Projeto Semantix: Análise de Dados de Transações

Índice

1. Compreensão do Negócio (Business Understanding):

O objetivo deste projeto é identificar variáveis que possam indicar sinais de inadimplência em um banco de dados pertencente a uma empresa de concessão de crédito. A análise destes dados tem o objetivo de colaborar para a identificação de padrões e tendências que possam contribuir para a diminuição significativa do índice de inadimplência à esta instituição, influenciando na melhora dos resultados da mesma.

2. Compreensão dos Dados (Data Understanding):

Será utilizado para a realização deste projeto um banco de dados adquirido via GitHub, adquirido via download e em formato .csv. Foram fornecidas 15 variáveis. O significado de cada uma dessas variáveis se encontra na tabela abaixo:

Dicionário de dados

Os dados estão dispostos em uma tabela com uma linha para cada cliente, e uma coluna para cada variável armazenando as características desses clientes. Há uma cópia do dicionário de dados (explicação dessas variáveis) abaixo:

Variável	Descrição	Tipo
id	Chave de registro do cliente	int
default	Flag de adimplência/inadimplência	int
idade	Idade do cliente	int
sexo	Sexo do cliente	object
dependentes	Número de dependentes	int
escolaridade	Nível de escolaridade	object
estado_civil	Se cliente é casado, solteiro, UE, viúvo	object
salário anual	Renda bruta do cliente	object
tipo_cartao	Tipo do cartão de crédito	object
meses_de_relacionamento	Há quanto tempo é cliente do banco	int
qtd_produtos	Quantidade de produtos que o cliente possui	int

Variável	Descrição	Tipo
iteracoes_12m	Quantas vezes houve contato entre cliente e instituição	int
meses_inativo_12m	Quantos meses em 12 meses não houveram movimentações financeiras	int
limite_credito	Limite de crédito disponível ao cliente pela instituição	float
valor_transacoes_12m	Quantidade de capital movimentada em 12 meses	float
qtd_transacoes_12m	Quantidade de transações realizadas pelo cliente em 12 meses	int

3. Limpeza e Pré-processamento de Dados

Nessa etapa realizamos tipicamente as seguintes operações com os dados:

- Seleção: Os dados já estão pré-selecionados.
- Limpeza: Os dados faltantes serão tratados nas células de código abaixo.
- Construção: Foi construída as variável 'cat_qtd_transacoes_12m' para uma avaliação mais aprofundada dos dados.
- Integração: Temos apenas uma fonte de dados, não é necessário integração.
- Formatação: Os dados já se encontram em formatos úteis? R. Alguns dados serão tratados para formato mais útil.

In [1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns # data visualization
sns.set_theme(style="darkgrid")
import matplotlib.pyplot as plt # data visualization

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A NumPy version >
=1.16.5 and <1.23.0 is required for this version of SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
/kaggle/input/analise-inadimplencia/Python_M10_support material.csv
```

In [2]:
```
# Lendo os dados, retirando os dados faltantes e removendo a coluna 'id',
# que não tem utilidade na análise dos dados:

df = pd.read_csv('/kaggle/input/analise-inadimplencia/Python_M10_support material.csv',
na_values = 'na')
df.dropna(inplace=True)
df.drop(columns=['id'], inplace = True)
df.head()
```

Out[2]:

	default	idade	sexo	dependentes	escolaridade	estado_civil	salario_anual	tipo_cartao	meses_de_relaciona
0	0	45	M	3	ensino medio	casado	$60K-80K$	blue	
1	0	49	F	5	mestrado	solteiro	menos que $40K	blue	
2	0	51	M	3	mestrado	casado	$80K-120K$	blue	
4	0	40	M	3	sem educacao formal	casado	$60K-80K$	blue	
5	0	44	M	2	mestrado	casado	$40K-60K$	blue	

In [3]:

```
# Tratando os valores para formato adequado:

