

# Introdução ao Quarto

ESTAT0109 – Mineração de Dados em Estatística

Prof. Dr. Sadraque E. F. Lucena

sadraquelucena@academico.ufs.br

<http://sadraquelucena.github.io/mineracao>

# Objetivo da Aula

Aprender a **criar, editar e renderizar** documentos **Quarto** no **RStudio**, integrando

- texto;
- código;
- tabelas;
- saídas do R;
- gráficos;
- referências.

# Estrutura da Aula

1. O que é o Quarto?
2. Quando e como usaremos o Quarto no RStudio
3. Estrutura de um documento `.qmd`
4. YAML e principais chaves
5. Texto e formatação (Markdown)
6. Blocos de código (R)
7. Renderização e formatos de saída
8. Dicas, dúvidas e referências

# O que é o Quarto?

Quarto é uma **plataforma unificada** para criar documentos, apresentações e sites combinando texto, código (em *R*, *Python*, *Julia* ou *Bash*) e suas saídas.

- Substitui o R Markdown
- Integra código, texto e resultados no mesmo arquivo
- Pode gerar **HTML, PDF, Word, slides, sites e livros**
- É o padrão atual para comunicação científica com RStudio.

# Pré-requisitos

Antes de começar, você precisa ter instalado:



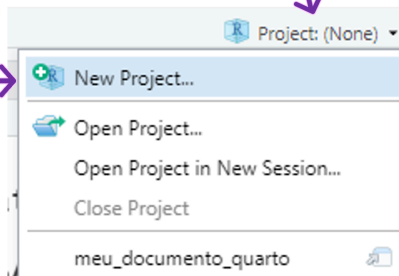
- Alternativa: use o **Posit Cloud** (<https://posit.cloud/>) para criar e renderizar documentos na nuvem.

# Começando

Primeiro crie um projeto.

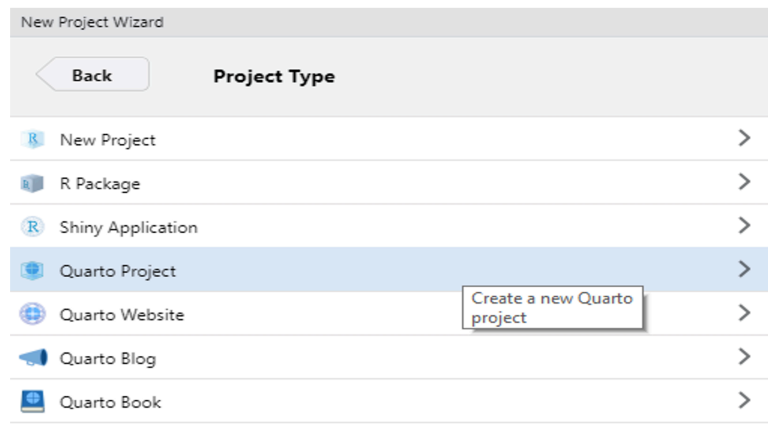
1º No canto superior direito do RStudio, clique em *Project*

2º Clicar aqui

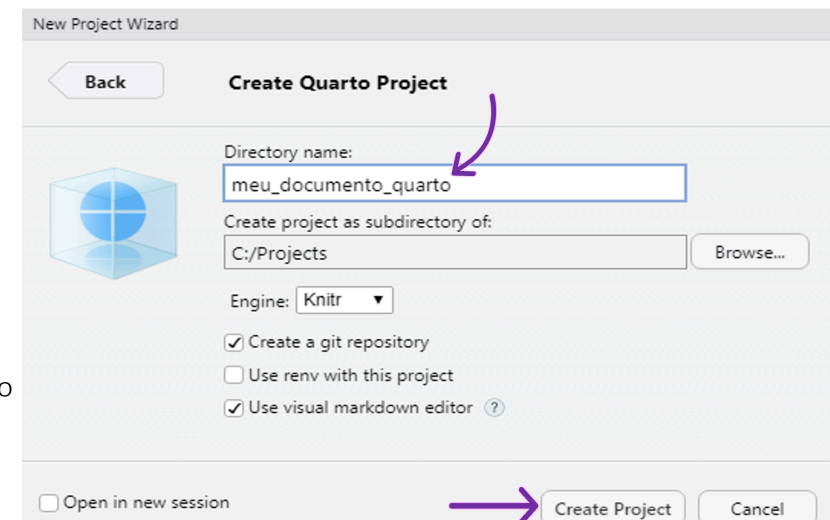


3º Escolha um novo diretório de trabalho (ou associe a algum já existente)

