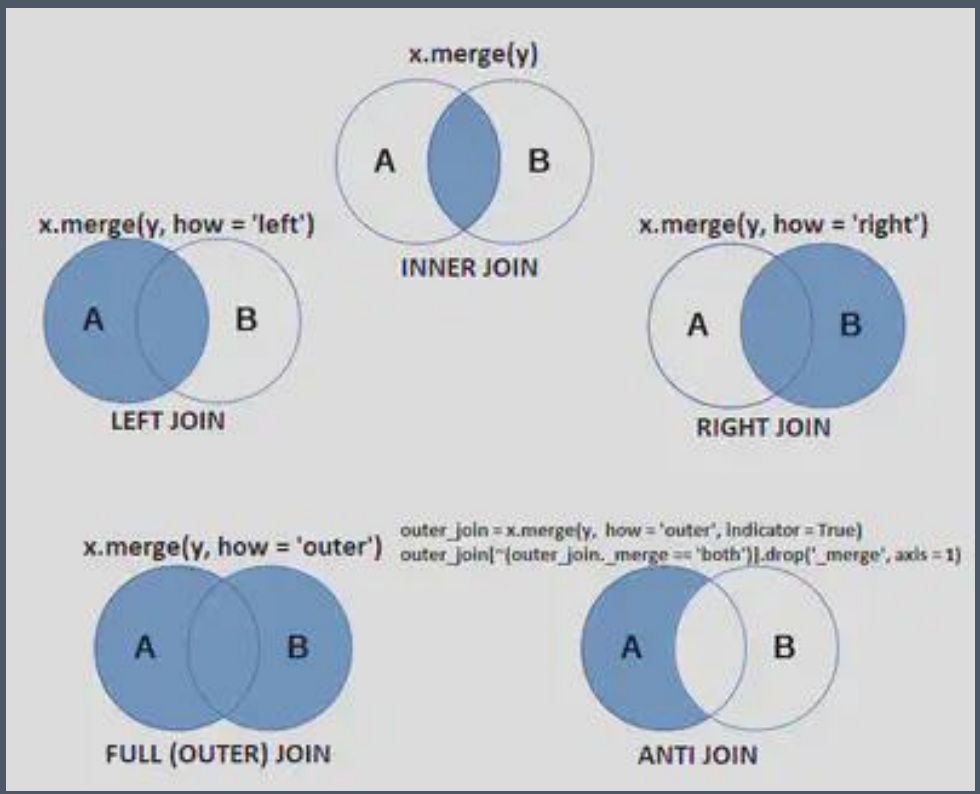
DATA Viking

Código rápido
Combinando tabelas no Pandas



Odemir Depieri Jr
Ronisson Lucas C. da Conceição
Yan Vianna Sym
Wesley Felix

Gerar dados

Iremos **criar dados aleatórios** dentro uma classe.
Nessa classe temos função para gerar **tabela A** e **tabela B**.

```
import pandas as pd
import numpy as np
import random
import datetime

# Gerador de dados aleatórios
class GeradorDados:

    def __init__(self, quantidade):
        self.quantidade = quantidade

    def gerar_dados_a(self):

        def gerar_data():
            '''
              Geradpr de datas aleatórias considerando meses bissexto e demais dias
            '''
            ano = random.randint(2020, datetime.date.today().year-1)
            mes = random.randint(1, 12)

            # fevereiro
            if mes == 2:

                # ano bissexto
                if (ano % 4 == 0 and ano % 100 != 0) or ano % 400 == 0:
                    dia = random.randint(1, 29)
                else:
                    dia = random.randint(1, 28)

            # meses com 30 dias
            elif mes in [4, 6, 9, 11]:
                dia = random.randint(1, 30)

            # meses com 31 dias
            else:
                dia = random.randint(1, 31)

            return datetime.date(ano, mes, dia)

        dicionario = {
            #'Id_Venda' : [ loop for loop in range( self.quantidade) ],
            'Id_Venda' : random.sample( range(0, (self.quantidade*2) ), self.quantidade) ,
            'Valor_Imovel': [ random.randint(150000, 350000) for loop in range(self.quantidade) ]
,
            'Comissao_Venda': [ random.randint(1, 6) / 100 for loop in range(self.quantidade) ],
            'Data_Venda': [ gerar_data() for loop in range(self.quantidade) ],
            'Tipo_Produto': [ np.random.choice(['Casa Residencial', 'Apartamento'], p=[0.25, 0.75]
) for loop in range(self.quantidade) ],
        }

        Base = pd.DataFrame( dicionario )
        return Base

    def gerar_dados_b(self):

        dicionario = {
            'Id_Venda' : random.sample( range(0, (self.quantidade*2) ), self.quantidade) ,
            'Vendedor': [ np.random.choice(['Odemir Depieri Jr', 'Ronisson Lucas Calmon']) for lo
op in range(self.quantidade) ]
        }

        Base = pd.DataFrame( dicionario )
        return Base
```

Dados tabela A

Vamos verificar como ficou nossa **tabela A**

```
# Gerando tabela A
Base_Dados_LadoA = GeradorDados(100).gerar_dados_a()
print( Base_Dados_LadoA.shape, '\n' )
Base_Dados_LadoA.head()
```

```
(100, 5)
```

