1.

Para relacionar os dois dataframes `DSOS` e `DSTEC` em um relacionamento 1-N (um-para-muitos) utilizando Python, você pode usar a função `merge` do pandas. O relacionamento 1-N significa que cada linha no dataframe `DSOS` pode estar associada a várias linhas no dataframe `DSTEC`, com base em uma chave comum, que no caso é o campo `numos`.

Aqui está um exemplo de como fazer isso:

```python

import pandas as pd

# Exemplo de dataframes

DSOS = pd.DataFrame({

'numos': [1, 2, 3],

'descricao': ['OS1', 'OS2', 'OS3']

})

DSTEC = pd.DataFrame({

'nome': ['Tec1', 'Tec2', 'Tec3', 'Tec4'],

'numos': [1, 1, 2, 3]

})

# Realizando o merge (relacionamento 1-N)

resultado = pd.merge(DSOS, DSTEC, on='numos', how='inner')

# Exibindo o resultado

print(resultado)

```

### Explicação:

- \*\*`DSOS`\*\*: Contém as ordens de serviço com os campos `numos` e `descricao`.

- \*\*`DSTEC`\*\*: Contém os técnicos associados a cada ordem de serviço, com os campos `nome` e `numos`.

- \*\*`pd.merge`\*\*: Realiza o relacionamento entre os dois dataframes com base na coluna `numos`.

- \*\*`on='numos'`\*\*: Especifica a coluna comum para o relacionamento.

- \*\*`how='inner'`\*\*: Define o tipo de join. Neste caso, `inner` retorna apenas as linhas que têm correspondência em ambos os dataframes. Você pode usar outros tipos de join, como `left`, `right`, ou `outer`, dependendo da sua necessidade.

### Resultado:

O dataframe `resultado` conterá todas as combinações possíveis onde o `numos` de `DSOS` corresponde ao `numos` de `DSTEC`. Por exemplo:

```

numos descricao nome

0 1 OS1 Tec1

1 1 OS1 Tec2

2 2 OS2 Tec3

3 3 OS3 Tec4

```

Neste exemplo, a ordem de serviço `1` está associada a dois técnicos (`Tec1` e `Tec2`), enquanto as ordens `2` e `3` estão associadas a um técnico cada.

### Observação:

- Se você quiser manter todas as ordens de serviço, mesmo aquelas que não têm técnicos associados, você pode usar `how='left'` no `merge`.

- Se você quiser incluir todos os técnicos, mesmo aqueles que não estão associados a uma ordem de serviço, use `how='right'`.

- Para incluir todos os registros de ambos os dataframes, use `how='outer'`.

2.

Se você deseja listar apenas os dados do dataframe `DSOS`, mas agrupando-os de acordo com as informações relacionadas no dataframe `DSTEC`, você pode usar o método `groupby` do pandas. Isso permite agrupar os dados com base em uma coluna específica (no caso, `numos`) e, em seguida, aplicar uma função de agregação, como `count`, `sum`, `mean`, etc.

Aqui está um exemplo de como fazer isso:

### Exemplo de código:

```python

import pandas as pd

# Exemplo de dataframes

DSOS = pd.DataFrame({

'numos': [1, 2, 3],

'descricao': ['OS1', 'OS2', 'OS3']

})

DSTEC = pd.DataFrame({

'nome': ['Tec1', 'Tec2', 'Tec3', 'Tec4'],

'numos': [1, 1, 2, 3]

})

# Realizando o merge (relacionamento 1-N)

resultado = pd.merge(DSOS, DSTEC, on='numos', how='left')

# Agrupando por 'numos' e 'descricao' e contando o número de técnicos associados

agrupado = resultado.groupby(['numos', 'descricao']).size().reset\_index(name='quantidade\_tecnicos')

# Exibindo o resultado

print(agrupado)

```

### Explicação:

1. \*\*`pd.merge`\*\*: Realiza o relacionamento entre `DSOS` e `DSTEC` usando `numos` como chave. O `how='left'` garante que todas as linhas de `DSOS` sejam mantidas, mesmo que não haja correspondência em `DSTEC`.

2. \*\*`groupby`\*\*: Agrupa os dados pelas colunas `numos` e `descricao`.

3. \*\*`size()`\*\*: Conta o número de ocorrências (técnicos associados) para cada grupo.

4. \*\*`reset\_index(name='quantidade\_tecnicos')`\*\*: Renomeia a coluna de contagem para `quantidade\_tecnicos` e transforma o resultado em um dataframe.

### Resultado:

O dataframe `agrupado` conterá as informações de `DSOS` com uma coluna adicional que indica a quantidade de técnicos associados a cada ordem de serviço:

```

numos descricao quantidade\_tecnicos

0 1 OS1 2

1 2 OS2 1

2 3 OS3 1

```

### Observações:

- Se uma ordem de serviço não tiver técnicos associados, a coluna `quantidade\_tecnicos` terá o valor `0`.

- Você pode adaptar o método de agregação (`size`, `count`, `sum`, etc.) dependendo da análise que deseja realizar. Por exemplo, se quiser listar os nomes dos técnicos em vez de contar, pode usar `agg` com uma função personalizada.

### Exemplo com lista de técnicos:

Se você quiser listar os nomes dos técnicos associados a cada ordem de serviço em vez de apenas contar, pode fazer o seguinte:

```python

# Agrupando e listando os nomes dos técnicos

agrupado\_com\_nomes = resultado.groupby(['numos', 'descricao'])['nome'].apply(lambda x: ', '.join(x.dropna())).reset\_index()

# Exibindo o resultado

print(agrupado\_com\_nomes)

```

Isso resultará em:

```

numos descricao nome

0 1 OS1 Tec1, Tec2

1 2 OS2 Tec3

2 3 OS3 Tec4

```

Neste caso, os nomes dos técnicos são concatenados em uma única string para cada ordem de serviço.