Atividade de Avaliação 3

Benilton Carvalho

3 e 4 de Novembro de 2021

O Conjunto de Dados

O arquivo `indice_bigmac.zip` está disponível neste link (clique aqui). Para baixar o arquivo, você deverá estar logado em sua conta UNICAMP. Ao descompactar o arquivo, você encontrará 59 arquivos nomeados com o seguinte padrão:

`indice_bigmac_<SIGLA_PAIS>_<NOME_PAIS>_<SIGLA_MOEDA>.csv`

Por exemplo, para o arquivo:

`indice_bigmac_BRA_Brazil_BRL.csv`

temos que o conteúdo do mesmo refere-se ao Brasil (sigla: BRA) com valores locais em *Reais do Brasil* (sigla: BRL).

Dentro de cada arquivo, você encontrará as seguintes colunas (que podem estar dispostas em ordem diferente e grafadas em tipos, maiúsculas/minúsculas/etc, também diferentes):

variável	definição	fonte
date	Data da observação (YYYY-MM-DD)	
iso_a3	Código do país [padrão ISO 3166-1]	
currency_code	Código da moeda [padrão ISO 4217]	
name	Nome do país	
local_price	Preço de um Big Mac na moeda local	McDonalds; *The Economist*
dollar_ex	Cotação do dólar (americano) na moeda local	*Reuters*
dollar_price	Preço de um Big Mac em dólares (americanos)	
USD_raw	Índice não ajustado, relativo ao Dólar Americano	
EUR_raw	Índice não ajustado, relativo ao Euro	
GBP_raw	Índice não ajustado, relativo à Libra Britânica	
JPY_raw	Índice não ajustado, relativo ao Yen Japonês	
CNY_raw	Índice não ajustado, relativo ao Yuan Chinês	

Objetivo

- Processar bases de dados heterogêneas distribuídas em múltiplos arquivos;
- Criar bases de dados em SQLite;
- Consultar bases de dados em SQLite;
- Combinar SQLite com processamento de dados via `tidyverse`;
- Produzir visualização de dados simples;
- Gerar um documento reprodutível por outros usuários em suas máquinas locais.

Entregável

Envie, por meio do Moodle, o arquivo R ou Rmd que você criou. Esta é uma tarefa do tipo *peer review* e, assim, você receberá tarefas de 3 outros colegas para avaliação e execução (i.e., você deverá executar o código do seu colega e certificar-se de que o código gera os resultados esperados).

Importante

1. Esta é uma atividade que deve ser resolvida individualmente.
2. Espera-se que o código enviado esteja bem organizado, bem comentado e seja reprodutível na máquina do corretor.
3. Não envie de volta os arquivos de dados disponibilizados para você.
4. Só serão aceitas atividades enviadas por meio do Moodle.
5. O prazo máximo para a entrega é 19:00 do dia 4 de novembro. Envios atrasados não serão aceitos.
6. Para evitar transtornos, procure enviar a solução até às 18:55 do dia 4 de novembro.
7. Observe que espera-se que esta atividade seja resolvida entre 1h30min e 1h45min de trabalho. Apesar de a atividade ficar disponível por cerca de 33h, isto não implica que se trata de uma atividade com esta duração. O prazo de 33h é concedido de forma a permitir ao aluno que se organize adequadamente (i.e., descanse apropriadamente, alimente-se, participe das atividades didáticas das disciplinas em que se encontra matriculado) e disponha de cerca de 1h30min-1h45min para a realização da atividade.