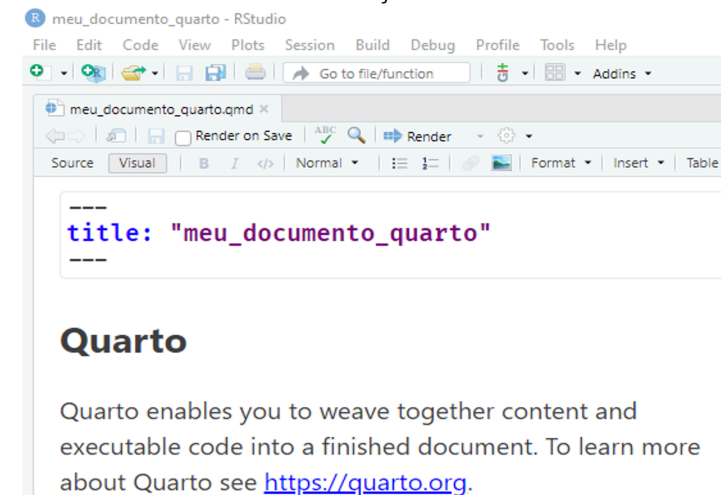
4º Selecione *Quarto Project*



5º Nomeie o diretório e clique em "Create Project"



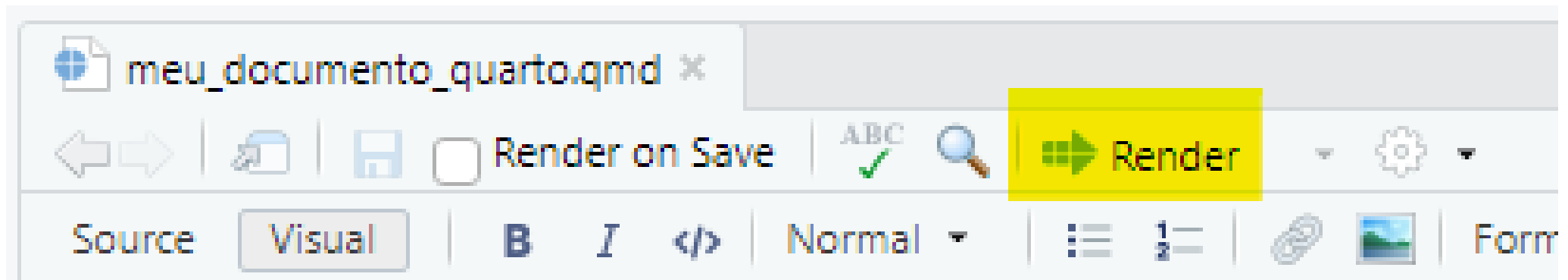
6º Pronto! Projeto criado! :D



Fonte: <https://rladies-sp.org/>

# Começando

O arquivo quarto editável tem extensão *.qmd*. Para renderizá-lo clique em render ou utilize o atalho do teclado **CTRL + SHIFT + K**.



Fonte: <https://rladies-sp.org/>

# Estrutura de um arquivo `.qmd`

Um arquivo `.qmd` é dividido em três partes:

1. **Cabeçalho (YAML)** — define o formatação do documento, título, autor, data e outros detalhes.
2. **Corpo (Markdown)** — texto, títulos, listas, tabelas, imagens.
3. **Blocos de código (Code chunks)** — código executável em R, Python, Julia e outras linguagens.



# YAML (Yet Another Markup Language)

O YAML fica no início do documento e define as configurações básicas. Exemplo:

```
---  
title: "Relatório Anual"  
author: "Equipe de Estatística"  
date: today  
lang: pt  
format: html  
---
```

- **Dica:** Tudo entre `---` e `---` é parte do YAML e deve ser escrito com atenção à indentação.

