MBA USP ESALQ

Data Wrangling

Prof. Dr. Wilson Tarantin Jr.

Preparação de Dados no R

Data wrangling

- Utilizaremos, principalmente, o dplyr
  - O dplyr é um pacote contido no tidyverse
  - Contém funções úteis para a manipulação/preparação de bancos de dados
  - Material para referência:
    - https://dplyr.tidyverse.org/
    - https://github.com/rstudio/cheatsheets/blob/master/data-transformation.pdf
    - Wickham, H. & Grolemund, G. **R for Data Science**: https://r4ds.had.co.nz/index.html

Data wrangling

- **Pipe**: encadeamento de diversas funções em sequência
- **Rename**: alteração de nomes de variáveis
- **Mutate**: alteração de conteúdo das variáveis e criação de novas variáveis
- **Filter**: seleção de observações com base em critérios lógicos
- **Select**: seleção de variáveis
- **Summarise**: criação de tabelas com medidas resumo (estatísticas descritivas)
- **Group by**: agrupamento das observações com base em critérios
- **Join**: junção (*merge*) de bancos de dados

Criação de Projects e Scripts R Markdown

R Markdown

- **Introdução ao R Markdown**
- **Formatação básica do texto**
- **Inserção de fórmulas**
- **Chunks**
- **Gerando outputs (HTML; PDF, DOC)**

- Material para referência:

  - https://rmarkdown.rstudio.com/index.html

Projetos de Data Science & Analytics no GitHub

Git

- Software útil para o controle de versões
- Registra as alterações feitas nos arquivos
- Vamos utilizá-lo em conjunto com o Github
- Instalar o Git no computador (https://git-scm.com/downloads)
  - Basta avançar todas as etapas nas configurações sugeridas

Github

- Site utilizado para hospedar os arquivos
  - https://github.com/
- Organizado em repositórios (pastas) que podem ser compartilhadas, inclusive, podem ser publicadas
  - Útil para armazenar e compartilhar seu portfólio de projetos
- Os arquivos do computador podem ser enviados ao Github (pelo Git)

Git e Github

- Add e Commit

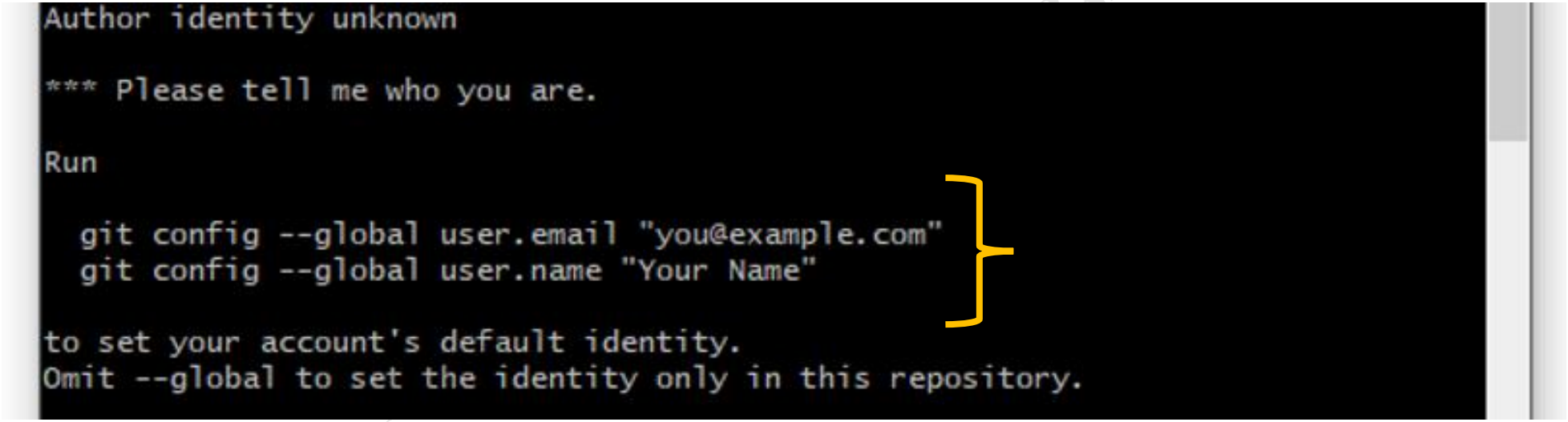
  - Crie uma pasta na área de trabalho de seu computador
  - No RStudio, crie um novo script e escreva apenas # Versão 1
  - Salve este arquivo na pasta com o nome Versão Exemplo.R
  - Dentro da pasta, clique com o botão direito do mouse e escolha Git Bash Here