df['valor_transacoes_12m'] = df['valor_transacoes_12m'].apply(lambda x: float(x.replace(".", "").replace(",", ".")))
df['limite_credito'] = df['limite_credito'].apply(lambda x: float(x.replace(".", "").replace(",", ".")))
```

4. Exploração de Dados (Análise Exploratória):

Abaixo serão criadas estatísticas descritivas e visualizações, como gráficos e histogramas, para identificar padrões e tendências para resolução do problema abordado.

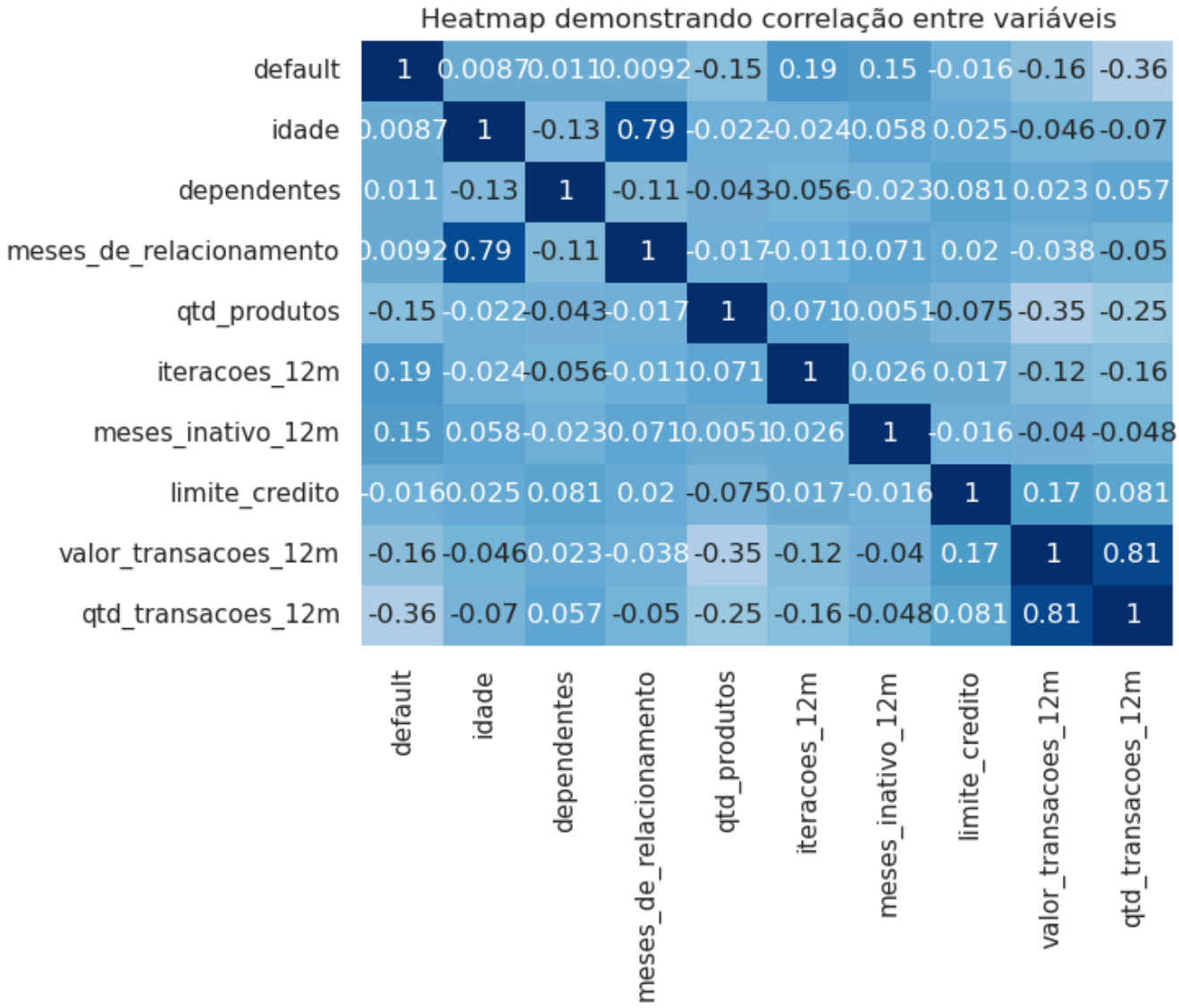
In [4]:

```
# Gerando um heatmap para analisar a correlação entre as variáveis numéricas:

sns.heatmap(df.corr(), cmap = 'Blues', center = 0, annot=True, cbar = False).set_title('Heatmap demonstrando correlação entre variáveis')
```

```
/tmp/ipykernel_20/3329051555.py:3: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  sns.heatmap(df.corr(), cmap = 'Blues', center = 0, annot=True, cbar = False).set_title('Heatmap demonstrando correlação entre variáveis')
```

Out[4]:

```
Text(0.5, 1.0, 'Heatmap demonstrando correlação entre variáveis')
```



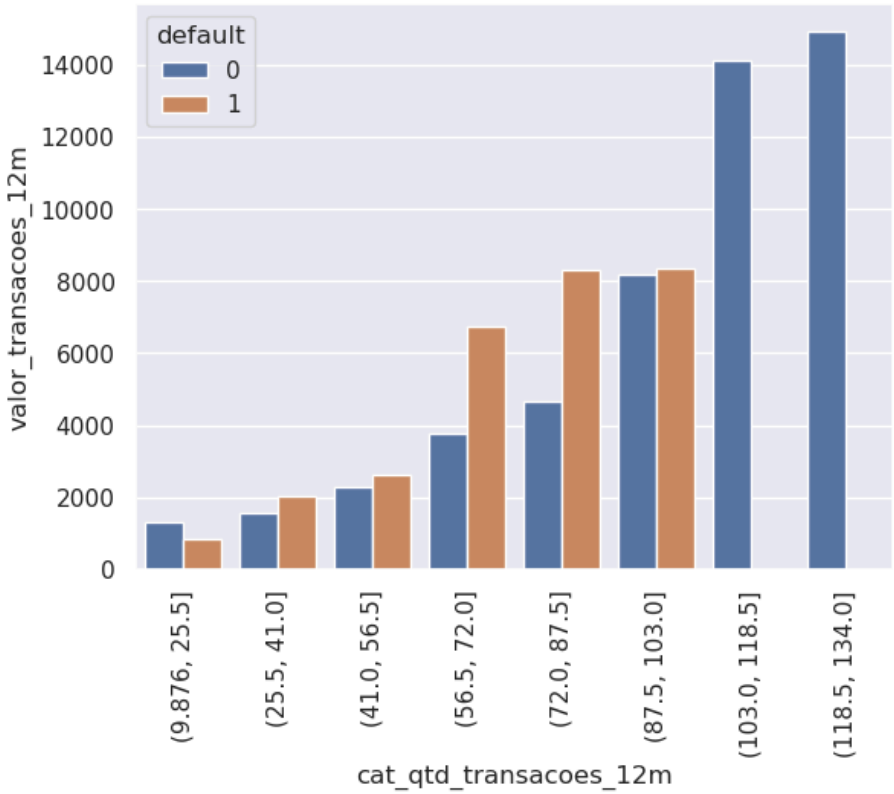
In [5]:

```
# Comparação entre quantidade de transações e valores de transações de adimplentes e
inadimplentes:



df['cat_qtd_transacoes_12m'] = pd.cut(df['qtd_transacoes_12m'], bins=8)
ax = sns.barplot(x = 'cat_qtd_transacoes_12m', y = 'valor_transacoes_12m', hue = 'default',
errorbar = None, data = df)
ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
ax.set_title('Comparação entre quantidade de transações e valores de transações de adimplentes
e inadimplentes:')
```

Out[5]:

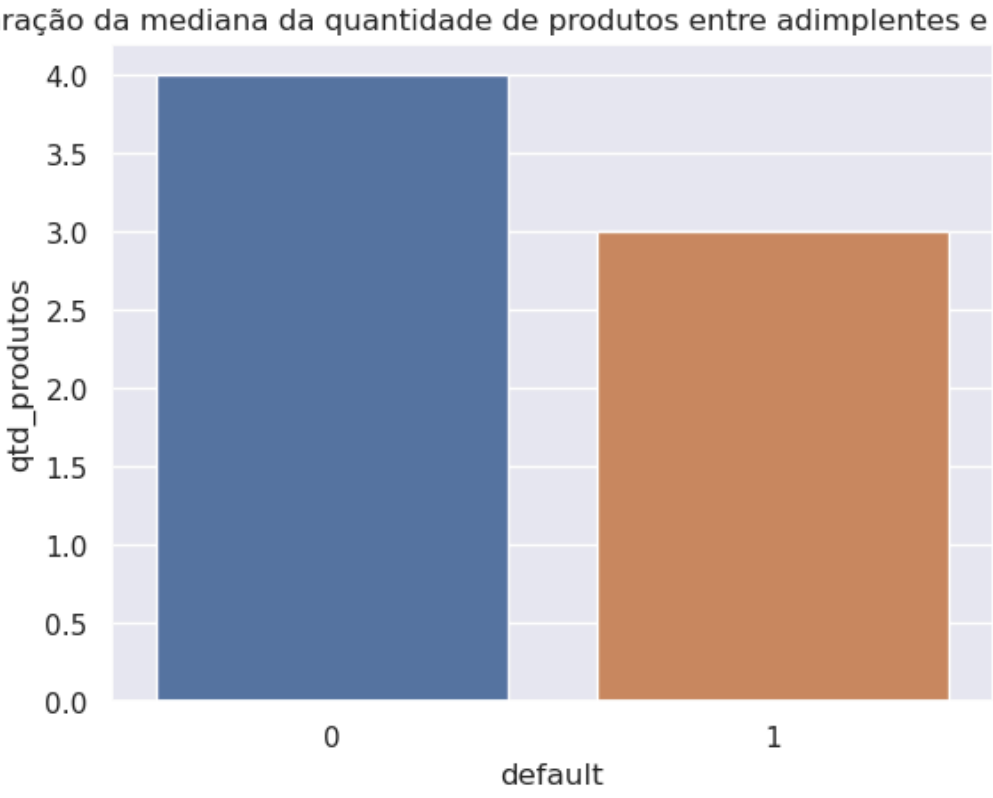
```
Text(0.5, 1.0, 'Comparação entre quantidade de transações e valores de transações de adimplen
tes e inadimplentes:')
```

