# Python Insights - Analisando Dados com Python

## Case - Cancelamento de Clientes

Você foi contratado por uma empresa com mais de 800 mil clientes para um projeto de Dados. Recentemente a empresa percebeu que da sua base total de clientes, a maioria são clientes inativos, ou seja, que já cancelaram o serviço.

Precisando melhorar seus resultados ela quer conseguir entender os principais motivos desses cancelamentos e quais as ações mais eficientes para reduzir esse número.

Base de dados e arquivos:
https://drive.google.com/drive/folders/1uDesZePdkhiraJmiyeZ-w5tfc8XsNYFZ?usp=drive_link

In [1]:
```
# Passo a passo
# Passo 1: Importar a base de dados
# Passo 2: Visualizar os dados (entender a base + identificar probl
# Passo 3: Descartar os dados que não são úteis para minha resoluçã
# Passo 4: Analisar os dados depois de limpos e começar o raciocini
# Passo 5: Montar os gráficos para vê o que impacta no cancelamento
```

In [2]:
```
# Passo 1: Importar a base de dados

import pandas as pd   # importando a biblioteca pandas


tabela = pd.read_csv('cancelamentos.csv')  # importando os dados

# Passo 2: Visualizar os dados (entender a base + identificar probl
display(tabela)
```

| | CustomerID | idade | sexo | tempo_como_cliente | frequencia_uso | lig |
|---|---|---|---|---|---|---|
| **0** | 2.0 | 30.0 | Female | 39.0 | 14.0 | |
| **1** | 3.0 | 65.0 | Female | 49.0 | 1.0 | |
| **2** | 4.0 | 55.0 | Female | 14.0 | 4.0 | |
| **3** | 5.0 | 58.0 | Male | 38.0 | 21.0 | |
| **4** | 6.0 | 23.0 | Male | 32.0 | 20.0 | |
| **...** | ... | ... | ... | ... | ... | |
| **881661** | 449995.0 | 42.0 | Male | 54.0 | 15.0 | |
| **881662** | 449996.0 | 25.0 | Female | 8.0 | 13.0 | |
| **881663** | 449997.0 | 26.0 | Male | 35.0 | 27.0 | |
| **881664** | 449998.0 | 28.0 | Male | 55.0 | 14.0 | |
| **881665** | 449999.0 | 31.0 | Male | 48.0 | 20.0 | |

881666 rows × 12 columns

In [3]:
```
# Passo 3: Descartar os dados que não são úteis para minha resoluçã

# CustomerID não interfere na taxa de cancelamento, logo não será n

tabela = tabela.drop(columns='CustomerID')

# verificar se existe dados vazios
display(tabela.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 881666 entries, 0 to 881665
Data columns (total 11 columns):
 #   Column                  Non-Null Count   Dtype  
---  ------                  --------------   -----  
 0   idade                   881664 non-null  float64
 1   sexo                    881664 non-null  object 
 2   tempo_como_cliente      881663 non-null  float64
 3   frequencia_uso          881663 non-null  float64
 4   ligacoes_callcenter     881664 non-null  float64
 5   dias_atraso             881664 non-null  float64
 6   assinatura              881661 non-null  object 
 7   duracao_contrato        881663 non-null  object 
 8   total_gasto             881664 non-null  float64
 9   meses_ultima_interacao  881664 non-null  float64
 10  cancelou                881664 non-null  float64
dtypes: float64(8), object(3)
memory usage: 74.0+ MB
None
```

In [4]:
```
# visualizando quais são os dados que possuem valores vazios
display(tabela[tabela.isna().any(axis=1)])
```

| | idade | sexo | tempo_como_cliente | frequencia_uso | ligacoes_callcen |
|---|---|---|---|---|---|
| **11** | 52.0 | Female | 21.0 | 6.0 | |
| **14** | 24.0 | Male | 4.0 | 9.0 | |
| **17** | 47.0 | Male | 41.0 | NaN | |
| **18** | 24.0 | Male | 44.0 | 13.0 | |
| **23** | 27.0 | Female | NaN | 8.0 | |
| **199295** | NaN | NaN | NaN | NaN | N |
| **640128** | NaN | NaN | NaN | NaN | N |

In [5]:
```
# excluir da tabela os valores vazios, já que são poucos valores
tabela = tabela.dropna()

# tabela atualizada sem nenhum valor vazio
display(tabela.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 881659 entries, 0 to 881665
Data columns (total 11 columns):
 #   Column                  Non-Null Count   Dtype  
---  ------                  --------------   -----  
 0   idade                   881659 non-null  float64
 1   sexo                    881659 non-null  object 
 2   tempo_como_cliente      881659 non-null  float64
 3   frequencia_uso          881659 non-null  float64
 4   ligacoes_callcenter     881659 non-null  float64
 5   dias_atraso             881659 non-null  float64
 6   assinatura              881659 non-null  object 
 7   duracao_contrato        881659 non-null  object 
 8   total_gasto             881659 non-null  float64
 9   meses_ultima_interacao  881659 non-null  float64
 10  cancelou                881659 non-null  float64
dtypes: float64(8), object(3)
memory usage: 80.7+ MB
None
```

In [6]:
```
# Passo 4: Analisar os dados depois de limpos e começar o raciocini
display(tabela['cancelou'].value_counts()) # visualizar quantos can

display(tabela['cancelou'].value_counts(normalize=True).map("{:.1%}"
```

```
cancelou
1.0    499993
0.0    381666
Name: count, dtype: int64
cancelou
1.0    56.7%
0.0    43.3%
Name: proportion, dtype: object
```

In [7]:
```
# Passo 5: Montar os gráficos para vê o que impacta no cancelamento
import plotly.express as px
```

```
# montar os gráficos em dois passos: criar o gráfico e exibir o grá
# criar o gráfico
grafico = px.histogram(tabela, x='duracao_contrato', color='cancelou
# montar o gráfico
grafico.show()
```

In [8]:
```
# criar os graficos relacionando com todos os dados

for coluna in tabela.columns:
    grafico = px.histogram(tabela, x=coluna, color='cancelou', text_

    grafico.show()
```

In [9]:
```
# Levantar os pontos chaves da observação:
    # P1: Pessoas com plano mensal cancelam

    # P2: dias_atraso maior que 20 gera cancelamento

    # P3: ligacoes_callcenter mais de 4x gera cancelamento
```

In [10]:
```
# visualizar como ficaria a taxa de cancelamento se esses problemas

# resolvendo o P1:
tabela = tabela[tabela['duracao_contrato'] != 'Monthly'] # taxa cai

# resolvendo P2:
tabela = tabela[tabela['dias_atraso'] <= 20] # taxa caiu para 35%

# resolvendo P3:
tabela = tabela[tabela['ligacoes_callcenter'] <= 4] # taxa caiu para


display(tabela['cancelou'].value_counts(normalize=True).map("{:.1%}"
```

```
cancelou
0.0    81.6%
1.0    18.4%
Name: proportion, dtype: object
```