Atividade

1. Você deverá criar uma função denominada `processaDados` que desempenhará as seguintes funções e terá as seguintes características:
   a. Receberá como único argumento a pasta/diretório onde estão gravados os 59 arquivos descompactados;
   b. Processará cada um dos arquivos apropriadamente, depositando os resultados numa base de dados SQL, chamada `bigmac.sqlite3` e gravada na mesma pasta/diretório em que estão os arquivos CSV supracitados;
   c. O banco de dados SQL deverá conter uma tabela chamada `ibm` (Índice Big Mac), que terá as colunas:
      i. `data`: data da observação;
      ii. `cod_pais`: código do país;
      iii. `cod_moeda`: código da moeda;
      iv. `nome_pais`: nome do país;
      v. `preco_local`: preço de um Big Mac em moeda local;
      vi. `preco_dolar`: preço de um Big Mac em Dólar Americano;
      vii. `cotacao`: cotação do Dólar Americano em moeda local;
      viii. `indice_dolar`: índice não ajustado, relativo ao Dólar Americano;
   d. O banco de dados SQL deverá conter todas as observações de todos os países e todas moedas;
   e. A função deverá retornar ao usuário uma conexão para o banco de dados recém-criado.

```
processaDados = function(directory){
  ## argumento 'directory' refere-se à pasta/dir onde estão os arquivos a serem processados

  ## determina o local e nome do banco de dados a ser criado
  ## abaixo, força-se a criação da bd no dir em que estão os arquivos de origem
  bdfile = file.path(directory, "bigmac.sqlite3")

  ## a função não deve ficar adicionando registros repetidos na base
  ## então, testa-se se a base já existe; se sim, remove a base
  ## (pq a função vai criar de novo)
  ## se não existe a base, segue normalmente
  if (file.exists(bdfile))
    file.remove(bdfile)

  ## identifica que arquivos devem ser processados
  ## (apenas os que começam com 'indice_bigmac' e terminam com '.csv')
  fns = list.files(directory, full.names = TRUE, pattern = '^indice\\_bigmac\\_.*\\.csv$')

  ## carregam-se os pacotes necessários
  library(RSQLite)
  library(tidyverse)

  ## conexao deve ser feita com driver SQLite() e a base deve ser criada na pasta/dir recebida
  ## acima, com nome completo especificado em 'bdfile'
  conexao = dbConnect(SQLite(), bdfile)

  ## repete para cada arquivo de interesse encontrado em fns
  for (i in 1:length(fns)){
    ## extrai o nome do arquivo
    this_file = fns[i]

    ## o nome do arquivo tem o caminho também, então é preciso usar basename()
    ## para obter apenas o nome do arquivo (e ignorar o nome da pasta/dir)
```

```
    ## com o nome do arquivo, remove-se a extensão e separa em cada "_"
    nomes = str_split(str_remove(basename(this_file), "\\.csv"), "_", simplify = TRUE)

    ## cada pedaço separado contem informações relevantes: cod pais, pais, cod moeda
    cod_pais = nomes[1, 3]
    cod_moeda = nomes[1, 5]
    nome_pais = nomes[1, 4]

    ## lê o arquivo csv usando readr, sem mostrar as informações de coluna,
    ## pq será repetido 59 vezes
    input = read_csv(this_file, show_col_types = FALSE)

    ## da entrada lida, renomeiam-se as colunas
    ## (todas para minusculas, foi a minha escolha)
    ## ao fazer o transmute, já determina-se a sequ&encia das variáveis a serem gravadas no bd
    ## será sempre a mesma sequência.
    ## o campo data precisa ser character, para facilitar conversão, quando necessário
    ## o codigo do pais, nome do pais e codigo da moeda foram obtidos a partir do nome do arquivo
    ## renomeiam-se também as colunas para preco_local, preco_dolar, cotacao e indice_dolar
    input = input %>%
      rename_with(str_to_lower) %>%
      transmute(data = as.character(date),
                cod_pais = cod_pais,
                cod_moeda = cod_moeda,
                nome_pais = nome_pais,
                preco_local = local_price,
                preco_dolar = dollar_price,
                cotacao = dollar_ex,
                indice_dolar = usd_raw)
    dbWriteTable(conexao, "ibm", input, append = TRUE)
  }
  conexao
}
```

