

 [guilhermeonrails / idcp-alura](#) Public[!\[\]\(919a2cb85b99741a73c0c31a427236a8_img.jpg\) Code](#) [!\[\]\(c9cd5a1c35167a83f09a35036fe5dcbd_img.jpg\) Issues](#) [!\[\]\(ae1936640fabdea8c18f922ca69733fe_img.jpg\) Pull requests](#) [!\[\]\(e81307241bb070bc7c1be4e4328b2244_img.jpg\) Actions](#) [!\[\]\(5145ac5c495d0d3391897543e0ba7223_img.jpg\) Projects](#) [!\[\]\(c54e412b04b5328aafba8694cbbf005c_img.jpg\) Security](#) [!\[\]\(bd2b6ac7b99de9813666552a602ceeab_img.jpg\) Insights](#)[idcp-alura / aulas.ipynb](#) 

guilhermeonrails 4

9df9e9f · 5 days ago



8302 lines (8302 loc) · 682 KB

[idcp-alura / aulas.ipynb](#)[↑ Top](#)[Preview](#)[Code](#)[Blame](#)[Raw](#)

Aula 1 - Análise de Dados com Pandas

```
In [ ]: import pandas as pd
```

```
In [ ]: df = pd.read_csv("https://raw.githubusercontent.com/guilhermeonrails/data-")
```

```
In [ ]: df.head()
```

```
Out[ ]:   work_year  experience_level  employment_type  job_title  salary  salary_currency
          0        2025.0              SE                FT Solutions Engineer    214000           US
          1        2025.0              SE                FT Solutions Engineer    136000           US
          2        2025.0              MI                FT Data Engineer     158800           US
          3        2025.0              MI                FT Data Engineer     139200           US
          4        2025.0              EN                FT Data Engineer      90000           US
```

```
In [ ]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 133349 entries, 0 to 133348
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype  
 ---  --  
 0   work_year         133339 non-null   float64 
 1   experience_level 133349 non-null   object  
 2   employment_type   133349 non-null   object  
 3   job_title          133349 non-null   object  
 4   salary             133349 non-null   int64  
 5   salary_currency   133349 non-null   object  
 6   salary_in_usd     133349 non-null   int64  
 7   employee_residence 133349 non-null   object  
 8   remote_ratio       133349 non-null   int64  
 9   company_location   133349 non-null   object  
 10  company_size       133349 non-null   object  
dtypes: float64(1), int64(3), object(7)
memory usage: 11.2+ MB
```

```
In [ ]: df.describe()
```

```
Out[ ]:   work_year      salary  salary_in_usd  remote_ratio
          count  133339.000000  1.333490e+05  133349.000000  133349.000000
```

```
mean    2024.358770  1.632833e+05  157617.272098  20.905669
std      0.680627  2.173860e+05  74288.363097  40.590044
min      2020.000000  1.400000e+04  15000.000000  0.000000
25%     2024.000000  1.060200e+05  106000.000000  0.000000
50%     2024.000000  1.470000e+05  146206.000000  0.000000
75%     2025.000000  1.990000e+05  198000.000000  0.000000
max     2025.000000  3.040000e+07  800000.000000  100.000000
```

In []: df.shape

Out[]: (133349, 11)

```
In [ ]: linhas, colunas = df.shape[0], df.shape[1]
print('Linhas:', linhas)
print('Colunas:', colunas)
```

Linhos: 133349

Colunas: 11

In []: df.columns

```
Out[ ]: Index(['work_year', 'experience_level', 'employment_type', 'job_title',
               'salary', 'salary_currency', 'salary_in_usd', 'employee_residence',
               'remote_ratio', 'company_location', 'company_size'],
              dtype='object')
```

Renomeando as colunas do DataFrame

```
# Dicionário de renomeação
novos_nomes = {
    'work_year': 'ano',
    'experience_level': 'senioridade',
    'employment_type': 'contrato',
    'job_title': 'cargo',
    'salary': 'salario',
    'salary_currency': 'moeda',
    'salary_in_usd': 'usd',
    'employee_residence': 'residencia',
    'remote_ratio': 'remoto',
    'company_location': 'empresa',
    'company_size': 'tamanho_empresa'
}

# Aplicando renomeação
df.rename(columns=novos_nomes, inplace=True)

# Verificando resultado
df.head()
```

Out[]: ano senioridade contrato cargo salario moeda usd residencia re

0	2025.0	SE	FT	Solutions Engineer	214000	USD	214000	US
1	2025.0	SE	FT	Solutions Engineer	136000	USD	136000	US
2	2025.0	MI	FT	Data Engineer	158800	USD	158800	AU
3	2025.0	MI	FT	Data Engineer	139200	USD	139200	AU
4	2025.0	EN	FT	Data Engineer	90000	USD	90000	US

Analisando quais são as categorias das colunas categóricas

Nível de senioridade

```
In [ ]: # O método .value_counts() serve para contar quantas vezes cada valor único
df['senioridade'].value_counts()
```

Out[]: **count**

senioridade

SE	77241
MI	40465
EN	12443
EX	3200

dtype: int64

Sigla	Significado	Descrição
SE	Senior	Profissional experiente (nível sênior)
MI	Mid	Nível intermediário
EN	Entry	Iniciante ou júnior (entry-level)
EX	Executive	Executivo, liderança sênior (C-Level)

Tipo de contrato

```
In [ ]: df['contrato'].value_counts()
```

Out[]: **count**

contrato

FT	132563
-----------	--------

CT 394

PT 376

FL 16

dtype: int64

Sigla	Significado	Descrição
FT	Full-time	Tempo integral – trabalho padrão de 40h/semana ou período completo
PT	Part-time	Meio período – carga horária reduzida
CT	Contract	Contrato temporário ou por projeto – geralmente por prazo determinado
FL	Freelance	Freelancer – trabalho autônomo ou por demanda, sem vínculo formal

Regime de trabalho

```
In [ ]: df['remoto'].value_counts()
```

```
Out[ ]: count
```

remoto	
0	105312
100	27718
50	319

dtype: int64

O modelo remoto permite trabalhar de qualquer lugar sem precisar ir à empresa. O modelo presencial exige ir até o escritório todos os dias. O híbrido mistura os dois modelos. O modelo 'everywhere' é mais raro e permite trabalhar de qualquer lugar do mundo sem restrição de país ou cidade.