Comparação entre quantidade de transações e valores de transações de adimplentes e inadimplentes:



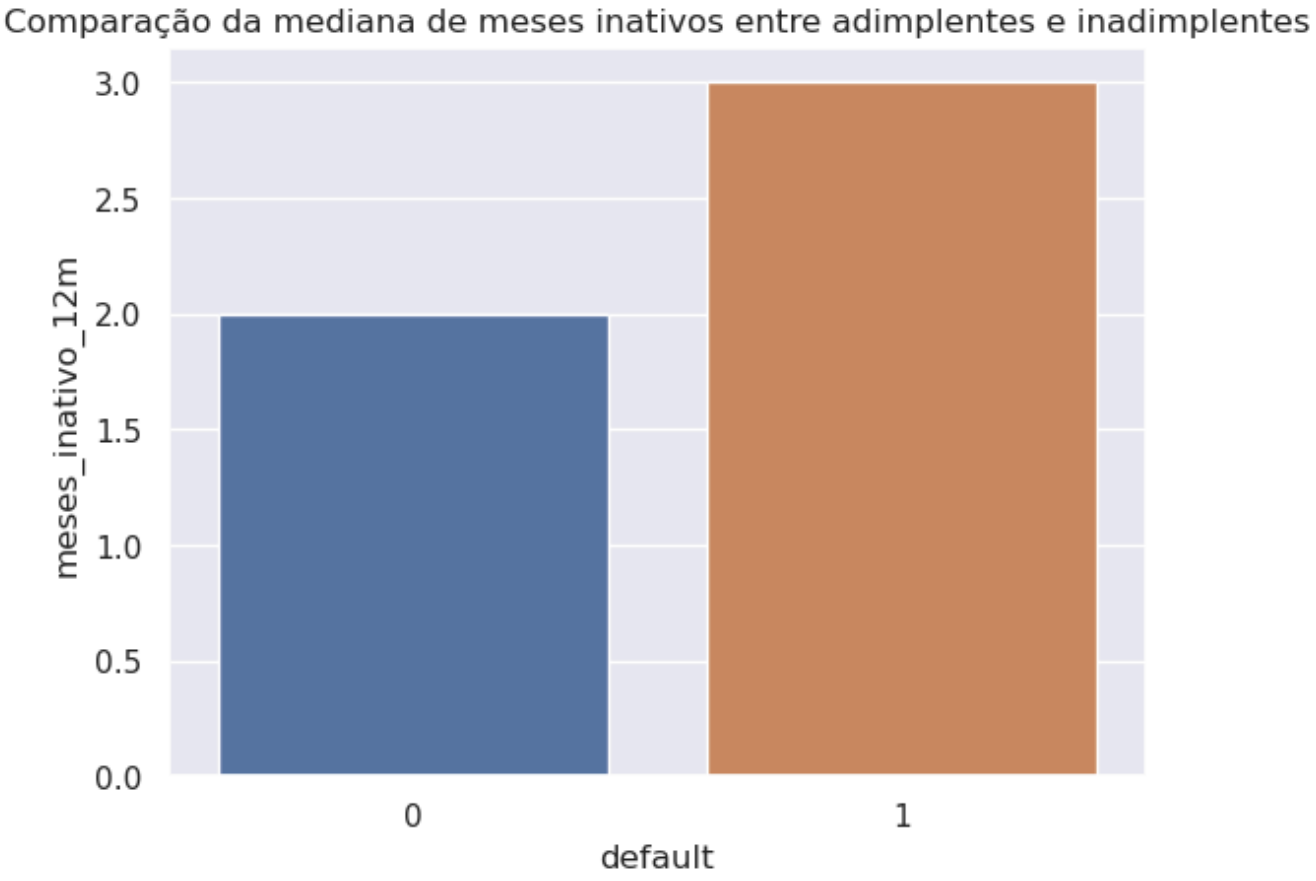
In [6]:
```
# # Comparação da mediana da quantidade de produtos entre adimplentes e inadimplentes:
sns.barplot(x = 'default', y = 'qtd_produtos', data = df, estimator =
'median').set_title('Comparação da mediana da quantidade de produtos entre adimplentes e
inadimplentes')
```

Out[6]:
```
Text(0.5, 1.0, 'Comparação da mediana da quantidade de produtos entre adimplentes e inadimple
ntes')
```



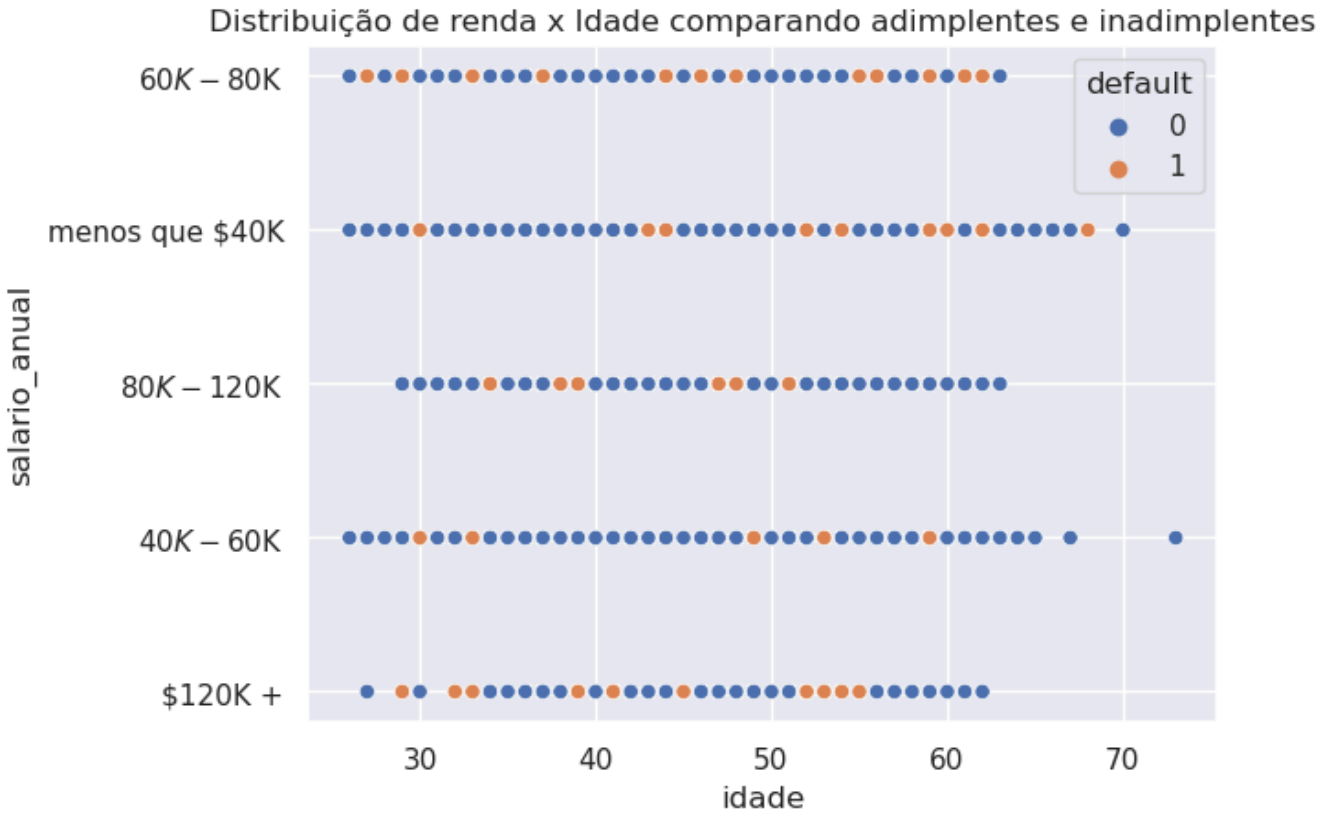
In [7]:
```
# Comparação da mediana de meses inativos entre adimplentes e inadimplentes:
sns.barplot(x = 'default', y = 'meses_inativo_12m', data = df, estimator =