Por exemplo, assumindo que os arquivos estejam dentro da pasta/diretório `~/Downloads/bigmac/untitled folder/indice_bigmac`, então este seria o comportamento esperado da função:

```
con = processaDados("~/Downloads/bigmac/untitled folder/indice_bigmac")
con
```

```
## <SQLiteConnection>
##   Path: /Users/benilton/Downloads/bigmac/untitled folder/indice_bigmac/bigmac.sqlite3
##   Extensions: TRUE
```

```
## tabelas criadas
dbListTables(con)
```

```
## [1] "ibm"
```

```
## colunas na tabela
dbListFields(con, "ibm")
```

```
## [1] "data"         "cod_pais"     "cod_moeda"    "nome_pais"    "preco_local"
## [6] "preco_dolar"  "cotacao"      "indice_dolar"
```

```
## podem haver casos com 57 países pelo fato de renomear UAE
## soluções com 57 países, por consequência de ajustar o nome do
```

```
## país, não devem ser penalizadas
dbGetQuery(con, "SELECT COUNT(DISTINCT nome_pais) AS npaises FROM ibm")
```

```
##   npaises
## 1      58
```

```
## número total de registros depositados
dbGetQuery(con, "SELECT COUNT(*) AS nregistros FROM ibm")
```

```
##   nregistros
## 1       1520
```

```
## valores médios de colunas numéricas
## valores serão diferentes se as colunas não
##    foram ordenadas corretamente e isso anulará a questão)
sql = paste(
  "SELECT nome_pais, AVG(preco_local) AS preco_medio,",
  "AVG(preco_dolar) AS preco_medio_dolar,",
  "AVG(cotacao) AS cotacao_media,",
  "AVG(indice_dolar) AS indice_dolar_medio",
  "FROM ibm GROUP BY nome_pais"
)
dbGetQuery(con, sql)
```

```
##              nome_pais  preco_medio preco_medio_dolar cotacao_media
## 1            Argentina 5.609886e+01          2.964523  1.707033e+01
## 2            Australia 4.724714e+00          3.757364  1.302365e+00
## 3           Azerbaijan 3.950000e+00          2.325829  1.698321e+00
## 4              Bahrain 1.371429e+00          3.634616  3.773714e-01
## 5               Brazil 1.157771e+01          4.094811  2.836000e+00
## 6              Britain 2.615143e+00          3.947897  6.605021e-01
## 7               Canada 5.027714e+00          4.164800  1.225979e+00
## 8                Chile 2.021571e+03          3.361148  6.027094e+02
## 9                China 1.561714e+01          2.295125  7.004914e+00
## 10            Colombia 9.073333e+03          3.664227  2.541800e+03
## 11          Costa Rica 2.002414e+03          3.674376  5.421995e+02
## 12             Croatia 2.214286e+01          3.420706  6.478600e+00
## 13      Czech Republic 6.969686e+01          3.139786  2.327024e+01
## 14             Denmark 2.905800e+01          4.715777  6.273559e+00
## 15               Egypt 2.118156e+01          2.163232  9.704963e+00
## 16           Euro area 3.491005e+00          4.223122  8.418632e-01
## 17           Guatemala 2.514286e+01          3.269246  7.691514e+00
## 18            Honduras 8.628571e+01          3.553368  2.428680e+01
## 19           Hong Kong 1.625714e+01          2.089194  7.784091e+00
## 20             Hungary 7.410857e+02          3.070137  2.460097e+02
## 21               India 1.417857e+02          2.175841  6.403417e+01
## 22           Indonesia 2.478911e+04          2.180917  1.123064e+04
## 23              Israel 1.617308e+01          4.403190  3.716542e+00
## 24               Japan 3.301143e+02          3.138230  1.070324e+02
## 25              Jordan 2.175714e+00          3.067795  7.092357e-01
## 26              Kuwait 1.121429e+00          3.693106  3.036857e-01
## 27             Lebanon 1.250000e+04          3.803870  5.471571e+03
## 28            Malaysia 7.220286e+00          1.951034  3.715881e+00
## 29              Mexico 3.844857e+01          2.643669  1.452993e+01
## 30             Moldova 4.571429e+01          2.634820  1.734711e+01
```

```
## 31          New Zealand 5.312857e+00         3.732477  1.479230e+00
## 32            Nicaragua 1.154286e+02         3.426888  3.363044e+01
## 33               Norway 4.501935e+01         6.377710  7.224868e+00
## 34                 Oman 1.116429e+00         2.899805  3.850000e-01
## 35             Pakistan 3.105043e+02         2.896864  1.028498e+02
## 36                 Peru 1.000287e+01         3.166910  3.178305e+00
## 37          Philippines 1.155329e+02         2.424544  4.826690e+01
## 38               Poland 8.726857e+00         2.529414  3.496123e+00
## 39                Qatar 1.271429e+01         3.491839  3.641143e+00
## 40              Romania 9.671429e+00         2.334470  4.144893e+00
## 41               Russia 8.715857e+01         1.997010  4.390669e+01
## 42         Saudi Arabia 1.091333e+01         2.909601  3.750725e+00
## 43            Singapore 4.608857e+00         3.295776  1.436489e+00
## 44         South Africa 2.192771e+01         2.115975  1.040084e+01
## 45          South Korea 3.734286e+03         3.333225  1.122384e+03
## 46            Sri Lanka 3.925000e+02         2.718564  1.375187e+02
## 47               Sweden 4.110664e+01         5.242570  7.969757e+00
## 48          Switzerland 6.421714e+00         6.137853  1.086516e+00
## 49               Taiwan 7.337143e+01         2.361024  3.117300e+01
## 50             Thailand 8.872829e+01         2.644149  3.459084e+01
## 51               Turkey 3.545536e+05         3.243562  1.350331e+05
## 52                  UAE 1.172727e+01         3.192876  3.672934e+00
## 53               Ukraine 2.947500e+01         1.884213  1.559115e+01
## 54 United Arab Emirates 1.453571e+01         3.957320  3.673121e+00
## 55        United States 4.262465e+00         4.262465  1.000000e+00
## 56              Uruguay 1.105360e+02         3.982612  2.738182e+01
## 57            Venezuela 1.373794e+06         4.317168  1.653621e+05
## 58              Vietnam 6.313333e+04         2.791883  2.260810e+04
##    indice_dolar_medio
## 1         -0.29337171
## 2         -0.12551829
## 3         -0.58797143
## 4         -0.35663286
## 5         -0.04343857
## 6         -0.03700457
## 7         -0.02528800
## 8         -0.19277286
## 9         -0.47299543
## 10        -0.16257100
## 11        -0.17965552
## 12        -0.39406000
## 13        -0.25262771
## 14         0.16471657
## 15        -0.49944969
## 16         0.01808743
## 17        -0.42078857
## 18        -0.37050714
## 19        -0.50556143
## 20        -0.25563514
## 21        -0.57103238
## 22        -0.47144200
## 23        -0.03949077
## 24        -0.24244714
## 25        -0.45697143
```

```
## 26       -0.34594286
## 27       -0.32637000
## 28       -0.53326029
## 29       -0.34403857
## 30       -0.53331429
## 31       -0.12100000
## 32       -0.39308429
## 33        0.50949065
## 34       -0.48654000
## 35       -0.35493333
## 36       -0.24543844
## 37       -0.44042676
## 38       -0.39444714
## 39       -0.38165429
## 40       -0.58640571
## 41       -0.51414914
## 42       -0.35288167
## 43       -0.22985886
## 44       -0.48201543
## 45       -0.19766914
## 46       -0.40670567
## 47        0.25531314
## 48        0.47560257
## 49       -0.41116857
## 50       -0.39695114
## 51       -0.20672909
## 52       -0.24160636
## 53       -0.57000900
## 54       -0.29913000
## 55        0.00000000
## 56       -0.12961833
## 57        0.03041591
## 58       -0.47140600
```