Sigla	Significado
0	Presencial
100	Remoto
50	Híbrido

Tamanho da empresa

```
In [ ]: df['tamanho_empresa'].value_counts()
```

Out[]:

count**tamanho_empresa**

M	129561
L	3574
S	214

dtype: int64

	Sigla	Significado	Descrição
	M	Medium	Empresa de tamanho médio
	L	Large	Empresa de tamanho grande
	S	Small	Empresa de tamanho pequeno

Modificando o nome das categorias:

In []:

```
senioridade = {
    'SE': 'senior',
    'MI': 'pleno',
    'EN': 'junior',
    'EX': 'executivo'
}
df['senioridade'] = df['senioridade'].replace(senioridade)
df['senioridade'].value_counts()
```

Out[]:

count**senioridade**

senior	77241
pleno	40465
junior	12443
executivo	3200

dtype: int64

In []:

```
contrato = {
    'FT': 'integral',
    'PT': 'parcial',
    'CT': 'contrato',
    'FL': 'freelancer'
}
df['contrato'] = df['contrato'].replace(contrato)
df['contrato'].value_counts()
```

Out[]:

count**contrato**

```
integral    132563
```

```
contrato     394
```

```
parcial      376
```

```
freelancer    16
```

dtype: int64

```
In [ ]: tamanho_empresa = {
    'L': 'grande',
    'S': 'pequena',
    'M':         'media'

}
df['tamanho_empresa'] = df['tamanho_empresa'].replace(tamanho_empresa)
df['tamanho_empresa'].value_counts()
```

```
Out[ ]:      count
```

tamanho_empresa

	count
media	129561

grande	3574
---------------	------

pequena	214
----------------	-----

dtype: int64

```
In [ ]: mapa_trabalho = {
    0: 'presencial',
    100: 'remoto',
    50: 'hibrido'
}

df['remoto'] = df['remoto'].replace(mapa_trabalho)
df['remoto'].value_counts()
```

```
Out[ ]:      count
```

remoto

	count
presencial	105312

remoto	27718
---------------	-------

hibrido	319
----------------	-----

dtype: int64

```
In [ ]: df.head()
```

```
Out[ 1]:    ano  senioridade  contrato      cargo  salario  moeda      ied  residencia
```

	ano	senioridade	contrato	cargo	moeda	residencia	remoto	empresa	tamanho
0	2025.0	senior	integral	Solutions Engineer	214000	USD	214000	US	
1	2025.0	senior	integral	Solutions Engineer	136000	USD	136000	US	
2	2025.0	pleno	integral	Data Engineer	158800	USD	158800	AU	presencial
3	2025.0	pleno	integral	Data Engineer	139200	USD	139200	AU	presencial
4	2025.0	junior	integral	Data Engineer	90000	USD	90000	US	presencial

Podemos também resumir as informações categóricas com o método `describe()`, exibindo a quantidade de categorias únicas, qual é categoria mais frequente e sua respectiva frequência:

In []: `df.describe(include='object')`

	senioridade	contrato	cargo	moeda	residencia	remoto	empresa	tamanho
count	133349	133349	133349	133349	133349	133349	133349	133349
unique	4	4	390	26	102	3	95	
top	senior	integral	Data Scientist	USD	US	presencial	US	
freq	77241	132563	17314	126140	119579	105312	119641	

Com isso já conseguimos responder algumas perguntas, como:

- Qual o nível de experiência mais comum na base de dados?
- Qual é o tipo de contrato mais frequente?
- Qual o cargo mais frequente na amostra?
- De qual país são a maioria dos profissionais da base?
- Qual é o país onde mais empresas da amostra estão sediadas?
- Qual o regime de trabalho mais comum?
- Qual é o tamanho mais comum das empresas na amostra?

O código é um passo a passo para entender, limpar e preparar a base de dados para análises mais profundas, facilitando a visualização de padrões e tendências no mercado de trabalho em ciência de dados.

Aula 2 - Preparação e limpeza dos Dados

In []: `df.isnull()`

Out[]:	ano	senioridade	contrato	cargo	salario	moeda	usd	residencia	ren
	0	False	False	False	False	False	False	False	False
	1	False	False	False	False	False	False	False	False
	2	False	False	False	False	False	False	False	False
	3	False	False	False	False	False	False	False	False
	4	False	False	False	False	False	False	False	False

	133344	False	False	False	False	False	False	False	False
	133345	False	False	False	False	False	False	False	False
	133346	False	False	False	False	False	False	False	False
	133347	False	False	False	False	False	False	False	False
	133348	False	False	False	False	False	False	False	False

133349 rows × 11 columns

In []: df.head()

Out[]:	ano	senioridade	contrato	cargo	salario	moeda	usd	residencia	
	0	2025.0	senior	integral	Solutions Engineer	214000	USD	214000	US
	1	2025.0	senior	integral	Solutions Engineer	136000	USD	136000	US
	2	2025.0	pleno	integral	Data Engineer	158800	USD	158800	AU pr
	3	2025.0	pleno	integral	Data Engineer	139200	USD	139200	AU pr
	4	2025.0	junior	integral	Data Engineer	90000	USD	90000	US pr

In []: df.isnull().sum()

Out[]:	0
	ano 10
	senioridade 0
	contrato 0
	cargo 0
	salario 0
	moeda 0

```
    usd      0
    residencia      0
    remoto      0
    empresa      0
    tamanho_empresa      0
```

dtype: int64

In []: df['ano'].unique()

Out[]: array([2025., nan, 2024., 2022., 2023., 2020., 2021.])