	Id_Venda	Valor_Imovel	Comissao_Venda	Data_Venda	Tipo_Produto
0	65	286809	0.05	2021-09-19	Apartamento
1	21	334085	0.03	2020-05-11	Apartamento
2	146	270463	0.06	2020-04-12	Apartamento
3	20	180182	0.04	2020-04-09	Apartamento
4	44	227426	0.06	2021-11-12	Apartamento

Dados tabela B

Vamos verificar como ficou nossa **tabela B**

```
# Gerando tabela B
Base_Dados_LadoB = GeradorDados(100).gerar_dados_b()
print( Base_Dados_LadoB.shape, '\n' )
Base_Dados_LadoB.head()
```

```
(100, 2)
```

	Id_Venda	Vendedor
0	126	Odemir Depieri Jr
1	75	Odemir Depieri Jr
2	148	Odemir Depieri Jr
3	122	Ronisson Lucas Calmon
4	198	Ronisson Lucas Calmon

Esse dados foram gerador para simular uma situação de vendas.
Na **tabela A** temos os dados das vendas (Valor, Comissão ...)
Na **tabela B** temos os dados do responsável da venda

Argumento INNER

Quando usado na função **pd.merge()** do Pandas, o argumento **how="inner"** especifica que a operação de junção deve manter apenas as linhas onde as chaves (ou colunas) da tabela à esquerda e à direita **correspondem exatamente**.

```
# Combina as tabelas
Base_junta = pd.merge(Base_Dados_LadoA, Base_Dados_
LadoB, on='Id_Venda', how='inner')
print( Base_junta.shape, '\n' )
Base_junta.head()
```

```
(51, 6)
```

	Id_Venda	Valor_Imovel	Comissao_Venda	Data_Venda	Tipo_Produto	Vendedor
0	119	216063	0.03	2022-02-05	Casa Residencial	Odemir Depieri Jr
1	41	325829	0.05	2020-03-27	Apartamento	Odemir Depieri Jr
2	66	188537	0.04	2022-03-06	Apartamento	Ronisson Lucas Calmon
3	81	155500	0.01	2020-11-14	Apartamento	Ronisson Lucas Calmon
4	150	234035	0.01	2021-11-06	Apartamento	Ronisson Lucas Calmon

Argumento RIGTH

Quando usado na função **pd.merge()** do Pandas, o argumento **how="right"** especifica que a operação de junção deve manter todas as linhas da tabela da direita e apenas as linhas **correspondentes da tabela da esquerda**.

```
# Combina direita
Base_direita = pd.merge(Base_Dados_LadoA, Base_Dados
_LadoB, on='Id_Venda', how='right')
print( Base_direita.shape, '\n' )
Base_direita.head()
```

```
(100, 6)
```

	Id_Venda	Valor_Imovel	Comissao_Venda	Data_Venda	Tipo_Produto	Vendedor
0	119	216063.0	0.03	2022-02-05	Casa Residencial	Odemir Depieri Jr
1	39	NaN	NaN	NaN	NaN	Ronisson Lucas Calmon
2	15	209791.0	0.03	2020-02-11	Casa Residencial	Odemir Depieri Jr
3	186	NaN	NaN	NaN	NaN	Ronisson Lucas Calmon
4	192	NaN	NaN	NaN	NaN	Odemir Depieri Jr

Argumento LEFT

Quando usado na função **pd.merge()** do Pandas, o argumento **how="left"** especifica que a operação de junção deve manter todas as linhas da tabela da direita e apenas as linhas **correspondentes da tabela da direita**.

```
# Combina esquerda
Base_esquerda = pd.merge(Base_Dados_LadoA, Base_Dad
os_LadoB, on='Id_Venda', how='left')
print( Base_esquerda.shape, '\n' )
Base_esquerda.head()
```

```
(100, 6)
```

	Id_Venda	Valor_Imovel	Comissao_Venda	Data_Venda	Tipo_Produto	Vendedor
0	119	216063	0.03	2022-02-05	Casa Residencial	Odemir Depieri Jr
1	128	340566	0.06	2020-05-12	Apartamento	NaN
2	43	344336	0.05	2021-07-05	Casa Residencial	NaN
3	132	208337	0.01	2021-06-07	Apartamento	NaN
4	58	222442	0.05	2021-12-06	Apartamento	NaN

Argumento OUTER

Quando usado na função **pd.merge()** do Pandas, o argumento **how="outer"** especifica que a operação de junção deve manter todas as linhas de ambas as tabelas, combinando os dados **onde houver correspondência** e preenchendo os valores ausentes com **valores nulos onde não houver correspondência**.

```
# Combina as ambas
Base_full = pd.merge(Base_Dados_LadoA, Base_Dados_La
doB, on='Id_Venda', how='outer')
print( Base_full.shape, '\n' )
Base_full.head()
```

```
(149, 6)
```

	Id_Venda	Valor_Imovel	Comissao_Venda	Data_Venda	Tipo_Produto	Vendedor
0	119	216063.0	0.03	2022-02-05	Casa Residencial	Odemir Depieri Jr
1	128	340566.0	0.06	2020-05-12	Apartamento	NaN
2	43	344336.0	0.05	2021-07-05	Casa Residencial	NaN
3	132	208337.0	0.01	2021-06-07	Apartamento	NaN
4	58	222442.0	0.05	2021-12-06	Apartamento	NaN