2. Construa uma função denominada `obsPais` que conta quantas observações foram obtidas para cada um dos países informados na chamada da função. Esta funcão:
   a. recebe dois argumentos: `conexao` (a conexão para o banco de dados) e `paises` (um vetor com os nomes dos países de interesse, que pode conter mais que um país);
   b. calcula o número de observações para cada país utilizando SQL e utiliza esta chamada para ordenar os resultados de maneira decrescente pelo número de observações;
   c. faz uma única chamada à base SQL;
   d. retorna ao usuário o `data.frame` resultante.

```
obsPais = function(conexao, paises){
  ## a funcao recebe conexao e paises como argumentos
  ## conexao é o resultado devolvido pela funcao processaDados
  ## paises é um vetor com os nomes (em inglês) dos países de interesse

  ## a chamada é do tipo
  ##  "SELECT nome_pais, COUNT(*) AS n FROM ibm WHERE nome_pais IN ('pais1', 'pais2', ...)"
  ## 1 comando paste() precisa ser usado para formar "('pais1', 'pais2', ...)"
  ## o resultado do paste() acima precisa ser combinado com outro paste para formar o SELECT
  sql = paste("SELECT nome_pais, COUNT(*) AS n FROM ibm ",
              "WHERE nome_pais IN ('", paste(paises, sep="", collapse = "', '"), "') ",
              "GROUP BY nome_pais ORDER BY n DESC", sep="")

  ## a função deve retornar um data.frame com o resultado desejado
  dbGetQuery(conexao, sql)
}
```

O esqueleto da função é dado a seguir

```
obsPais = function(conexao, paises){
  <codigo que consulta a base SQLite para os paises listados>
}
```

Resultados esperados:

```
obsPais(con, c("Brazil", "United States"))
```

```
##       nome_pais  n
## 1 United States 35
## 2        Brazil 35
```

```
obsPais(con, "Brazil")
```

```
##   nome_pais  n
## 1    Brazil 35
```

```
obsPais(con, c("Venezuela", "Brazil", "Argentina", "United States"))
```

```
##       nome_pais  n
## 1 United States 35
## 2        Brazil 35
## 3     Argentina 35
## 4     Venezuela 22
```