In []: df[df.isnull().any(axis=1)]

	ano	senioridade	contrato	cargo	salario	moeda	usd	residencia
5588	NaN	senior	integral	Product Manager	184500	USD	184500	US
59692	NaN	pleno	integral	Engineer	110000	USD	110000	DE
59710	NaN	junior	integral	Data Scientist	208800	USD	208800	US
59759	NaN	senior	integral	Software Engineer	135000	USD	135000	US
59789	NaN	senior	integral	Engineer	112000	USD	112000	US
131000	NaN	senior	integral	Machine Learning Engineer	163800	USD	163800	US
131006	NaN	senior	integral	Data Analytics Manager	204500	USD	204500	US
133054	NaN	junior	integral	Data Scientist	40000	USD	40000	JP
133281	NaN	pleno	integral	Machine Learning Engineer	180000	PLN	46597	PL
133317	NaN	pleno	integral	Data Scientist	130000	USD	130000	US

In []: import numpy as np

```
# Criação de um dataframe de teste para usar de exemplo
df_salarios = pd.DataFrame({
    'nome': ["Ana", "Bruno", "Carlos", "Daniele", "Val"],
    'salario': [4000, np.nan, 5000, np.nan, 100000]
})
```

```
# calcula a média salarial e substitui os nulos pela média e arredonda para cima
df_salarios['salario_media'] = df_salarios['salario'].fillna(df_salarios['salario'].mean().round(2))

'''Calcula mediana e substitui os nulos pela mediana
'''

df_salarios['salario_mediana'] = df_salarios['salario'].fillna(df_salarios['salario'].median())

df_salarios
```

Out[]:

	nome	salario	salario_media	salario_mediana
0	Ana	4000.0	4000.00	4000.0
1	Bruno	NaN	36333.33	5000.0
2	Carlos	5000.0	5000.00	5000.0
3	Daniele	NaN	36333.33	5000.0
4	Val	100000.0	100000.00	100000.0

In []:

```
df_temperaturas = pd.DataFrame({
    "Dia": ["Segunda", "Terça", "Quarta", "Quinta", "Sexta"],
    "Temperatura": [30, np.nan, np.nan, 28, 27]
})

df_temperaturas["preenchido_ffill"] = df_temperaturas["Temperatura"].ffill()
df_temperaturas
```

Out[]:

	Dia	Temperatura	preenchido_ffill
0	Segunda	30.0	30.0
1	Terça	NaN	30.0
2	Quarta	NaN	30.0
3	Quinta	28.0	28.0
4	Sexta	27.0	27.0

In []:

```
df_temperaturas = pd.DataFrame({
    "Dia": ["Segunda", "Terça", "Quarta", "Quinta", "Sexta"],
    "Temperatura": [30, np.nan, np.nan, 28, 27]
})

df_temperaturas["preenchido_bfill"] = df_temperaturas["Temperatura"].bfill()
df_temperaturas
```

Out[]:

	Dia	Temperatura	preenchido_bfill
0	Segunda	30.0	30.0
1	Terça	NaN	28.0
2	Quarta	NaN	28.0
3	Quinta	28.0	28.0

5	Quinta	20.0	20.0
4	Sexta	27.0	27.0

```
In [ ]: df_cidades = pd.DataFrame({
    'nome': ["Ana", "Bruno", "Carlos", "Daniele", "Val"],
    'cidade': ["São Paulo", np.nan, "Curitiba", np.nan, "Belém"]
})

df_cidades['cidade_preenchida'] = df_cidades["cidade"].fillna("Não informado")
display(df_cidades)
```

	nome	cidade	cidade_preenchida
0	Ana	São Paulo	São Paulo
1	Bruno	NaN	Não informado
2	Carlos	Curitiba	Curitiba
3	Daniele	NaN	Não informado
4	Val	Belém	Belém

```
In [ ]: df_limpo = df.dropna()
```

```
In [ ]: df_limpo.isnull().sum()
```

Out[]: 0

ano	0
senioridade	0
contrato	0
cargo	0
salario	0
moeda	0
usd	0
residencia	0
remoto	0
empresa	0
tamanho_empresa	0

dtype: int64

```
In [ ]: df_limpo.head()
```

Out[]: ano senioridade contrato cargo salario moeda usd residencia

0	2025.0	senior	integral	Solutions Engineer	214000	USD	214000	US	re
1	2025.0	senior	integral	Solutions Engineer	136000	USD	136000	US	re
2	2025.0	pleno	integral	Data Engineer	158800	USD	158800	AU	pres
3	2025.0	pleno	integral	Data Engineer	139200	USD	139200	AU	pres
4	2025.0	junior	integral	Data Engineer	90000	USD	90000	US	pres

In []: df_limpo.info()

```
<class 'pandas.core.frame.DataFrame'>
Index: 133339 entries, 0 to 133348
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   ano              133339 non-null   float64
 1   senioridade      133339 non-null   object 
 2   contrato         133339 non-null   object 
 3   cargo             133339 non-null   object 
 4   salario           133339 non-null   int64  
 5   moeda             133339 non-null   object 
 6   usd               133339 non-null   int64  
 7   residencia        133339 non-null   object 
 8   remoto            133339 non-null   object 
 9   empresa            133339 non-null   object 
 10  tamanho_empresa   133339 non-null   object 
dtypes: float64(1), int64(2), object(8)
memory usage: 12.2+ MB
```

In []: df_limpo = df_limpo.assign(ano = df_limpo['ano'].astype('int64'))

Aula 3 - Visualização de Dados

Aprendendo a criar gráficos estatísticos para explorar e comunicar informações presentes nos dados. Histogramas, boxplots, barras, etc.

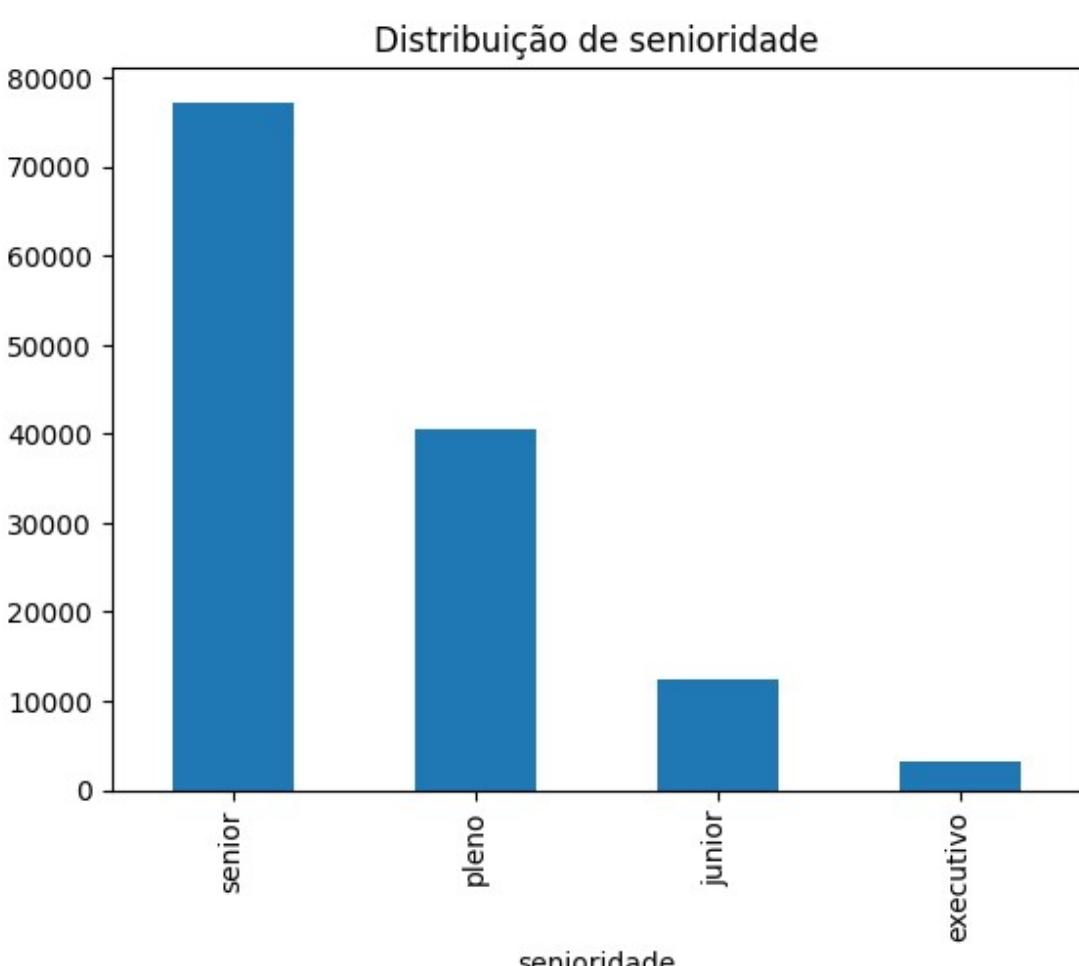
In []: df_limpo.head()