# YAML (Yet Another Markup Language)

Em YAML, os elementos são chamados de pares chave-valor. Algumas chaves são:

- `title`: título do documento.
- `author`: nome do(a) autor(a). Pode ser mais de um(a).
- `date`: Define a data do documento. Exemplos: `"13-03-2025"`, `today`.
- `lang`: Define o idioma do documento. Exemplos: `pt`, `en`.
- `format`: tipo de arquivo de saída.

Mais detalhes: [quarto.org/docs/authoring/front-matter.html](https://quarto.org/docs/authoring/front-matter.html)

# Blocos de código

Blocos de código são onde você executa o R (ou outra linguagem). Começam com três crases e o nome da linguagem. Exemplo:

```
```{r}
#| echo: false
#| eval: true
head(airquality)
```
```

|   | Ozone<br><int> | Solar.R<br><int> | Wind<br><dbl> | Temp<br><int> | Month<br><int> | Day<br><int> |
|---|----------------|------------------|---------------|---------------|----------------|--------------|
| 1 | 41             | 190              | 7.4           | 67            | 5              | 1            |
| 2 | 36             | 118              | 8.0           | 72            | 5              | 2            |
| 3 | 12             | 149              | 12.6          | 74            | 5              | 3            |
| 4 | 18             | 313              | 11.5          | 62            | 5              | 4            |
| 5 | NA             | NA               | 14.3          | 56            | 5              | 5            |
| 6 | 28             | NA               | 14.9          | 66            | 5              | 6            |

6 rows

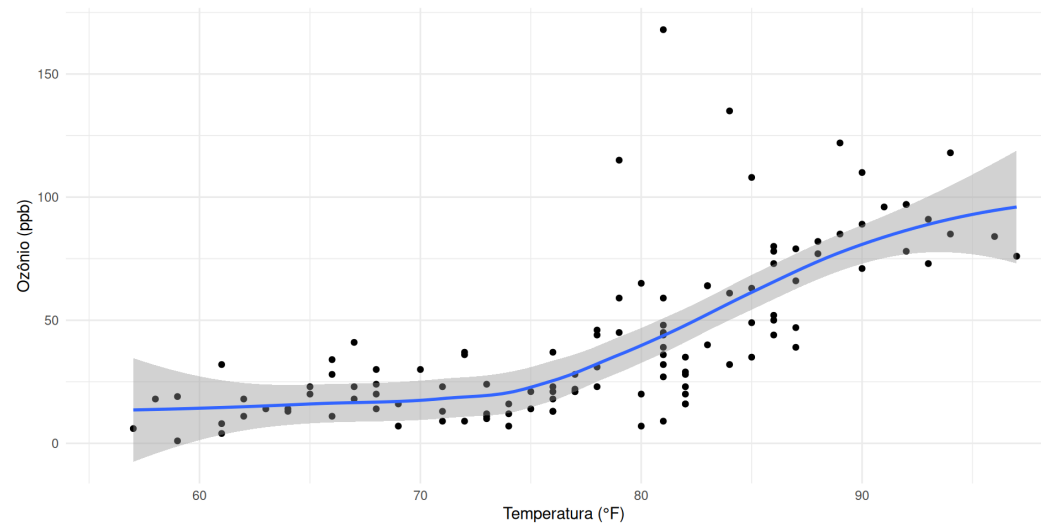
# Blocos de código

Algumas das principais opções são:

- `#| echo`: controla se o código é exibido no documento.
- `#| eval`: determina se o código é executado e a saída apresentada.
- `#| warning`: controla a exibição de mensagens de aviso geradas pelo código.
- `#| error`: determina se os erros devem ser incluídos na saída.
- `#| label`: rótulo dado para fazer referência. Uma vez definido um rótulo, você pode referenciá-lo usando `(@nome_do_rotulo)`.
- `#| fig-cap`: adiciona uma legenda a gráficos gerados pelo código.