3. Crie uma função chamada `coletaDados` para obter as observações referentes a países de interesse.
    a. A função receberá dois argumentos: `conexao` (conexão para o banco de dados) e `paises` (um vetor com países de interesse, que pode ter mais que 1 país listado);
    b. A função retornará um `data.frame` contendo `data`, `nome_pais`, `preco_dolar` e `indice_dolar`;
    c. A coluna `data` deve estar no formato apropriado para data (i.e., não deve ser do tipo `character`);
    d. O `data.frame` de saída deve estar ordenado por nome do país e por data;
    e. O `data.frame` de saída deve conter apenas os países listados no argumento `paises`;
    f. A extração do subconjunto de dados deve acontecer utilizando SQL (uma única chamada à base SQL);
    g. A ordenação do subconjunto de dados, por país e data, pode acontecer fora do SQL.

```
coletaDados = function(conexao, paises){
  ## a funcao recebe conexao e paises como argumentos
  ## conexao é o resultado devolvido pela funcao processaDados
  ## paises é um vetor com os nomes (em inglês) dos países de interesse

  ## a chamada é do tipo
  ##  "SELECT data, nome_pais, preco_dolar, indice_dolar FROM ibm WHERE nome_pais IN ('pais1', 'pais2',
  ## 1 comando paste() precisa ser usado para formar "('pais1', 'pais2', ...)"
  ## o resultado do paste() acima precisa ser combinado com outro paste para formar o SELECT
  sql = paste("SELECT data, nome_pais, preco_dolar, indice_dolar FROM ibm WHERE nome_pais IN ('",
              paste(paises, collapse = "', '"),
              "')", sep = '')

  ## extrai um data.frame da base de dados
  output = dbGetQuery(conexao, sql)

  ## a data no data.frame vem em formato character e precisa ser convertido para data
  ## o resultado é, então, ordenado por nome do país e data
  output = output %>%
    mutate(data = as.Date(data)) %>%
    arrange(nome_pais, data)

  ## data.frame ordenado é retornado ao usuário
  output
}
```