'median').set_title('Comparação da mediana de meses inativos entre adimplentes e
inadimplentes')
```

Out[7]:
```
Text(0.5, 1.0, 'Comparação da mediana de meses inativos entre adimplentes e inadimplentes')
```



In [8]:
```
sns.scatterplot(data=df, x='idade', y='salario_anual', hue='default').set_title('Distribuição de renda x Idade comparando adimplentes e inadimplentes')
```

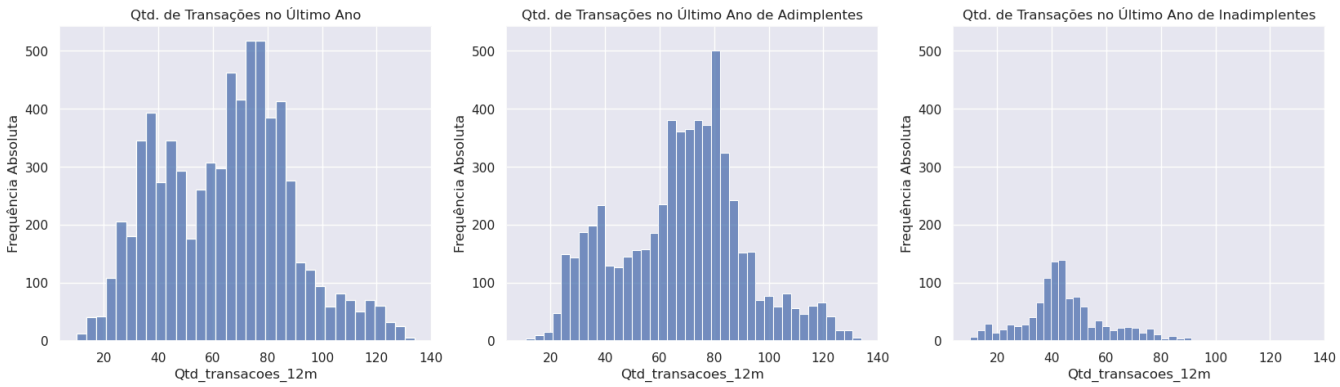
Out[8]:
```
Text(0.5, 1.0, 'Distribuição de renda x Idade comparando adimplentes e inadimplentes')
```



In [9]:
```
df_adimplente = df[df['default'] == 0]
df_inadimplente = df[df['default'] == 1]
```

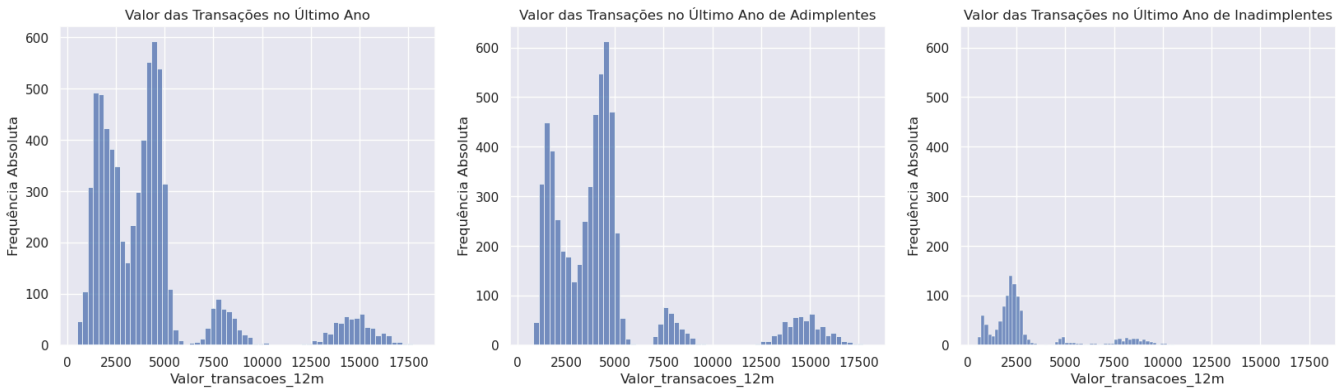
In [10]:

```
coluna = 'qtd_transacoes_12m'
titulos = ['Qtd. de Transações no Último Ano', 'Qtd. de Transações no Último Ano de
Adimplentes', 'Qtd. de Transações no Último Ano de Inadimplentes']

eixo = 0
max_y = 0
figura, eixos = plt.subplots(1,3, figsize=(20, 5), sharex=True)

for dataframe in [df, df_adimplente, df_inadimplente]:

  f = sns.histplot(x=coluna, data=dataframe, stat='count', ax=eixos[eixo])
  f.set(title=titulos[eixo], xlabel=coluna.capitalize(), ylabel='Frequência Absoluta')

  _, max_y_f = f.get_ylim()
  max_y = max_y_f if max_y_f > max_y else max_y
  f.set(ylim=(0, max_y))

  eixo += 1

figura.show()
```



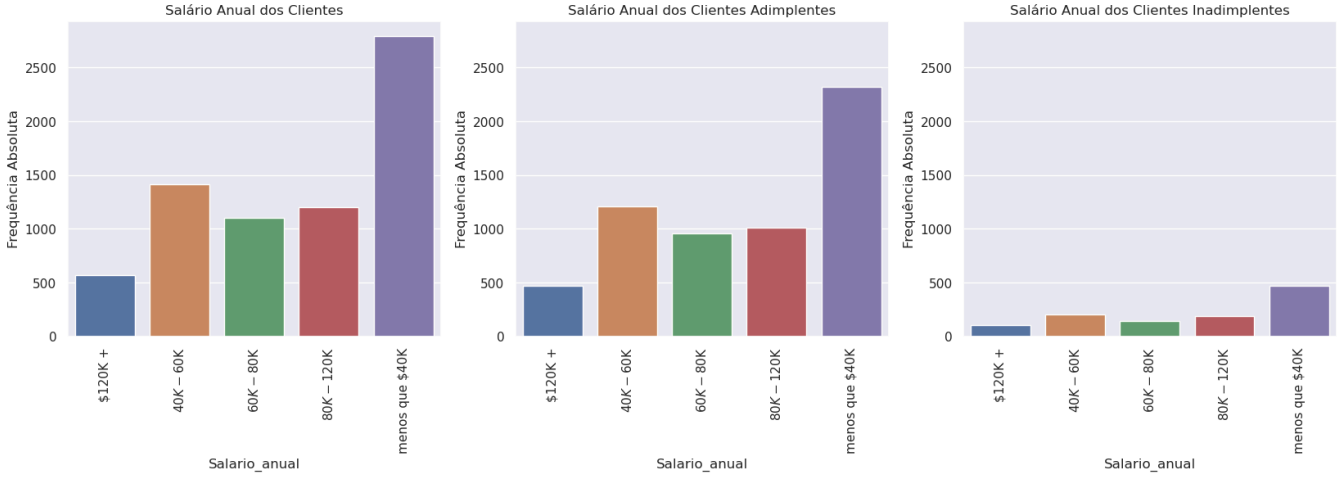
In [11]:

```
coluna = 'valor_transacoes_12m'
titulos = ['Valor das Transações no Último Ano', 'Valor das Transações no Último Ano de
Adimplentes', 'Valor das Transações no Último Ano de Inadimplentes']

eixo = 0
max_y = 0
figura, eixos = plt.subplots(1,3, figsize=(20, 5), sharex=True)

for dataframe in [df, df_adimplente, df_inadimplente]:

  f = sns.histplot(x=coluna, data=dataframe, stat='count', ax=eixos[eixo])
  f.set(title=titulos[eixo], xlabel=coluna.capitalize(), ylabel='Frequência Absoluta')