# Blocos de código

```
```{r}
#| echo: false
#| eval: true
#| fig-align: center
library(ggplot2)
ggplot(airquality, aes(Temp, Ozone)) +
  geom_point() +
  geom_smooth(method = "loess") +
  labs(x = "Temperatura (°F)", y = "Ozônio (ppb)") +
  theme_minimal()
```
```



# Formatação de texto

| Sintaxe                            | Saída                         |
|------------------------------------|-------------------------------|
| <code>*itálico*</code>             | <i>itálico</i>                |
| <code>**negrito**</code>           | <b>negrito</b>                |
| <code>***itálico negrito***</code> | <b><i>itálico negrito</i></b> |
| <code>sobrescrito^2^</code>        | sobrescrito <sup>2</sup>      |
| <code>subscrito~2~</code>          | subscrito <sub>2</sub>        |
| <code>~~riscado~~</code>           | <del>riscado</del>            |
| <code>`código literal`</code>      | <code>código literal</code>   |

# Seções

Uma seção e as subseções são definidas pela quantidade de #. O limite são seis níveis. Exemplos:

| Sintaxe           | Saída                |
|-------------------|----------------------|
| # Seção 1         | <b>Seção 1</b>       |
| ## Subseção 1     | <b>Subseção 1</b>    |
| ### Subsubseção 1 | <b>Subsubseção 1</b> |

# Links e imagens

- `<https://quarto.org>` produz: <https://quarto.org>
- `[Quarto](https://quarto.org)` produz: Quarto
- `` produz:





# Lista sem ordem

## Sintaxe

\* Item

+ subitem 1

+ subitem 2

- subsubitem 1

## Saída

- Lista sem ordem
  - subitem 1
  - subitem 2
    - subsubitem 1

# Lista ordenada

## Sintaxe

1. Item 1
2. Item 2
  - i) subitem 1
    - A. subsubitem 1

## Saída

1. Item 1
2. Item 2
  - i. subitem 1
    - a. subsubitem 1

# Tabelas

| Direita | Esquerda | Padrão | Centralizada |
|---------|----------|--------|--------------|
| -----:  | :-----   | -----  | :-----:      |
| 12      | 12       | 12     | 12           |
| 123     | 123      | 123    | 123          |
| 1       | 1        | 1      | 1            |

produz:

| Direita | Esquerda | Padrão | Centralizada |
|---------|----------|--------|--------------|
| 12      | 12       | 12     | 12           |
| 123     | 123      | 123    | 123          |
| 1       | 1        | 1      | 1            |

# Equações

- Para fórmulas e símbolos matemáticos embutidos no texto use `$`.
- Para fórmulas e símbolos matemáticos destacados use `$$`.
- `$X \sim N(\mu, \sigma^2)$` produz:  $X \sim N(\mu, \sigma^2)$
- `$$ f(x) = \frac{a}{b} e^{-x} $$` produz

$$f(x) = \frac{a}{b} e^{-x}$$

# Citação

- Informe o arquivo com as referências no YAML:

```
---  
title: Título  
bibliography: referencias.bib  
csl: abnt.csl  
---
```

- Formato padrão: *Chicago Manual of Style* com autor e data.
- Você pode especificar uma formatação personalizada usando o CSL (*Citation Style Language*) nos repositórios
  - <https://github.com/citation-style-language/styles> ou
  - <https://www.zotero.org/styles>

# Citação

- @Wickham2023 produz (*autor, ano*):
  - WICKHAM; ÇETINKAYA-RUNDEL; GROLEMUND (2023)
- [@Wickham2023] produz *autor (ano)*:
  - (WICKHAM; ÇETINKAYA-RUNDEL; GROLEMUND, 2023)
- O arquivo csl com a formatação da ABNT pode ser encontrada em <https://www.zotero.org/styles/universidade-federal-de-sergipe-departamento-de-engenharia-de-producao-abnt>

# Fim

Agora vamos fazer no RStudio.

# Referências

WICKHAM, H.; ÇETINKAYA-RUNDEL, M.; GROLEMUND, G. **R for data science: Import, tidy, transform, visualize, and model data**. 2nd. ed. Sebastopol, CA: O'Reilly, 2023.