Resultados esperados:

```
coletaDados(con, c("Brazil", "United States"))
```

```
##          data     nome_pais preco_dolar indice_dolar
## 1  2000-04-01        Brazil    1.648045     -0.34341
## 2  2001-04-01        Brazil    1.643836     -0.35282
## 3  2002-04-01        Brazil    1.538462     -0.38214
## 4  2003-04-01        Brazil    1.482085     -0.45311
## 5  2004-05-01        Brazil    1.698113     -0.41444
## 6  2005-06-01        Brazil    2.393703     -0.21774
## 7  2006-01-01        Brazil    2.741543     -0.12967
## 8  2006-05-01        Brazil    2.777175     -0.10414
## 9  2007-01-01        Brazil    2.999766     -0.06840
## 10 2007-06-01        Brazil    3.606900      0.05774
## 11 2008-06-01        Brazil    4.733056      0.32579
## 12 2009-07-01        Brazil    4.020830      0.12628
## 13 2010-01-01        Brazil    4.758313      0.32914
## 14 2010-07-01        Brazil    4.907180      0.31442
```

```
## 15 2011-07-01        Brazil    6.162429      0.51597
## 16 2012-01-01        Brazil    5.678670      0.35296
## 17 2012-07-01        Brazil    4.935974      0.14061
## 18 2013-01-01        Brazil    5.643766      0.29225
## 19 2013-07-01        Brazil    5.284830      0.15980
## 20 2014-01-01        Brazil    5.247456      0.13479
## 21 2014-07-01        Brazil    5.855196      0.22110
## 22 2015-01-01        Brazil    5.206827      0.08702
## 23 2015-07-01        Brazil    4.282519     -0.10595
## 24 2016-01-01        Brazil    3.354204     -0.31963
## 25 2016-07-01        Brazil    4.781811     -0.05123
## 26 2017-01-01        Brazil    5.117945      0.01145
## 27 2017-07-01        Brazil    5.101568     -0.03744
## 28 2018-01-01        Brazil    5.111683     -0.03188
## 29 2018-07-01        Brazil    4.402934     -0.20092
## 30 2019-01-01        Brazil    4.545516     -0.18539
## 31 2019-07-09        Brazil    4.596433     -0.19923
## 32 2020-01-14        Brazil    4.804558     -0.15264
## 33 2020-07-01        Brazil    3.913528     -0.31462
## 34 2021-01-01        Brazil    3.978491     -0.29709
## 35 2021-07-01        Brazil    4.363027     -0.22778
## 36 2000-04-01 United States    2.510000      0.00000
## 37 2001-04-01 United States    2.540000      0.00000
## 38 2002-04-01 United States    2.490000      0.00000
## 39 2003-04-01 United States    2.710000      0.00000
## 40 2004-05-01 United States    2.900000      0.00000
## 41 2005-06-01 United States    3.060000      0.00000
## 42 2006-01-01 United States    3.150000      0.00000
## 43 2006-05-01 United States    3.100000      0.00000
## 44 2007-01-01 United States    3.220000      0.00000
## 45 2007-06-01 United States    3.410000      0.00000
## 46 2008-06-01 United States    3.570000      0.00000
## 47 2009-07-01 United States    3.570000      0.00000
## 48 2010-01-01 United States    3.580000      0.00000
## 49 2010-07-01 United States    3.733333      0.00000
## 50 2011-07-01 United States    4.065000      0.00000
## 51 2012-01-01 United States    4.197220      0.00000
## 52 2012-07-01 United States    4.327500      0.00000
## 53 2013-01-01 United States    4.367396      0.00000
## 54 2013-07-01 United States    4.556667      0.00000
## 55 2014-01-01 United States    4.624167      0.00000
## 56 2014-07-01 United States    4.795000      0.00000
## 57 2015-01-01 United States    4.790000      0.00000
## 58 2015-07-01 United States    4.790000      0.00000
## 59 2016-01-01 United States    4.930000      0.00000
## 60 2016-07-01 United States    5.040000      0.00000
## 61 2017-01-01 United States    5.060000      0.00000
## 62 2017-07-01 United States    5.300000      0.00000
## 63 2018-01-01 United States    5.280000      0.00000
## 64 2018-07-01 United States    5.510000      0.00000
## 65 2019-01-01 United States    5.580000      0.00000
## 66 2019-07-09 United States    5.740000      0.00000
## 67 2020-01-14 United States    5.670000      0.00000
## 68 2020-07-01 United States    5.710000      0.00000
```

```
## 69 2021-01-01 United States    5.660000      0.00000
## 70 2021-07-01 United States    5.650000      0.00000
```

```
coletaDados(con, "Brazil")
```

```