  _, max_y_f = f.get_ylim()
  max_y = max_y_f if max_y_f > max_y else max_y
  f.set(ylim=(0, max_y))

  eixo += 1

figura.show()
```



In [12]:

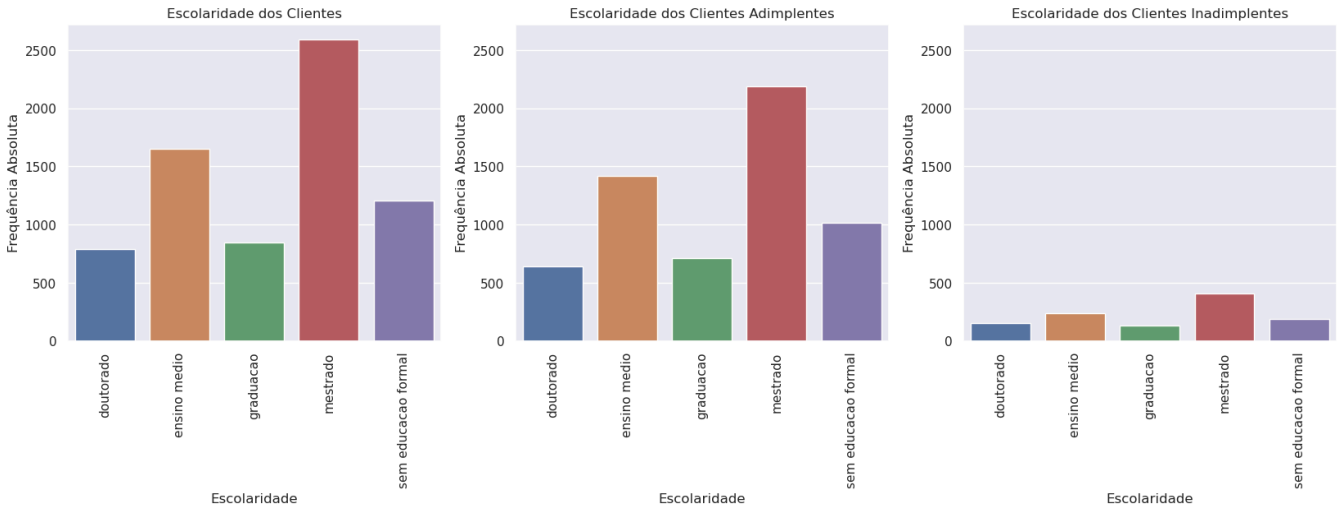
```
coluna = 'salario_anual'
titulos = ['Salário Anual dos Clientes', 'Salário Anual dos Clientes Adimplentes', 'Salário Anual dos Clientes Inadimplentes']

eixo = 0
max_y = 0
figura, eixos = plt.subplots(1,3, figsize=(20, 5), sharex=True)

for dataframe in [df, df_adimplente, df_inadimplente]:

  df_to_plot = dataframe[coluna].value_counts().to_frame()
  df_to_plot.rename(columns={coluna: 'frequencia_absoluta'}, inplace=True)
  df_to_plot[coluna] = df_to_plot.index
  df_to_plot.reset_index(inplace=True, drop=True)
  df_to_plot.sort_values(by=[coluna], inplace=True)

  f = sns.barplot(x=df_to_plot[coluna], y=df_to_plot['frequencia_absoluta'], ax=eixos[eixo])
  f.set(title=titulos[eixo], xlabel=coluna.capitalize(), ylabel='Frequência Absoluta')
  f.set_xticklabels(labels=f.get_xticklabels(), rotation=90)
  _, max_y_f = f.get_ylim()
  max_y = max_y_f if max_y_f > max_y else max_y
  f.set(ylim=(0, max_y))
  eixo += 1

figura.show()
```



In [13]:

```
coluna = 'escolaridade'
titulos = ['Escolaridade dos Clientes', 'Escolaridade dos Clientes Adimplentes', 'Escolaridade dos Clientes Inadimplentes']
```

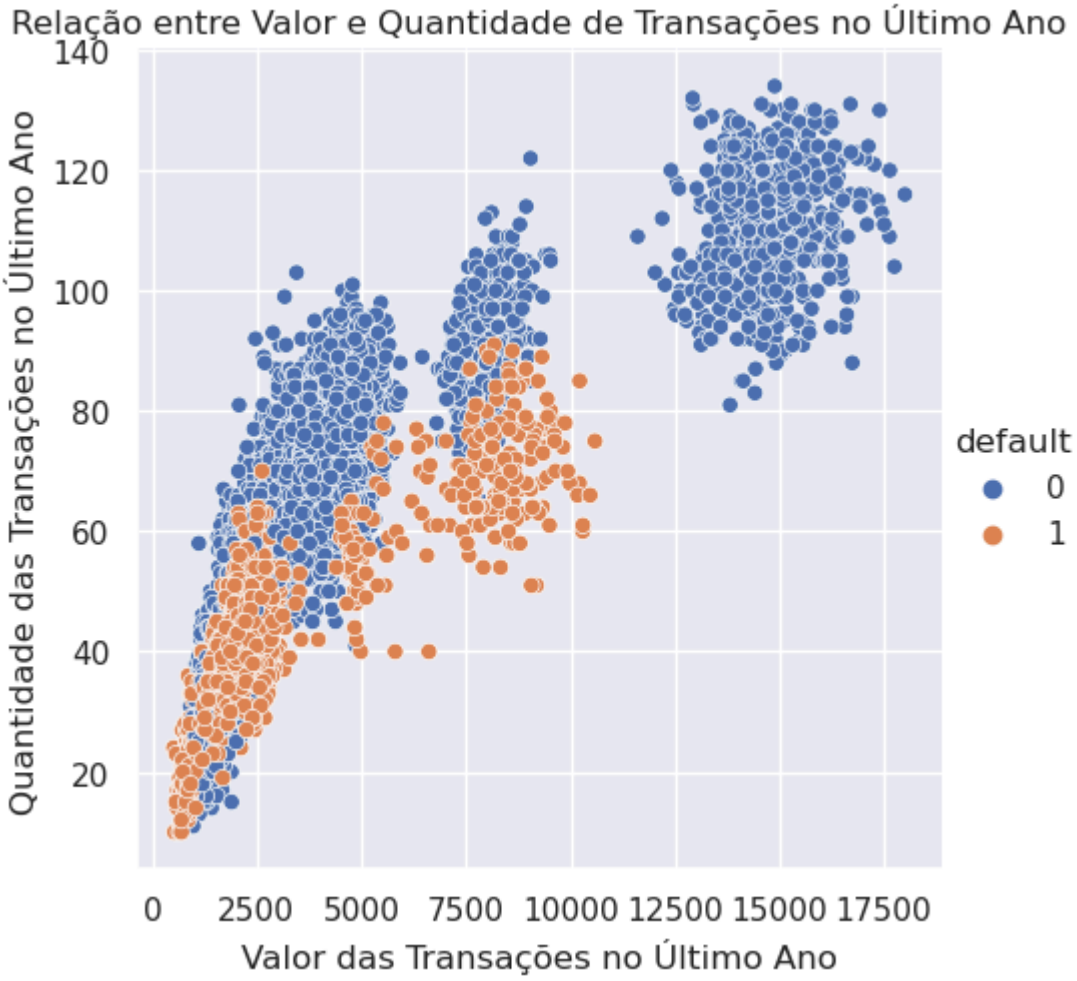
```
eixo = 0
max_y = 0
max = df.select_dtypes('object').describe()[coluna]['freq'] * 1.1

figura, eixos = plt.subplots(1,3, figsize=(20, 5), sharex=True)

for dataframe in [df, df_adimplente, df_inadimplente]:

  df_to_plot = dataframe[coluna].value_counts().to_frame()
  df_to_plot.rename(columns={coluna: 'frequencia_absoluta'}, inplace=True)
  df_to_plot[coluna] = df_to_plot.index
  df_to_plot.sort_values(by=[coluna], inplace=True)
  df_to_plot.sort_values(by=[coluna])

  f = sns.barplot(x=df_to_plot[coluna], y=df_to_plot['frequencia_absoluta'], ax=eixos[eixo])
  f.set(title=titulos[eixo], xlabel=coluna.capitalize(), ylabel='Frequência Absoluta')
  f.set_xticklabels(labels=f.get_xticklabels(), rotation=90)

  _, max_y_f = f.get_ylim()
  max_y = max_y_f if max_y_f > max_y else max_y
  f.set(ylim=(0, max_y))

  eixo += 1

figura.show()
```



In [14]:

```
f = sns.relplot(x='valor_transacoes_12m', y='qtd_transacoes_12m', data=df, hue='default')
_ = f.set(
    title='Relação entre Valor e Quantidade de Transações no Último Ano',
    xlabel='Valor das Transações no Último Ano',
    ylabel='Quantidade das Transações no Último Ano'
  )
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/axisgrid.py:118: UserWarning: The figure layo
ut has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



5. Interpretação de Resultados:

Através da análise dos dados podemos identificar alguns padrões e tendências:

- Há uma diferença notável entre o comportamento de adimplentes e inadimplentes quanto a quantidade e valor de transações realizados por ano.
- Existe também uma correlação entre a quantidade de produtos na cesta de produtos da instituição e inadimplência, onde há um aumento na chance de inadimplência quando o cliente possui 3 ou menos produtos em sua cesta.
- Ocorre ainda uma relação entre a quantidade de meses inativos por ano, onde inadimplentes passam em sua maioria cerca de 3 meses inativos em 12 meses, em comparação a 2 meses inativos de adimplentes.

6. Tomada de Decisão e Ações:

Os padrões identificados podem ser explorados para prever os clientes que podem se tornar inadimplentes ao longo do tempo e analisar mais atenciosamente a concessão de crédito para portadores das devidas características.