2.1 – Explicação do passo a passo utilizado no código.

1.1. Quantos filmes estão disponíveis no dataset?

CÓDIGO: *import pandas as pd*
COMENTÁRIO: Importando a biblioteca pandas com o comando *import* e dando um apelido para ela, facilitando o uso nas próximas linhas do código, com o comando *as*.

CÓDIGO: *movies = pd.read_csv("movies.csv", sep=";", names=('ID', "Filmes/Ano"))*
customer = pd.read_csv ('customers_rating.csv', sep=";")
COMENTÁRIO: Lendo os arquivos base (comando pd.read_csv), separando os dados pelo caractere ; "ponto e vírgula" com o comando *sep* para o novo dataframe que se chama customer e atribuindo nomes as colunas (com o comando *names*) do novo dataframe que agora se chama *movies*, atribuído pelo sinal de = em ambos dataframes.

CÓDIGO: *movies.shape[0]*
COMENTÁRIO: O comando *shape* retorna o tamanho em linhas e colunas do dataframe, neste caso, o *shape[0]* está especificando o retorno apenas do número de linhas do dataframe chamado *movies*. **Trazendo assim, a resposta para a primeira pergunta.**

1.2. Qual é o nome dos 5 filmes com melhor média de avaliação?

CÓDIGO: *customer.head()*
COMENTÁRIO: Com as informações separadas e apenas para título de visualização, usamos o *head()* que nesse caso retorna as 5 primeiras linhas do dataframe *customer*.

CÓDIGO: *top_five = customer.groupby("Movie_Id").mean()["Rating"].nlargest(5)*
top_five.head()
COMENTÁRIO: Criando um dataframe de nome *top_five*, agrupando as informações baseadas no Movie_Id pelo comando *groupby()*, e trazendo suas respectivas médias de rating, com o comando *mean()* retornando apenas os 5 maiores com o *nlargest(5)*. O *top_five.head()* mostra esse novo dataframe criado.

CÓDIGO: *movies.set_index('ID', inplace=True)*
COMENTÁRIO: Definindo o índice do dataframe movies e substituindo o dataframe anterior pelo atual. O comando set.index atribui um índice ao dataframe a partir de uma coluna já existente e o inplace, nesse caso é pra gravar as alterações nesse dataframe atual. Esse comando serve para facilitar a chamada do próximo comando

CÓDIGO: *movies.loc[[3456,3033,2102,4238,13]].head()*
COMENTÁRIO: Comando *loc* serve para consultar linhas e colunas específicas, nesse caso, juntamente com o *head()* retornando apenas os ID (que separamos no dataframe *top_five*), agora também com os nomes dos filmes (Filme/Ano). **Trazendo assim, a resposta para a segunda pergunta.**

1.3. Quais os 5 anos com menos lançamentos de filmes?

CÓDIGO: *movies = pd.read_csv("movies.csv", sep=",", names=('ID/Filmes', "Ano"))*
COMENTÁRIO: Lendo novamente a base de dados (comando *read_csv*), nomeando como *movies*, separando dessa vez por , “vírgula” com o comando *sep* e nomeado as colunas com o comando *names*.

CÓDIGO: *bad_movies = movies["Ano"].value_counts().nsmallest(5)*
bad_movies.head()
COMENTÁRIO: Criando um dataframe de nome bad_movies, organizando pelo Ano (comando *movies["Ano"]*) e contando quantas vezes esse valor Ano é “citado” (comando *value_count*), filtrando apenas os 5 menores, de menor valor, nesse caso, menor aparição com o comando *nsmallest(5)*. O bad_movies.head() retorna as 5 primeiras linhas do dataframe. **Trazendo assim, a resposta para a terceira pergunta.**

1.4. Quantos filmes que possuem avaliação maior ou igual a 4.7, considerando apenas os filmes avaliados na última data de avaliação do dataset?

CÓDIGO: *large_rating = customer.query('Rating >= 4.7')*
large_hating
COMENTÁRIO: Criando um dataframe chamado *large_rating* sendo filtrado APENAS os filmes com o rating maior ou igual a 4.7, com o comando *query* (que é similar ao *loc*, serve para consulta, explicado anteriormente) com o parâmetro específico *Rating>=4.7*. O comando *large_hating* em seguida, serve para retornar o resultado dessa consulta.