  - No Git, escreva **git init** (inicializa o Git na pasta selecionada)
  - Escreva **git add "Versão Exemplo.R"** (adiciona o arquivo para o índice)
  - Para gerar versões utilize o comando **git commit -m "título"** (são as versões)

O nome do commit, exemplo: "Primeira Versão"

Git: configuração inicial

- Na primeira vez em que utiliza o Git, há um cadastro inicial



- Após surgir esta mensagem, digite um comando e depois o outro
  - **git config --global user.email “seu email”**
  - **git config --global user.name “seu nome”**

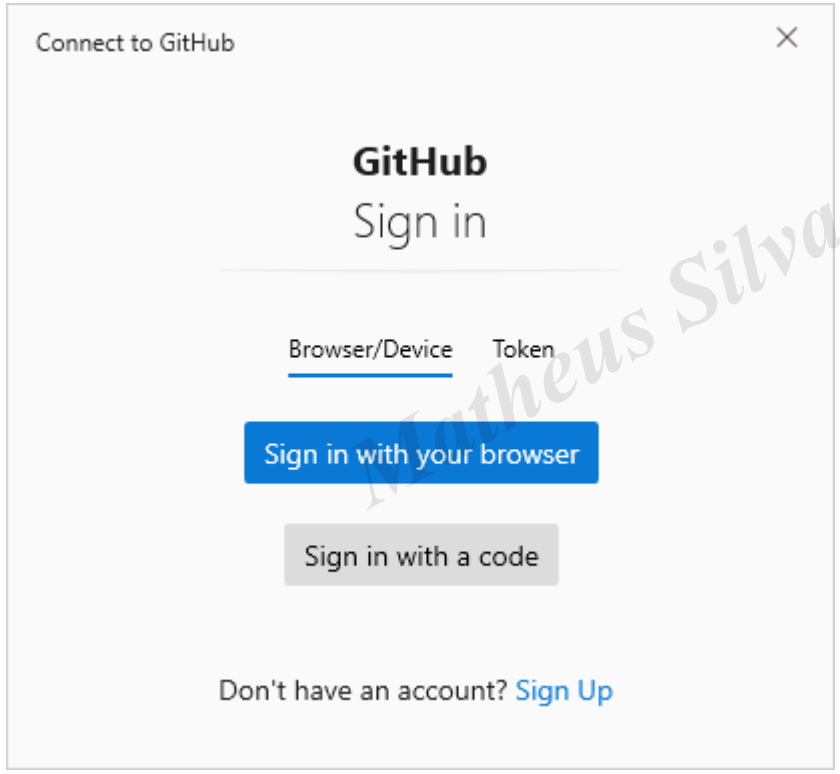
Normalmente, ela surge após o primeiro commit. Depois de cadastrar, refaça o commit

