Desafio 13

2025-10-16

```
# --------- Pacotes ----------
library(DBI)
```

```
## Warning: pacote 'DBI' foi compilado no R versão 4.5.1
```

```
library(RSQLite)
```

```
## Warning: pacote 'RSQLite' foi compilado no R versão 4.5.1
```

```
library(readr)
library(dplyr)
```

```
##
## Anexando pacote: 'dplyr'
```

```
## Os seguintes objetos são mascarados por 'package:stats':
##
##     filter, lag
```

```
## Os seguintes objetos são mascarados por 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# caminhos exatos dos seus arquivos
arq_basics     <- "C:/Users/pepem/OneDrive/Documentos/Dados_Filmes/title.basics0.tsv"
arq_principals <- "C:/Users/pepem/OneDrive/Documentos/Dados_Filmes/title.principals0.tsv"
arq_ratings    <- "C:/Users/pepem/OneDrive/Documentos/Dados_Filmes/title.ratings.tsv"

# onde salvar o banco SQLite
arq_sqlite <- "C:/Users/pepem/OneDrive/Documentos/Dados_Filmes/movies.sqlite3"
if (file.exists(arq_sqlite)) file.remove(arq_sqlite)
```

```
## Warning in file.remove(arq_sqlite): não foi possível remover o arquivo
## 'C:/Users/pepem/OneDrive/Documentos/Dados_Filmes/movies.sqlite3', motivo
## 'Permission denied'
```

```
## [1] FALSE
```

```
con <- dbConnect(SQLite(), arq_sqlite)
```

```
basics <- read_tsv(
  arq_basics,
  na = c("", "NA", "\\N"),
  col_types = cols(
    tconst        = col_character(),
    titleType     = col_character(),
    primaryTitle  = col_character(),
    originalTitle = col_character(),
    isAdult       = col_integer(),
    startYear     = col_character(),
    endYear       = col_character(),
    runtimeMinutes= col_character(),
    genres        = col_character()
  ),
  progress = TRUE
)
```

```
## indexing title.basics0.tsv [--------------------------------] 2.15GB/s, eta:  0sindexing title.basics
```

```
principals <- read_tsv(
  arq_principals,
  na = c("", "NA", "\\N"),
  col_types = cols(
    tconst     = col_character(),
    nconst     = col_character(),
    category   = col_character(),
    job        = col_character(),
    characters = col_character()
  ),
  progress = TRUE
)
```

```
## indexing title.principals0.tsv [----------------------------] 2.15GB/s, eta:  0sindexing title.princi
```

```
ratings <- read_tsv(
  arq_ratings,
  na = c("", "NA", "\\N"),
  col_types = cols(
    tconst        = col_character(),
    averageRating = col_double(),
    numVotes      = col_double()
  ),
  progress = TRUE
)
```

```
## indexing title.ratings.tsv [-------------------------------] 126.99MB/s, eta:  0sindexing title.rating
```

```
dbWriteTable(con, "basics", basics, overwrite = TRUE)
dbWriteTable(con, "ratings", ratings, overwrite = TRUE)
dbWriteTable(con, "principals", principals, overwrite = TRUE)

dbExecute(con, "CREATE INDEX idx_b_tconst ON basics(tconst)")
```

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

```
dbExecute(con, "CREATE INDEX idx_r_tconst ON ratings(tconst)")
```

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

```
dbExecute(con, "CREATE INDEX idx_p_tconst ON principals(tconst)")
```

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

```
dbExecute(con, "CREATE INDEX idx_b_type   ON basics(titleType)")
```

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

```
dbExecute(con, "CREATE INDEX idx_r_score  ON ratings(averageRating DESC, numVotes DESC)")
```

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

```
"Top-5 filmes por nota"
```

```
## [1] "Top-5 filmes por nota"
```

```
sql_top5 <- "
SELECT
  b.tconst,
  b.primaryTitle AS titulo,
  b.startYear    AS ano,
  r.averageRating AS nota,
  r.numVotes      AS votos
FROM ratings r
JOIN basics  b ON b.tconst = r.tconst
WHERE b.titleType = 'movie' AND (b.isAdult IS NULL OR b.isAdult = 0)
ORDER BY r.averageRating DESC, r.numVotes DESC
LIMIT 5;
"
top5 <- dbGetQuery(con, sql_top5)
top5
```

```
##       tconst                   titulo  ano nota votos
## 1 tt33075815                   Kaveri 2024   10  1023
## 2 tt27815015                  Kurukku 2024   10   451
## 3 tt28450376 Jedal Dar Omghe 30 Metri 2009   10   142
## 4 tt27859713                Sargashte 2016   10   134
## 5 tt33507675          Gorgeous Rascal 2024   10   115
```

```
"Gênero mais frequente entre filmes com nota > 8"
```

```
## [1] "Gênero mais frequente entre filmes com nota > 8"
```

```
sql_genero <- "
WITH filmes8 AS (
  SELECT b.tconst, b.genres
  FROM basics b
  JOIN ratings r ON r.tconst = b.tconst
  WHERE b.titleType = 'movie'
    AND (b.isAdult IS NULL OR b.isAdult = 0)
    AND r.averageRating > 8
    AND b.genres IS NOT NULL
),
split AS (
  SELECT
    tconst,
    TRIM(SUBSTR(genres, 1, COALESCE(NULLIF(INSTR(genres, ','),0)-1, LENGTH(genres)))) AS genre,
    CASE WHEN INSTR(genres, ',') = 0 THEN NULL
         ELSE SUBSTR(genres, INSTR(genres, ',')+1) END AS rest
  FROM filmes8
  UNION ALL
  SELECT
    tconst,
    TRIM(SUBSTR(rest, 1, COALESCE(NULLIF(INSTR(rest, ','),0)-1, LENGTH(rest)))) AS genre,
    CASE WHEN rest IS NULL OR INSTR(rest, ',')=0 THEN NULL
         ELSE SUBSTR(rest, INSTR(rest, ',')+1) END AS rest
  FROM split
  WHERE rest IS NOT NULL
)
SELECT genre, COUNT(*) AS qtd
FROM split
WHERE genre IS NOT NULL AND genre <> '\\N'
GROUP BY genre
ORDER BY qtd DESC
LIMIT 1;
"
genero_mais_freq <- dbGetQuery(con, sql_genero)
genero_mais_freq
```

```
##         genre   qtd
## 1 Documentary 10563
```

```
"Top-3 atores/atrizes em filmes com nota > 7.5"
```

```
## [1] "Top-3 atores/atrizes em filmes com nota > 7.5"
```

```
sql_atores <- "
SELECT
  p.nconst,
  COUNT(*) AS qtd_filmes
```

```
FROM principals p
JOIN basics b  ON b.tconst = p.tconst
JOIN ratings r ON r.tconst = p.tconst
WHERE b.titleType = 'movie'
  AND (b.isAdult IS NULL OR b.isAdult = 0)
  AND r.averageRating > 7.5
  AND p.category IN ('actor','actress')
GROUP BY p.nconst
ORDER BY qtd_filmes DESC
LIMIT 3;
"
top_atores <- dbGetQuery(con, sql_atores)
top_atores
```

```
##      nconst qtd_filmes
## 1 nm0004660        231
## 2 nm0595934        155
## 3 nm3183374        124
```

```
"Fechar conexão"
```

```
## [1] "Fechar conexão"
```

```
dbDisconnect(con)
```