	ano	senioridade	contrato	cargo	salario	moeda	usd	residencia	remoto
0	2025	senior	integral	Solutions Engineer	214000	USD	214000	US	re
1	2025	senior	integral	Solutions Engineer	136000	USD	136000	US	re
2	2025	pleno	integral	Data Engineer	158800	USD	158800	AU	pres
3	2025	pleno	integral	Data Engineer	139200	USD	139200	AU	pres

	ano	periodo	integrat	Engineer	199200	USD	199200	mo	pre:
4	2025	junior	integral	Data Engineer	90000	USD	90000	US	pre:

```
In [ ]: df_limpo['senioridade'].value_counts().plot(kind='bar', title="Distribuição de senioridade")
```

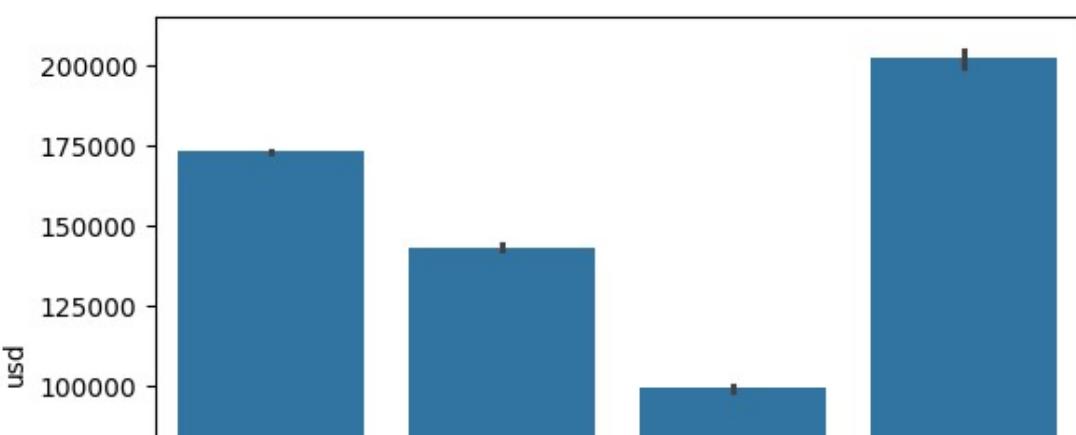
Out[]: <Axes: title={'center': 'Distribuição de senioridade'}, xlabel='senioridade'>

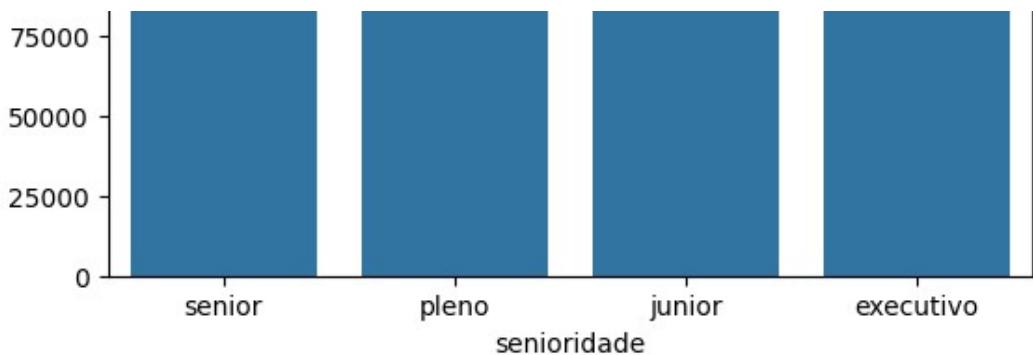


```
In [ ]: import seaborn as sns
```

```
In [ ]: sns.barplot(data=df_limpo, x='senioridade', y='usd')
```

