```

```
sql = paste("SELECT ArtistId FROM artists"
,
"WHERE Name = ?")
query <- dbSendQuery(db, sql)
dbBind(query, list("Gilberto Gil"))
aId <- dbFetch(query)
dbClearResult(query)
# Segundo passo interno, não deve causar problema
sql = paste('SELECT Title FROM albums'
,
'WHERE ArtistId ='
, aId)
dbGetQuery(db, sql)
```

```
##                                  Title
## 1             As Canções de Eu Tu Eles
## 2            Quanta Gente Veio Ver (Live)
## 3 Quanta Gente Veio ver--Bônus De Carnaval
```

```
dbListFields(db,'instruments')
```

```
## [1] "AlbumId"       "TrackId"       "ElectricGuitar" "Singer"
## [5] "Trumpet"
```

```
sql = paste('SELECT TrackId, Name FROM tracks'
,
```

```
'WHERE AlbumId = 85')
dbGetQuery(db, sql) %>% head
```

```
##   TrackId                Name
## 1    1073 Óia Eu Aqui De Novo
## 2    1074      Baião Da Penha
## 3    1075 Esperando Na Janela
## 4    1076            Juazeiro
## 5    1077 Último Pau-De-Arara
## 6    1078          Asa Branca
```

```
dbExecute(db,
"INSERT INTO instruments
VALUES ('85'
,
'1075'
, 0, 1, 0),
('85'
,
'1078'
, 0, 1, 0); ")
```

```
## [1] 2
```

```
dbGetQuery(db,
"SELECT * FROM instruments")
```

```
##   AlbumId TrackId ElectricGuitar Singer Trumpet
## 1      85    1075              0      1       0
## 2      85    1078              0      1       0
## 3      85    1075              0      1       0
## 4      85    1078              0      1       0
## 5      85    1075              0      1       0
## 6      85    1078              0      1       0
## 7      85    1075              0      1       0
## 8      85    1078              0      1       0
```

```
dbWriteTable(db,
             "mtcars",
             mtcars,
             overwrite = TRUE)
```

```
dbListTables(db)
```

```
##  [1] "albums"         "artists"         "customers"       "employees"
##  [5] "genres"         "inst"            "instr"           "instru"
##  [9] "instrument"     "instrumentos"    "instruments"     "invoice_items"
## [13] "invoices"       "media_types"     "mtcars"          "playlist_track"
## [17] "playlists"      "sqlite_sequence" "sqlite_stat1"    "tracks"
```

```
dbGetQuery(db,
"SELECT * FROM mtcars") %>% head(3)
```

```
##    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## 1 21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## 2 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## 3 22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
```

```r
theAvgCar <- mtcars %>%
summarise_all(function(x) round(mean(x), 2))
theAvgCar
```

```
##      mpg  cyl   disp     hp drat   wt  qsec   vs   am gear carb
## 1 20.09 6.19 230.72 146.69  3.6 3.22 17.85 0.44 0.41 3.69 2.81
```

```r
dbWriteTable(db,
"mtcars"
, theAvgCar, append = TRUE)
dbGetQuery(db,
"SELECT * FROM mtcars") %>% tail(3)
```

```
##       mpg  cyl   disp     hp drat   wt  qsec   vs   am gear carb
## 31 15.00 8.00 301.00 335.00 3.54 3.57 14.60 0.00 1.00 5.00 8.00
## 32 21.40 4.00 121.00 109.00 4.11 2.78 18.60 1.00 1.00 4.00 2.00
## 33 20.09 6.19 230.72 146.69 3.60 3.22 17.85 0.44 0.41 3.69 2.81
```

```r
dbWriteTable(db,
"mtcars"
, mtcars, overwrite = TRUE)
dbGetQuery(db,
"SELECT * FROM mtcars") %>% tail(3)
```

```
##     mpg cyl disp  hp drat   wt qsec vs am gear carb
## 30 19.7   6  145 175 3.62 2.77 15.5  0  1    5    6
## 31 15.0   8  301 335 3.54 3.57 14.6  0  1    5    8
## 32 21.4   4  121 109 4.11 2.78 18.6  1  1    4    2
```

```r
res <- dbSendQuery(db,
"SELECT * FROM mtcars WHERE cyl = 4")
while(!dbHasCompleted(res)){
chunk <- dbFetch(res, n = 5)
print(nrow(chunk))
}
```

```
## [1] 5
## [1] 5
## [1] 1
```

```r
dbDisconnect(db)
```

```
## Warning in connection_release(conn@ptr): There are 1 result in use. The
## connection will be released when they are closed
```

```r
if("discoCopy.db" %in% list.files("//smb/ra254672/Downloads/discocopy.db")){
file.remove("//smb/ra254672/Downloads/discocopy.db")
}
```

```r
{r

library(RSQLite)
library(tidyverse)
library(dbplyr)
```

```
##
## Attaching package: 'dbplyr'
```

```
## The following objects are masked from 'package:dplyr':
```

```
## 
##     ident, sql
db <- dbConnect(SQLite(),
"//smb/ra254672/Downloads/discocopy.db") # original
tracks <- tbl(db,
"tracks") # dplyr
tracks %>% head(3)
```

```
## # Source:   SQL [3 x 9]
## # Database: sqlite 3.46.0 [\\smb\ra254672\Downloads\discocopy.db]
##   TrackId Name          AlbumId MediaTypeId GenreId Composer Milliseconds  Bytes
##     <int> <chr>           <int>       <int>   <int> <chr>          <int>  <int>
## 1       1 For Those Ab~       1           1       1 Angus Y~      343719 1.12e7
## 2       2 Balls to the~       2           2       1 <NA>         342562 5.51e6
## 3       3 Fast As a Sh~       3           2       1 F. Balt~     230619 3.99e6
## # i 1 more variable: UnitPrice <dbl>
```

```
meanTracks <- tracks %>%
group_by(AlbumId) %>%
summarise(AvLen = mean(Milliseconds, na.rm = TRUE),
AvCost = mean(UnitPrice, na.rm = TRUE))
meanTracks
```

```
## # Source:   SQL [?? x 3]
## # Database: sqlite 3.46.0 [\\smb\ra254672\Downloads\discocopy.db]
##    AlbumId   AvLen AvCost
##      <int>   <dbl>  <dbl>
## 1        1 240042.   0.99
## 2        2 342562    0.99
## 3        3 286029.   0.99
## 4        4 306657.   0.99
## 5        5 294114.   0.99
## 6        6 265456.   0.99
## 7        7 270780.   0.99
## 8        8 207638.   0.99
## 9        9 333926.   0.99
## 10      10 280551.   0.99
## # i more rows
```

```
meanTracks %>% show_query()
```

```
## <SQL>
## SELECT `AlbumId`, AVG(`Milliseconds`) AS `AvLen`, AVG(`UnitPrice`) AS `AvCost`
## FROM `tracks`
## GROUP BY `AlbumId`
```

```
mT <- meanTracks %>% collect()
mT
```

```
## # A tibble: 347 x 3
##    AlbumId   AvLen AvCost
##      <int>   <dbl>  <dbl>
## 1        1 240042.   0.99
## 2        2 342562    0.99
## 3        3 286029.   0.99
## 4        4 306657.   0.99
```

```
##  5         5 294114.    0.99
##  6         6 265456.    0.99
##  7         7 270780.    0.99
##  8         8 207638.    0.99
##  9         9 333926.    0.99
## 10        10 280551.    0.99
## # i 337 more rows
```

```r
dbDisconnect(db)
```