CÓDIGO: *best_movies = large_rating.sort_values(['Date'], ascending=[False])*
COMENTÁRIO: Criando um dataframe chamado best_movies retornando os valores organizados pela data da maior para a menor, com os comandos sort_values e ascending, respectivamente. O ascending=[False] é similar à “ordem decrescente”, ou seja, da maior para a menor. O comando best_movies em seguida, serve para retornar o resultado dessa consulta.

CÓDIGO: *best_movies.query('Date == "2005-12-31"').shape[0]*
COMENTÁRIO: Como já descobrimos a maior data de publicação, agora estamos consultado todos os filmes desse dataframe (com o comando query) e retornando sua contagem de linhas (comando shape[0]). **Chegamos então à resposta da quarta pergunta.**

1.5. Dos filmes encontrados na questão anterior, quais são os 10 filmes com as piores notas e quais as notas?

CÓDIGO: *small_rating = best_movies.sort_values(["Date"], ascending=[True])*
small_rating.head(10)
COMENTÁRIO: Criando um dataframe chamado *small_rating* retornando os valores organizados pela data menor (mais antiga) para a maior (mais atual), (já que todas as notas são iguais, atribui como nota “menor” a que foi atribuída primeiro, ou seja, a mais antiga) com os comandos *sort_values* e *ascending*, respectivamente. O *ascending=[True]* é similar à “ordem crescente”, ou seja, da data mais antiga para a mais nova. O comando *small_rating(10)* em seguida, serve para retornar os 10 primeiros itens como resultado dessa consulta.

CÓDIGO: *small_rating = small_rating.drop(columns=['Cust_Id'])*
COMENTÁRIO: Deletando a coluna *Cust_Id* do nosso dataframe, (com o comando *drop(columns=['Cust_Id'])*, pois não será necessária para essa questão e atribuindo à um novo dataframe de mesmo nome.

CÓDIGO: *last_ten = small_rating.head(10)*
COMENTÁRIO: Consultando apenas os 10 primeiros itens do dataframe (comando *head(10)*) e atribuindo o resultado a um novo dataframe chamado *last_ten*.

CÓDIGO: *last_ten*
COMENTÁRIO: Comando para retornar o dataframe criado no comando anterior, apenas para melhor visualização.

CÓDIGO: *last_ten = last_ten.reindex(columns=['Movie_Id', 'Rating', 'Date'])*
COMENTÁRIO: Alterando a posição(sequencia) das colunas do dataframe(para melhor visualização posterior) passando a nova sequencia das posições. Comando *reindex(columns=['Movie_Id', 'Rating', 'Date'])*.

CÓDIGO: *last_ten*
COMENTÁRIO: Comando para retornar o dataframe *last_ten*, agora com as colunas já na ordem informada no comando anterior.

CÓDIGO: *last_ten = last_ten.drop(columns=['Date'])*
COMENTÁRIO: Deletando a coluna Date com o comando *drop(columns=['Date'])*, pois não será necessário para o resultado da questão e estava até aqui para melhor visualização do dataframe e estamos salvando em um dataframe de mesmo nome (ou pode ser entendido como 'gravando a alteração feita no próprio dataframe').

CÓDIGO: *last_ten*
COMENTÁRIO: Comando para retornar o dataframe last_ten, agora sem a coluna Date que foi excluída no comando anterior, restando apenas as colunas Movie_Id e Rating, respectivamente. **Chegamos então à resposta para a quinta pergunta.**

1.6. Quais os id's dos 5 customer que mais avaliaram filmes e quantas avaliações cada um fez?

CÓDIGO: *best_customer = customer.groupby("Cust_Id").count()["Rating"].nlargest(5)*
COMENTÁRIO: Retornando o *Cust_Id*, agrupando cada um a sua respectiva quantidade de *Rating* atrelada a esse *Cust_Id* com os comandos *groupby* e *count*, respectivamente. Trazendo apenas os 5 maiores (em números de "aparições" ou "quantidade" de ratings) com o comando *nlargest(5)*, atribuindo o resultado a um novo dataframe chamado de *best_customer*.

CÓDIGO: *best_customer*
COMENTÁRIO: Retornando o dataframe criado no comando anterior. **Chegamos então à resposta da sexta questão.**