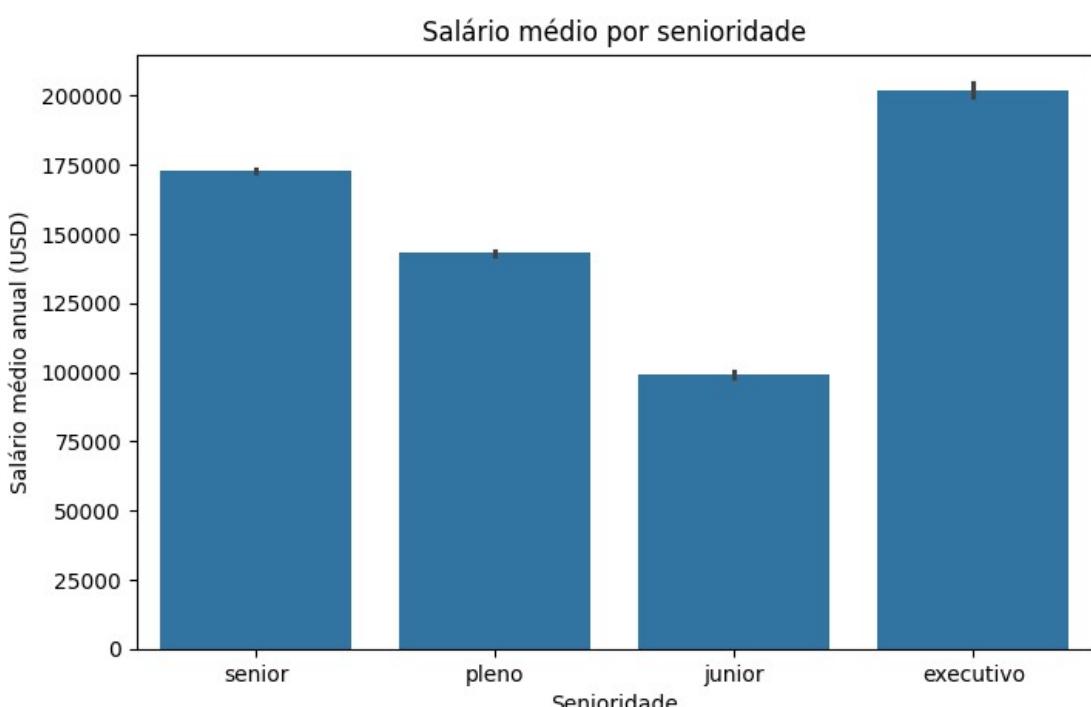
Out[]: <Axes: xlabel='senioridade', ylabel='usd'>





```
In [ ]: import matplotlib.pyplot as plt
```

```
In [ ]: plt.figure(figsize=(8,5))
sns.barplot(data=df_limpo, x='senioridade', y='usd')
plt.title("Salário médio por senioridade")
plt.xlabel("Senioridade")
plt.ylabel("Salário médio anual (USD)")
plt.show()
```



```
In [ ]: df_limpo.groupby('senioridade')['usd'].mean().sort_values(ascending=False)
```

```
Out[ ]: usd
```

senioridade	usd
executivo	202027.667813
senior	172850.838301
pleno	143044.845979
júnior	99034.963267

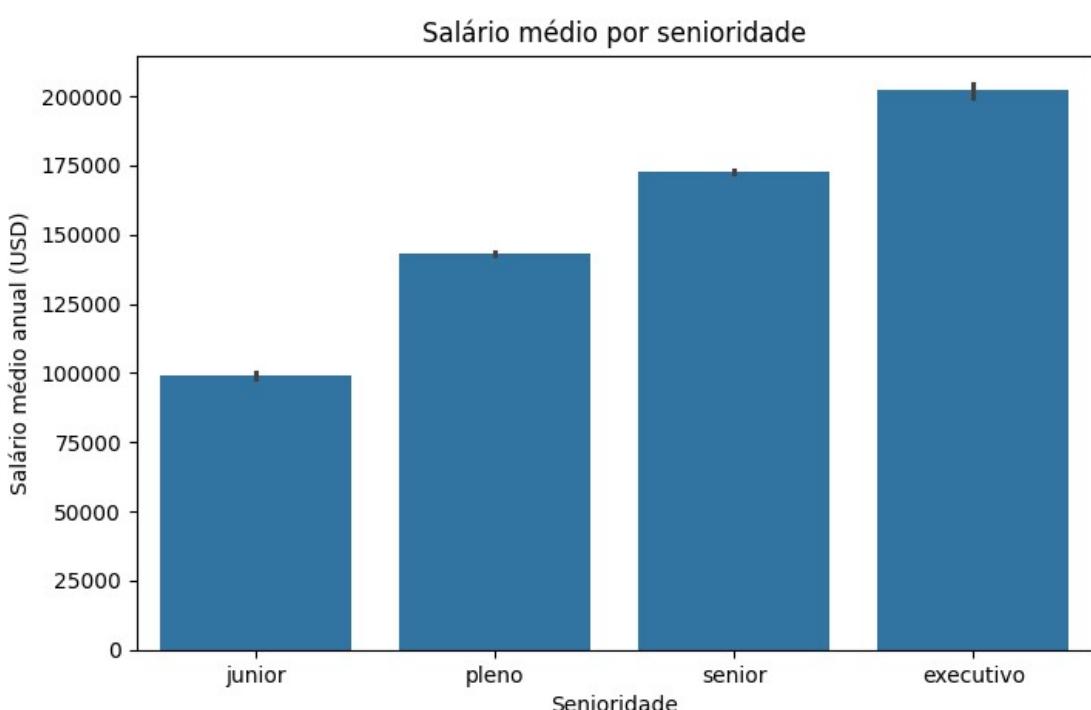
dtype: float64

```
In [ ]: ordem = df_limpo.groupby('senioridade')['usd'].mean().sort_values(ascending=True)
```

```
In [ ]: ordem
```

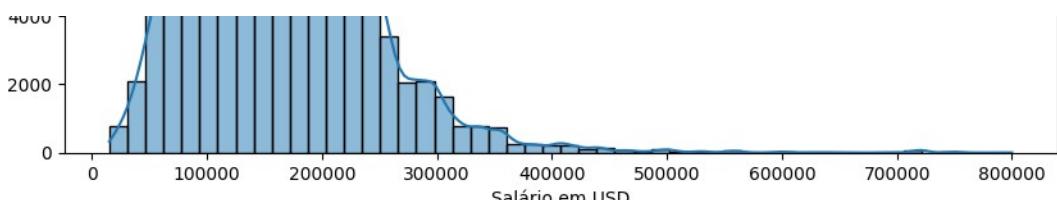
```
Out[ ]: Index(['junior', 'pleno', 'senior', 'executivo'], dtype='object', name='senioridade')
```

```
In [ ]: plt.figure(figsize=(8,5))
sns.barplot(data=df_limpo, x='senioridade', y='usd', order=ordem)
plt.title("Salário médio por senioridade")
plt.xlabel("Senioridade")
plt.ylabel("Salário médio anual (USD)")
plt.show()
```