##          data nome_pais preco_dolar indice_dolar
## 1  2000-04-01    Brazil    1.648045     -0.34341
## 2  2001-04-01    Brazil    1.643836     -0.35282
## 3  2002-04-01    Brazil    1.538462     -0.38214
## 4  2003-04-01    Brazil    1.482085     -0.45311
## 5  2004-05-01    Brazil    1.698113     -0.41444
## 6  2005-06-01    Brazil    2.393703     -0.21774
## 7  2006-01-01    Brazil    2.741543     -0.12967
## 8  2006-05-01    Brazil    2.777175     -0.10414
## 9  2007-01-01    Brazil    2.999766     -0.06840
## 10 2007-06-01    Brazil    3.606900      0.05774
## 11 2008-06-01    Brazil    4.733056      0.32579
## 12 2009-07-01    Brazil    4.020830      0.12628
## 13 2010-01-01    Brazil    4.758313      0.32914
## 14 2010-07-01    Brazil    4.907180      0.31442
## 15 2011-07-01    Brazil    6.162429      0.51597
## 16 2012-01-01    Brazil    5.678670      0.35296
## 17 2012-07-01    Brazil    4.935974      0.14061
## 18 2013-01-01    Brazil    5.643766      0.29225
## 19 2013-07-01    Brazil    5.284830      0.15980
## 20 2014-01-01    Brazil    5.247456      0.13479
## 21 2014-07-01    Brazil    5.855196      0.22110
## 22 2015-01-01    Brazil    5.206827      0.08702
## 23 2015-07-01    Brazil    4.282519     -0.10595
## 24 2016-01-01    Brazil    3.354204     -0.31963
## 25 2016-07-01    Brazil    4.781811     -0.05123
## 26 2017-01-01    Brazil    5.117945      0.01145
## 27 2017-07-01    Brazil    5.101568     -0.03744
## 28 2018-01-01    Brazil    5.111683     -0.03188
## 29 2018-07-01    Brazil    4.402934     -0.20092
## 30 2019-01-01    Brazil    4.545516     -0.18539
## 31 2019-07-09    Brazil    4.596433     -0.19923
## 32 2020-01-14    Brazil    4.804558     -0.15264
## 33 2020-07-01    Brazil    3.913528     -0.31462
## 34 2021-01-01    Brazil    3.978491     -0.29709
## 35 2021-07-01    Brazil    4.363027     -0.22778
```

```
coletaDados(con, c("Venezuela", "Brazil", "Argentina", "United States"))
```

```
##           data     nome_pais preco_dolar indice_dolar
## 1   2000-04-01     Argentina   2.5000000     -0.00398
## 2   2001-04-01     Argentina   2.5000000     -0.01575
## 3   2002-04-01     Argentina   0.7987220     -0.67923
## 4   2003-04-01     Argentina   1.4236111     -0.47468
## 5   2004-05-01     Argentina   1.4779661     -0.49036
## 6   2005-06-01     Argentina   1.6396272     -0.46417
## 7   2006-01-01     Argentina   1.5503623     -0.50782
## 8   2006-05-01     Argentina   2.2902012     -0.26123
## 9   2007-01-01     Argentina   2.6709834     -0.17050
## 10  2007-06-01     Argentina   2.6686075     -0.21742
```

```
## 11  2008-06-01     Argentina   3.6429872      0.02044
## 12  2009-07-01     Argentina   3.0173827     -0.15479
## 13  2010-01-01     Argentina   1.8428327     -0.48524
## 14  2010-07-01     Argentina   3.5589450     -0.04671
## 15  2011-07-01     Argentina   4.8396854      0.19057
## 16  2012-01-01     Argentina   4.6366060      0.10469
## 17  2012-07-01     Argentina   4.1609636     -0.03848
## 18  2013-01-01     Argentina   3.8179443     -0.12581
## 19  2013-07-01     Argentina   3.8799076     -0.14852
## 20  2014-01-01     Argentina   3.0341340     -0.34385
## 21  2014-07-01     Argentina   2.5707728     -0.46386
## 22  2015-01-01     Argentina   3.2520325     -0.32108
## 23  2015-07-01     Argentina   3.0651341     -0.36010
## 24  2016-01-01     Argentina   2.3897026     -0.51527
## 25  2016-07-01     Argentina   3.3478406     -0.33575
## 26  2017-01-01     Argentina   3.4683904     -0.31455
## 27  2017-07-01     Argentina   4.1255341     -0.22160
## 28  2018-01-01     Argentina   3.9603960     -0.24992
## 29  2018-07-01     Argentina   2.7051398     -0.50905
## 30  2019-01-01     Argentina   2.0024029     -0.64115
## 31  2019-07-09     Argentina   2.8705044     -0.49991
## 32  2020-01-14     Argentina   2.8468874     -0.49790
## 33  2020-07-01     Argentina   3.5092324     -0.38542
## 34  2021-01-01     Argentina   3.7482313     -0.33777
## 35  2021-07-01     Argentina   3.9446196     -0.30184
## 36  2000-04-01        Brazil   1.6480447     -0.34341
## 37  2001-04-01        Brazil   1.6438356     -0.35282
## 38  2002-04-01        Brazil   1.5384615     -0.38214
## 39  2003-04-01        Brazil   1.4820847     -0.45311
## 40  2004-05-01        Brazil   1.6981132     -0.41444
## 41  2005-06-01        Brazil   2.3937033     -0.21774
## 42  2006-01-01        Brazil   2.7415432     -0.12967
## 43  2006-05-01        Brazil   2.7771751     -0.10414
## 44  2007-01-01        Brazil   2.9997656     -0.06840
## 45  2007-06-01        Brazil   3.6069002      0.05774
## 46  2008-06-01        Brazil   4.7330557      0.32579
## 47  2009-07-01        Brazil   4.0208302      0.12628
## 48  2010-01-01        Brazil   4.7583125      0.32914
## 49  2010-07-01        Brazil   4.9071805      0.31442
## 50  2011-07-01        Brazil   6.1624286      0.51597
## 51  2012-01-01        Brazil   5.6786704      0.35296
## 52  2012-07-01        Brazil   4.9359743      0.14061
## 53  2013-01-01        Brazil   5.6437655      0.29225
## 54  2013-07-01        Brazil   5.2848303      0.15980
## 55  2014-01-01        Brazil   5.2474556      0.13479
## 56  2014-07-01        Brazil   5.8551965      0.22110
## 57  2015-01-01        Brazil   5.2068267      0.08702
## 58  2015-07-01        Brazil   4.2825194     -0.10595
## 59  2016-01-01        Brazil   3.3542039     -0.31963
## 60  2016-07-01        Brazil   4.7818106     -0.05123
## 61  2017-01-01        Brazil   5.1179454      0.01145
## 62  2017-07-01        Brazil   5.1015676     -0.03744
## 63  2018-01-01        Brazil   5.1116825     -0.03188
## 64  2018-07-01        Brazil   4.4029336     -0.20092
```

