

Explicação do passo a passo utilizado no código

Olá, esta é a explicação passo a passo do código desenvolvido para o desafio técnico da Solvimm, utilizando as tecnologias: Python (pandas) e Jupyter Notebook.

Bloco 1: Importando as bibliotecas Pandas e Numpy;

Bloco 2: Utilizando o método `.read_csv` para importar as duas base de dados;

Bloco 3: Para garantir que não tem valores nulos nas bases, chamei os métodos `.isnull().values.any()` que retornou `False`.

Bloco 4: Nesse bloco eu utilizei a coluna `'Nome_Ano'` da base `movies.csv`, chamando o método `.str.split(', ', expand=True)` para transformar os valores em strings e dividi las em torno do separador definido que foi “vírgula + espaço”, após isso criei duas colunas novas com `df[['Nome', 'Ano']]` e atribui a primeira parte da string à primeira coluna (`'Nome'`) e a segunda parte à segunda coluna (`'Ano'`);

Bloco 5: Aqui utilizei o `.drop('Nome_Ano')` passando a coluna `'Nome_Ano'` para eliminá-la da base já que não é mais útil, depois usei o `.strip('(')` e `.strip('')` nas colunas `'Nome'` e `'Ano'` para eliminar os parênteses que ficaram sobrando nos nomes e anos dos filmes.

Bloco 6 (Resposta da pergunta 1.1): Como já garanti que a base `movies.csv` não possui linha em branco, cada linha da tabela representa um filme, logo basta usar o `.shape[0]` para contar e printar a quantidade de linhas que a base de dados possui;

Bloco 7 (Resposta da pergunta 1.2): Utilizando a segunda base e chamando o método `.groupby('Movie_Id').mean()` e passando o nome da coluna `'Movie_Id'` para agrupar os ids iguais e fazer a média dos valores, atribuindo-os aos respectivos ids. Chamar `.sort_values('Rating', ascending=False)` passando a coluna `'Rating'` como parâmetro para organizar a tabela com base nos melhores rating e chamando também o `.head(5)` passando o parâmetro “5” para retornar as 5 primeiras linhas com base no rating. Chamando o `.reset_index()` para manter as colunas `'Movie_Id'` e `'Rating'`, depois utilizar o `.to_list()` na coluna “`Movie_Id`” para criar uma lista com os ids dos filmes com melhor média, em seguida usar o `.query('ID == @best_ids')` para comparar os ids na coluna “ID” com os ids presentes na lista, e por fim usar `['Nome'].to_list()` para retornar uma lista com os nomes dos 5 filmes com melhor média de avaliação;

Bloco 8 (Resposta da pergunta 1.3): Nessa pergunta tive que usar `.astype(int)` para transformar os valores da coluna em int, em seguida usei `df.groupby(['Ano'])` para ordenar a tabela com base na coluna ano e `.size()` contar quantos filmes foram lançados em cada ano, e por fim `.nsmallest(9)` para selecionar os 9 anos com os menores valores na coluna lançamento. Na linha seguinte coloquei `.reset_index()` para manter as colunas definidas e `.rename(columns={0: 'Lançamentos'}, inplace = True)` chamei o rename

para renomear a coluna de nome '0' para 'Lançamentos' e por fim printei a tabela com os 9 anos com menos lançamentos de filmes;

Bloco 9 (Resposta da pergunta 1.4): Comecei usando `df2['Date'].max()` para achar qual era a última data, depois fiz essa comparação `df2['Date'] == last_date` para pegar somente as avaliações que foram feitas na última data, e usei o `.groupby('Movie_Id').mean()` para agrupar os ids iguais da coluna e fazer a média dos valores das outras colunas e `.sort_values('Rating', ascending=False)['Rating']` Para ordenar a tabela com base nos 'Rating', para separar as colunas usei `.reset_index()`. E para pegar apenas as médias maiores ou iguais a 4.7 fiz `movies = order_rating[order_rating['Rating'] >= 4.7]`, por fim é só printar `movies.shape[0]` para saber a quantidade de linhas que é igual a quantidade de filmes que atendem as restrições;

Bloco 10 (Resposta da pergunta 1.5): Nessa questão usei a tabela `movies` obtida na questão anterior, que já possui as colunas 'Movie_Id' e 'Rating' organizadas por rating, então fiz `movies.iloc[-10:]` para pegar os 10 últimos da tabela, que são as 10 piores médias, e fiz a variável `last = list['Movie_Id'].to_list()` que recebe os ids em forma de lista, para comparar os ids da lista com os ids da coluna 'ID' da primeira base de filmes, fiz `df.query('ID == @last')['Nome'].reset_index()` mostrando o nome do filme e mantendo separando as colunas, depois criei a variável `nova = list['Rating'].to_list()` que recebe os valores da coluna 'Rating' como uma lista, `data.insert(2, "rating", nova)` e então peguei a coluna com os nomes do filmes e inseri a coluna com as notas, por fim exclui a coluna que continha os ids usando `data.drop(['index'], axis=1)` e printei a tabela, exibindo os nomes e notas dos 10 filmes com as piores notas;

Bloco 11 (Resposta da pergunta 1.6): Comecei com `df2.groupby(['Cust_Id']).size().nlargest(5)` para organizar a tabela pela coluna 'Cust_Id', usei o `.size()` para contar quantas avaliações cada avaliador fez e `.nlargest(5)` para pegar os 5 maiores números de avaliações, então usei `.reset_index()` para manter os valores separados em colunas e `.rename(columns={0: 'Avaliações'}, inplace = True)` para renomear a coluna '0' para 'Avaliações' e para encerrar é só printar a tabela para mostrar os 5 avaliadores que mais avaliaram e quantas avaliações cada um fez.

Explicação de como executar o código do zero

Para executar o código basta garantir que está importando corretamente as duas bases de dados, e importar as bibliotecas Pandas e Numpy, e então ir executando os blocos de código em sequência.