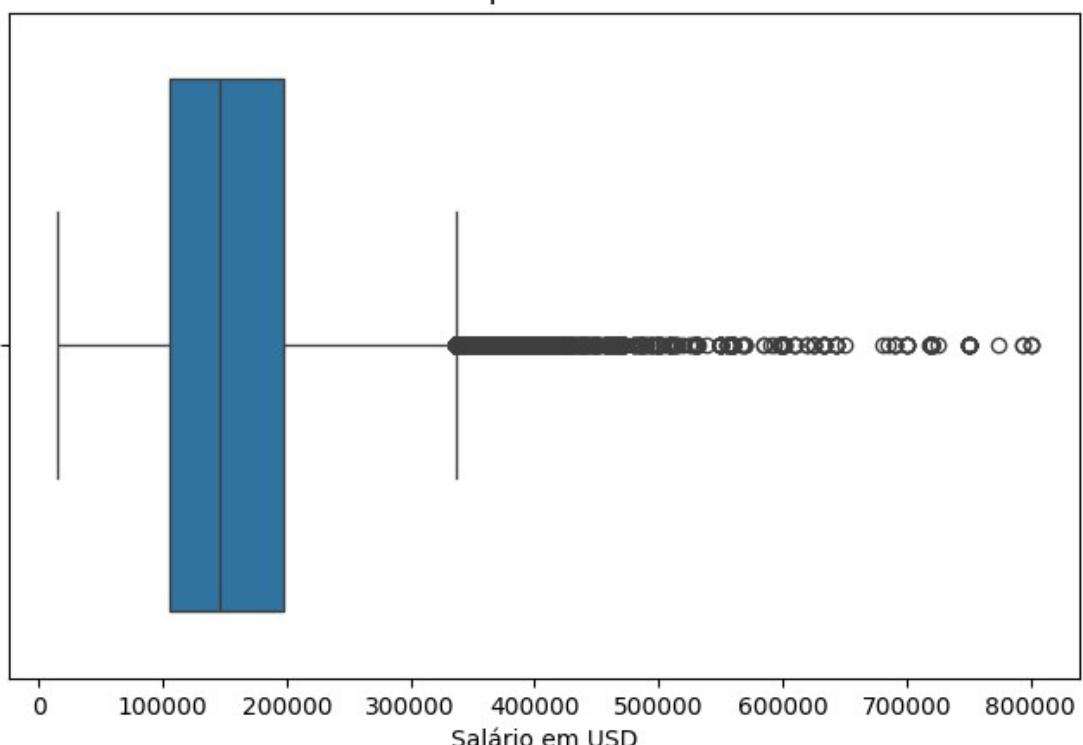
```
In [ ]: plt.figure(figsize=(10,5))
sns.histplot(df_limpo['usd'], bins = 50, kde=True)
plt.title("Distribuição dos salários anuais")
plt.xlabel("Salário em USD")
plt.ylabel("Frequência")
plt.show()
```





```
In [ ]: plt.figure(figsize=(8,5))
sns.boxplot(x=df_limpo['usd'])
plt.title("Boxplot Salário")
plt.xlabel("Salário em USD")
plt.show()
```

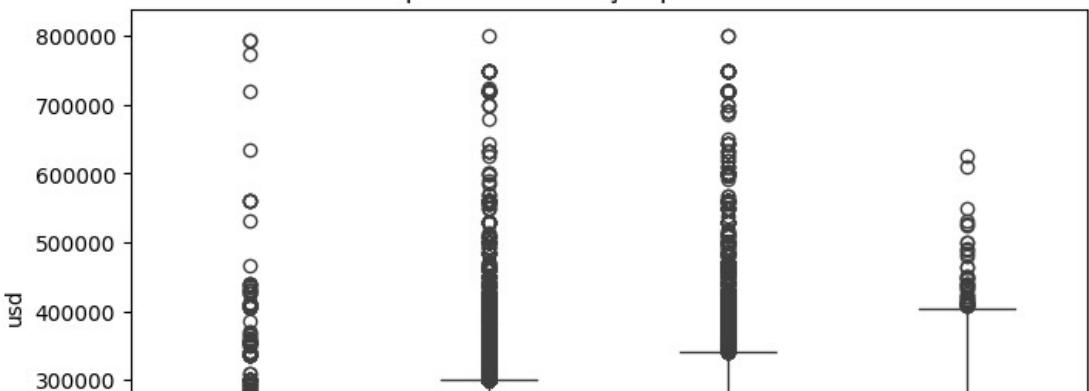
Boxplot Salário

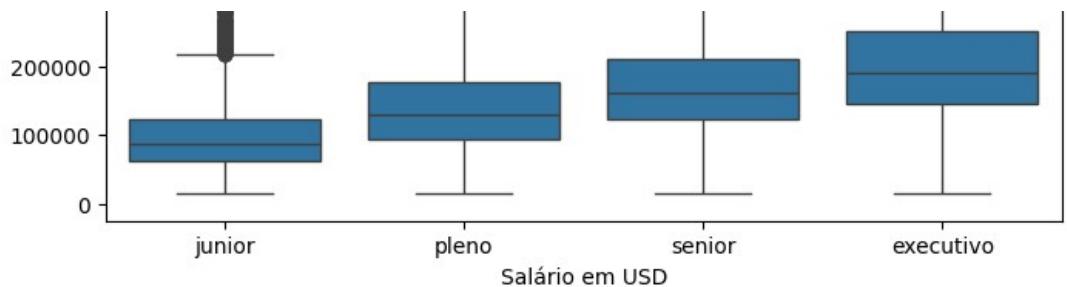


```
In [ ]: ordem_senioridade = ['junior', 'pleno', 'senior', 'executivo']

plt.figure(figsize=(8,5))
sns.boxplot(x='senioridade', y='usd', data=df_limpo, order=ordem_senioridade)
plt.title("Boxplot da distribuição por senioridade")
plt.xlabel("Salário em USD")
plt.show()
```