```
## 65  2019-01-01        Brazil   4.5455157     -0.18539
## 66  2019-07-09        Brazil   4.5964332     -0.19923
## 67  2020-01-14        Brazil   4.8045583     -0.15264
## 68  2020-07-01        Brazil   3.9135279     -0.31462
## 69  2021-01-01        Brazil   3.9784907     -0.29709
## 70  2021-07-01        Brazil   4.3630267     -0.22778
## 71  2000-04-01 United States   2.5100000      0.00000
## 72  2001-04-01 United States   2.5400000      0.00000
## 73  2002-04-01 United States   2.4900000      0.00000
## 74  2003-04-01 United States   2.7100000      0.00000
## 75  2004-05-01 United States   2.9000000      0.00000
## 76  2005-06-01 United States   3.0600000      0.00000
## 77  2006-01-01 United States   3.1500000      0.00000
## 78  2006-05-01 United States   3.1000000      0.00000
## 79  2007-01-01 United States   3.2200000      0.00000
## 80  2007-06-01 United States   3.4100000      0.00000
## 81  2008-06-01 United States   3.5700000      0.00000
## 82  2009-07-01 United States   3.5700000      0.00000
## 83  2010-01-01 United States   3.5800000      0.00000
## 84  2010-07-01 United States   3.7333333      0.00000
## 85  2011-07-01 United States   4.0650000      0.00000
## 86  2012-01-01 United States   4.1972200      0.00000
## 87  2012-07-01 United States   4.3275000      0.00000
## 88  2013-01-01 United States   4.3673958      0.00000
## 89  2013-07-01 United States   4.5566667      0.00000
## 90  2014-01-01 United States   4.6241667      0.00000
## 91  2014-07-01 United States   4.7950000      0.00000
## 92  2015-01-01 United States   4.7900000      0.00000
## 93  2015-07-01 United States   4.7900000      0.00000
## 94  2016-01-01 United States   4.9300000      0.00000
## 95  2016-07-01 United States   5.0400000      0.00000
## 96  2017-01-01 United States   5.0600000      0.00000
## 97  2017-07-01 United States   5.3000000      0.00000
## 98  2018-01-01 United States   5.2800000      0.00000
## 99  2018-07-01 United States   5.5100000      0.00000
## 100 2019-01-01 United States   5.5800000      0.00000
## 101 2019-07-09 United States   5.7400000      0.00000
## 102 2020-01-14 United States   5.6700000      0.00000
## 103 2020-07-01 United States   5.7100000      0.00000
## 104 2021-01-01 United States   5.6600000      0.00000
## 105 2021-07-01 United States   5.6500000      0.00000
## 106 2002-04-01     Venezuela   2.9171529      0.17155
## 107 2003-04-01     Venezuela   2.3153942     -0.14561
## 108 2004-05-01     Venezuela   1.4779980     -0.49035
## 109 2005-06-01     Venezuela   2.1295853     -0.30406
## 110 2006-01-01     Venezuela   2.2554207     -0.28399
## 111 2006-05-01     Venezuela   2.2627779     -0.27007
## 112 2007-01-01     Venezuela   1.6483738     -0.48808
## 113 2007-06-01     Venezuela   3.4461882      0.01061
## 114 2011-07-01     Venezuela   6.5197397      0.60387
## 115 2012-01-01     Venezuela   6.9854354      0.66430
## 116 2012-07-01     Venezuela   7.9168267      0.82942
## 117 2013-01-01     Venezuela   9.0810660      1.07929
## 118 2013-07-01     Venezuela   7.1518253      0.56953
```

```
## 119 2014-01-01     Venezuela   7.1518253      0.54662
## 120 2014-07-01     Venezuela   6.8181818      0.42194
## 121 2015-01-01     Venezuela   2.5334871     -0.47109
## 122 2015-07-01     Venezuela   0.6700508     -0.86011
## 123 2016-01-01     Venezuela   0.6643227     -0.86525
## 124 2016-07-01     Venezuela   3.3832237     -0.32873
## 125 2017-01-01     Venezuela   5.2467859      0.03691
## 126 2017-07-01     Venezuela   4.0555556     -0.23480
## 127 2021-07-01     Venezuela   8.3464835      0.47725
```

4. Utilizando a função criada no item anterior, apresente o código (utilizando `tidyverse` e `ggplot`) que replica exatamente o gráfico abaixo:

```
## a função coletaDados() extrai as informações de interesse para os países apresentados no gráfico
coletaDados(con, c("Brazil", "Argentina", "United States")) %>%
  ## eixo x: data (em formato data)
  ## eixo y: índice não-ajustado relativo ao dolar
  ## cores: países
  ggplot(aes(data, indice_dolar, colour = nome_pais)) +
  ## gráfico de linha
  geom_line() +
  ## tema preto e branco
  theme_bw() +
  ## títulos das diferentes características do gráfico
  labs(x = "Data", y = "Índice Big Mac (não ajustado) em USD",
       title = "Evolução do Índice Big Mac",
       subtitle = "Comparação: Argentina, Brasil, Estados Unidos",
       colour = "Países")
```