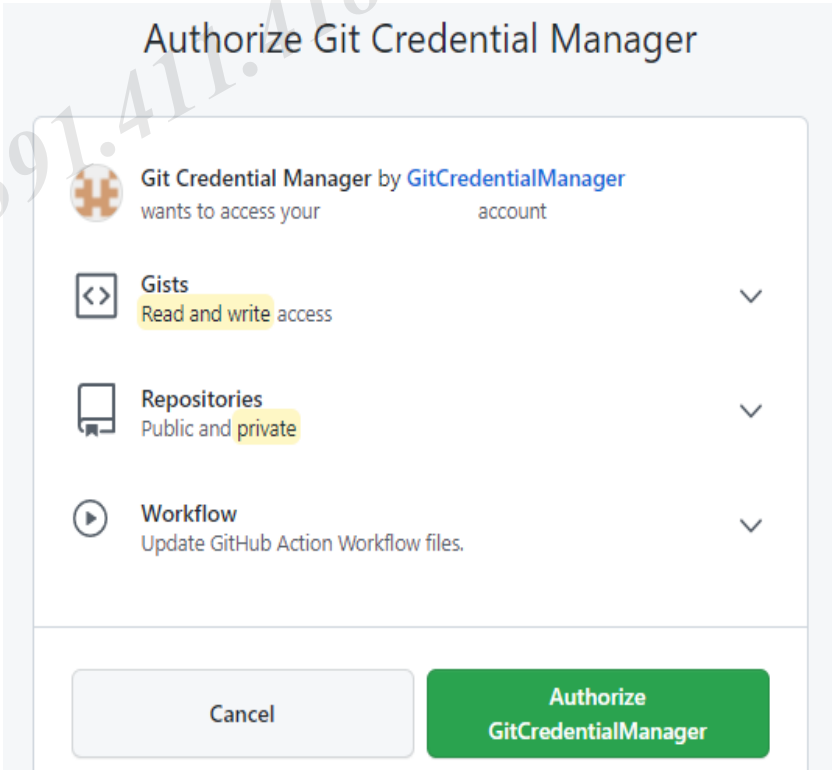
Git e Github

- Push
  - Em seu Github, crie um novo repositório e nomeie como preferir
  - Copie o link do repositório criado
  - No Git, escreva **git remote add origin .....(link de sua pasta).....**
  - Por fim, digite **git push –u origin master** (envia o arquivo para o repositório, ficando na ramificação principal)
    - Na primeira vez que for feito, solicitará login no Github
  - Após atualizar, é possível verificar que o arquivo já está em seu Github!

Git e Github: conexão inicial

- Caso seja a primeira vez que utiliza o Git, há um login





Esta é por meio do browser

Git e Github

- Criando e comparando versões

  - Abra o arquivo Versão Exemplo e escreva mais uma linha: # Versão 2
  - Após salvar, feche e com o botão direito abra o Git Bash Here na pasta
  - Utilize os mesmos procedimentos:

  - **git add “Versão Exemplo.R”**
  - **git commit -m “Segunda Versão”**
  - **git push –u origin master**

Note que não foi necessário informar novamente o endereço

  - No Github, a nova versão já está disponível e podemos compará-las!

Git e Github

- Criando ramificações no repositório

  - Nos comandos anteriores, alteramos a ramificação principal do repositório
  - Poderíamos criar ramificações novas no Github

  - **git checkout -b “nome da nova branch”**
  - No Git, já há a indicação de mudança da “master” para a “nova”

  - Os mesmos procedimentos de add e commit
  - **git push -u origin “nome da nova branch”**

Git e Github

- Importando repositórios (Clone e Pull)

  - Pode ser útil trazer para seu computador arquivos que estão no Github
  - Uma forma de “baixar” tais arquivos é por meio da função clone

  - Crie uma pasta em seu computador
  - Dentro da pasta, com o botão direito do mouse, abra o Git Bash Here
  - No Github, no repositório de interesse, clique em **code** e copie o link
  - No Git, digite **git clone .....(link do repositório).....**
  - Para baixar novamente, após alterações no Github, indique **cd “repositório”**
  - Na sequencia, digite **git pull** (o arquivo foi atualizado no computador)

Git e Github

- Copiando repositórios públicos (Fork)
  - É possível copiar repositórios que estão publicados no Github
  - Procure por algum tema de interesse
  - Acesse o repositório
  - No canto superior direito, existe o botão **Fork**
  - Após clicar, poderá ver o repositório em sua lista (em seu perfil)

Git, Github e RStudio

- É possível integrar o Git, Github e RStudio

- No RStudio, clique em File → New Project → Version Control → Git

  - Em “Repository URL” basta indicar o link do repositório no Github

- Após criar um documento (R Script, R Markdown), clique em Git e faça o **commit** e, em seguida, o **push**

  - Também é possível fazer o **pull** dos arquivos do repositório que foi indicado

Funções e Iterações com Pacote Purrr

Functions, Purrr

- **Criando funções no R**
- **Atribuindo condições (“IF”)**
- **Iterações com Purrr (funções map)**

- Material para referência:

  - Wickham, H. & Grolemund, G. **R for Data Science**: https://r4ds.had.co.nz/index.html
  - https://github.com/rstudio/cheatsheets/blob/master/purrr.pdf