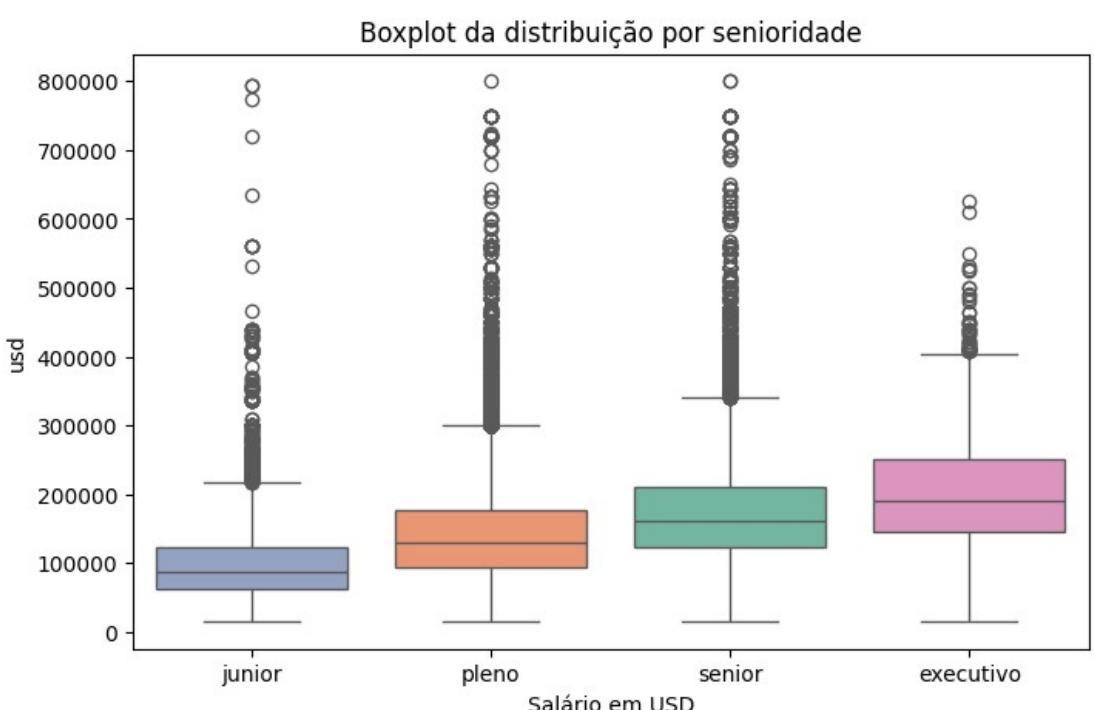
Boxplot da distribuição por senioridade





```
In [ ]: ordem_senioridade = ['junior', 'pleno', 'senior', 'executivo']

plt.figure(figsize=(8,5))
sns.boxplot(x='senioridade', y='usd', data=df_limpo, order=ordem_senioridade)
plt.title("Boxplot da distribuição por senioridade")
plt.xlabel("Salário em USD")
plt.show()
```



```
In [ ]: import plotly.express as px
```

```
In [ ]: # prompt: Crie um gráfico de média salarial por senioridade em barras usarámos o plotly express

senioridade_media_salario = df_limpo.groupby('senioridade')['usd'].mean()

fig = px.bar(senioridade_media_salario,
              x='senioridade',
              y='usd',
              title='Média Salarial por Senioridade',
              labels={'senioridade': 'Nível de Senioridade', 'usd': 'Média'})

fig.show()
```

```
In [ ]: remoto_contagem = df_limpo['remoto'].value_counts().reset_index()
remoto_contagem.columns = ['tipo_trabalho', 'quantidade']
```

```

fig = px.pie(remoto_contagem,
              names='tipo_trabalho',
              values='quantidade',
              title='Proporção dos tipos de trabalho'

)

fig.show()

```

```

In [ ]:
remoto_contagem = df_limpo['remoto'].value_counts().reset_index()
remoto_contagem.columns = ['tipo_trabalho', 'quantidade']

fig = px.pie(remoto_contagem,
              names='tipo_trabalho',
              values='quantidade',
              title='Proporção dos tipos de trabalho',
              hole=0.5
)

fig.show()

```

```

In [ ]:
remoto_contagem = df_limpo['remoto'].value_counts().reset_index()
remoto_contagem.columns = ['tipo_trabalho', 'quantidade']

fig = px.pie(remoto_contagem,
              names='tipo_trabalho',
              values='quantidade',
              title='Proporção dos tipos de trabalho',
              hole=0.5
)
fig.update_traces(textinfo='percent+label')
fig.show()

```

```
In [ ]:
```

```
In [ ]: df.head()
```

	ano	senioridade	contrato	cargo	salario	moeda	usd	residencia
0	2025.0	senior	integral	Solutions Engineer	214000	USD	214000	US
1	2025.0	senior	integral	Solutions Engineer	136000	USD	136000	US
2	2025.0	pleno	integral	Data Engineer	158800	USD	158800	AU pr
3	2025.0	pleno	integral	Data Engineer	139200	USD	139200	AU pr
4	2025.0	junior	integral	Data Engineer	90000	USD	90000	US pr

```
In [ ]: pip install pycountry
```

```
Requirement already satisfied: pycountry in /usr/local/lib/python3.11/dist-packages (24.6.1)
```

```
In [ ]: import pycountry
```

```
# Função para converter ISO-2 para ISO-3
def iso2_to_iso3(code):
    try:
        return pycountry.countries.get(alpha_2=code).alpha_3
    except:
        return None

# Criar nova coluna com código ISO-3
df_limpo['residencia_iso3'] = df_limpo['residencia'].apply(iso2_to_iso3)

# Calcular média salarial por país (ISO-3)
df_ds = df_limpo[df_limpo['cargo'] == 'Data Scientist']
media_ds_pais = df_ds.groupby('residencia_iso3')['usd'].mean().reset_index()

# Gerar o mapa
fig = px.choropleth(media_ds_pais,
                      locations='residencia_iso3',
                      color='usd',
                      color_continuous_scale='rdylgn',
                      title='Salário médio de Cientista de Dados por país',
                      labels={'usd': 'Salário médio (USD)', 'residencia_iso3': 'País'})

fig.show()
```

```
In [ ]: df_limpo.head()
```

	ano	senioridade	contrato	cargo	salario	moeda	usd	residencia	re
0	2025	senior	integral	Solutions Engineer	214000	USD	214000	US	re
1	2025	senior	integral	Solutions Engineer	136000	USD	136000	US	re
2	2025	pleno	integral	Data Engineer	158800	USD	158800	AU	pres
3	2025	pleno	integral	Data Engineer	139200	USD	139200	AU	pres
4	2025	junior	integral	Data Engineer	90000	USD	90000	US	pres

```
In [ ]: df_limpo.to_csv('dados-imersao-final.csv', index=False)
```

Aula 4 - Construindo um Dashboard com Streamlit

Aprender a usar a biblioteca Streamlit para a criação de um dashboard interativo simples, que permite visualizar dados filtrados e gerar gráficos de forma prática.

<https://dashboard-salarios-dados.streamlit.app/>

1. Criar o ambiente virtual:

```
python3 -m venv .venv
```

2. Ativar o ambiente virtual em Windows:

```
.venv\Scripts\Activate
```

3. Ativar o ambiente virtual em MAC/LINUX:

```
source .venv/bin/activate
```

4. Criar um arquivo chamado requirements.txt e adicionar os pacotes necessários

```
pandas==2.2.3  
streamlit==1.44.1  
plotly==5.24.1
```

5. Instalar as bibliotecas necessárias

```
pip install -r requirements.txt
```

6. Criar a Interface do Dashboard com Streamlit

7. Realizar o deploy do Dashboard no Streamlit Cloud: <https://streamlit.io/cloud>

```
In [ ]:  
import streamlit as st  
import pandas as pd  
import plotly.express as px  
  
# --- Configuração da Página ---  
# Define o título da página, o ícone e o Layout para ocupar a Largura inteira  
st.set_page_config(  
    page_title="Dashboard de Salários na Área de Dados",  
    page_icon="📊",  
    layout="wide",  
)  
  
# --- Carregamento dos dados ---  
df = pd.read_csv("https://raw.githubusercontent.com/vqrca/dashboard_salarios/branch/main/dados.csv")  
  
# --- Barra Lateral (Filtros) ---  
st.sidebar.header("🔍 Filtros")  
  
# Filtro de Ano  
anos_disponiveis = sorted(df['ano'].unique())
```

```
anos_selecionados = st.sidebar.multiselect("Ano", anos_disponiveis, default=2023)

# Filtro de Senioridade
senioridades_disponiveis = sorted(df['senioridade'].unique())
senioridades_selecionadas = st.sidebar.multiselect("Senioridade", senioridades_disponiveis)

# Filtro por Tipo de Contrato
contratos_disponiveis = sorted(df['contrato'].unique())
contratos_selecionados = st.sidebar.multiselect("Tipo de Contrato", contratos_disponiveis)

# Filtro por Tamanho da Empresa
tamanhos_disponiveis = sorted(df['tamanho_empresa'].unique())
tamanhos_selecionados = st.sidebar.multiselect("Tamanho da Empresa", tamanhos_disponiveis)

# --- Filtragem do DataFrame ---
# O dataframe principal é filtrado com base nas seleções feitas na barra lateral
df_filtrado = df[
    (df['ano'].isin(anos_selecionados)) &
    (df['senioridade'].isin(senioridades_selecionadas)) &
    (df['contrato'].isin(contratos_selecionados)) &
    (df['tamanho_empresa'].isin(tamanhos_selecionados))
]

# --- Conteúdo Principal ---
st.title("📊 Dashboard de Análise de Salários na Área de Dados")
st.markdown("Explore os dados salariais na área de dados nos últimos anos.")

# --- Métricas Principais (KPIs) ---
st.subheader("Métricas gerais (Salário anual em USD)")

if not df_filtrado.empty:
    salario_medio = df_filtrado['usd'].mean()
    salario_maximo = df_filtrado['usd'].max()
    total_registros = df_filtrado.shape[0]
    cargo_mais_frequente = df_filtrado["cargo"].mode()[0]
else:
    salario_medio, salario_mediano, salario_maximo, total_registros, cargo_mais_frequente = None, None, None, None, None

col1, col2, col3, col4 = st.columns(4)
col1.metric("Salário médio", f"${salario_medio:,.0f}")
col2.metric("Salário máximo", f"${salario_maximo:,.0f}")
col3.metric("Total de registros", f"{total_registros:,}")
col4.metric("Cargo mais frequente", cargo_mais_frequente)

st.markdown("---")

# --- Análises Visuais com Plotly ---
st.subheader("Gráficos")

col_graf1, col_graf2 = st.columns(2)

with col_graf1:
    if not df_filtrado.empty:
        top_cargos = df_filtrado.groupby('cargo')['usd'].mean().nlargest(10)
        grafico_cargos = px.bar(
            top_cargos,
            x='usd',
            y='cargo',
            orientation='h',
            title="Top 10 cargos por salário médio",
            labels={'usd': 'Média salarial anual (USD)', 'cargo': ''}
        )
        st.plotly_chart(grafico_cargos)
```

```
        )
        grafico_cargos.update_layout(title_x=0.1, yaxis={'categoryorder': 'st.plotly_chart(grafico_cargos, use_container_width=True)
    else:
        st.warning("Nenhum dado para exibir no gráfico de cargos.")

with col_graf2:
    if not df_filtrado.empty:
        grafico_hist = px.histogram(
            df_filtrado,
            x='usd',
            nbins=30,
            title="Distribuição de salários anuais",
            labels={'usd': 'Faixa salarial (USD)', 'count': ''})
    )
    grafico_hist.update_layout(title_x=0.1)
    st.plotly_chart(grafico_hist, use_container_width=True)
else:
    st.warning("Nenhum dado para exibir no gráfico de distribuição.")

col_graf3, col_graf4 = st.columns(2)

with col_graf3:
    if not df_filtrado.empty:
        remoto_contagem = df_filtrado['remoto'].value_counts().reset_index()
        remoto_contagem.columns = ['tipo_trabalho', 'quantidade']
        grafico_remoto = px.pie(
            remoto_contagem,
            names='tipo_trabalho',
            values='quantidade',
            title='Proporção dos tipos de trabalho',
            hole=0.5
        )
        grafico_remoto.update_traces(textinfo='percent+label')
        grafico_remoto.update_layout(title_x=0.1)
        st.plotly_chart(grafico_remoto, use_container_width=True)
    else:
        st.warning("Nenhum dado para exibir no gráfico dos tipos de trabalho")

with col_graf4:
    if not df_filtrado.empty:
        df_ds = df_filtrado[df_filtrado['cargo'] == 'Data Scientist']
        media_ds_pais = df_ds.groupby('residencia_iso3')['usd'].mean().reset_index()
        grafico_paises = px.choropleth(media_ds_pais,
            locations='residencia_iso3',
            color='usd',
            color_continuous_scale='rdylgn',
            title='Salário médio de Cientista de Dados por país',
            labels={'usd': 'Salário médio (USD)', 'residencia_iso3': 'País'}
        )
        grafico_paises.update_layout(title_x=0.1)
        st.plotly_chart(grafico_paises, use_container_width=True)
    else:
        st.warning("Nenhum dado para exibir no gráfico de países.")
```