Init

import	import pandas as pd

Creating Reading Writing

read_csv()	pd.read_csv('example.csv', index_col=0, usecols=['title'], sep=';')
to_csv()	df.to_csv('example.csv', index=False, sep=';', header=False)
head()	df.head(5)
copy()	df_copy = df[['title', 'year']].copy()

index_col=0: La primera columna del df, será el índice.
usecols=['title']: Solo carga del df, la columna title.
sep=';':Tipo de separador que contiene el CSV.
header=False: No contiene una fila con nombres de la columna, se sustituyen por números.
index=False: Solo se guardan las columnas.
copy(): Cambiar en df_copy, no afecta a df. Sin copy, crea una referencia de df, cambios en df_copia puede afectar a df.

Indexing Selecting Assigning

loc()	df.loc[df['year'] >= 2018]
loc() with ==	df.loc[df['title'] == 'Black Panther', 'people_score'] = 100
loc() with &	df.loc[(df['critic_score'] > 90) & (df['people_score'] > 80)]
str.startswith()	df.loc[df['title'].str.startswith('S')]
str.contain()	df.loc[df['title'].str.contains('Spider'), 'critic_score'] = 95
iloc()	df.iloc[5:11, [0, 1, 3]]
subtract	df['critic_score'] - df['people_score']
median()	df.loc[df['critic_score'] > df['critic_score'].median()]

loc(): Selecciona filas y columnas basado en una etiqueta, en este caso los que cumplan x condición.
iloc(): Selecciona filas y columnas basado en su índice, en este caso las filas del 5 al 10 y las columnas 0, 1, 3.
str.startswith(): En este caso, selecicona las películas cuyo título comience por S.
str.contains(): En este caso, selecicona las películas cuyo título contengan 'Spider'.
median(): Hace la mediana de la columna 'critic_score'

Summary Functions Maps

mean()	df['critic_score'].mean()
describe()	df.describe()
value_counts()	df['type'].value_counts()
min()	df['people_score'].min()
max()	df['people_score'].max()
def	def score_ratio(row):
	newcolumn = row['critic_score'] / row['people_score']
	return newcolumn
apply()	df['score_ratio'] = df.apply(score_ratio, axis=1)
map()	df['type'] = df['type'].map(lambda x: x.lower())
std()	df['people_score'].std()
agg()	critic_score_agg = df['critic_score'].agg(['mean', 'std'])

mean(): Calcula la media.
describe(): Resumen estadístico del df.
value_counts(): Cuenta cuantas películas hay de cada tipo.
apply(): En la columna nueva (score_ratio) los valores se obtendrán por la función 'score_ratio'.
map(): Cambia todos los valores a minúscula con **lower()**.
std(): Mide la dispersión de los datos respecto a la media.
agg(): Calcula múltiples estadísticas a la vez (mean y std).

Grouping Sorting

sort_values() with loc()	df.sort_values('year').loc[:, ['title', 'year', 'critic_score']]
sort_values() with agg()	df.groupby('type').agg({'critic_score': ['mean', 'std']})
groupby() with sum()	df.groupby('production_co')['people_score'].sum()
groupby() with size()	df.groupby('year').size()
groupby() with mean() and sort_values() and sort_values()	df.groupby('production_co').mean(numeric_only=True).sort_values('critic_score', ascending=False)

df.groupby('production_co').mean(numeric_only=True).sort_values('critic_score', ascending=False): Agrupa por columna, calcula la media de columnas numéricas y ordena por la columna 'critic_score' de mayor a menos.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Data_type Missing_values	
astype()	df['year'].astype(str)
	df['critic_score'].astype(float)
isnull() with sum()	df.isnull().sum()
fillna()	df['people_score'].fillna(df['people_score'].median())
	df['people_score'].fillna(0)
dropna()	df.dropna(subset=['critic_score'])
str.replace()	df['release_date_(theaters)'].str.replace('wide', '').str.replace('limited', '')
to_datetime()	pd.to_datetime(df['release_date_(theaters)'])

astype(): convierte una columna a otro tipo de dato.
isnull(): detecta valores nulos en el DataFrame.
fillna(): reemplaza valores nulos con un valor especificado.
dropna(): elimina filas o columnas con valores nulos.
str.replace(): reemplaza patrones en texto dentro de una serie.
to_datetime(): convierte datos a formato de fecha y hora.

Renaming Combining	
rename()	movies.rename(columns={'title': 'movie_title', 'release_year': 'release_year'}, inplace=True)
concat()	pd.concat([movies_info, box_office], axis=1)
set_index()	movies.set_index('id')
join()	movies.join(reviews.set_index('id'))
df.index.name	movies.index.name = 'movies_index'
df.columns.name	movies.columns.name = 'movie_attributes'

rename(): cambia nombres de columnas o índices en el DataFrame.
concat(): une múltiples DataFrames a lo largo de un eje.
set_index(): establece una columna como índice del DataFrame.
join(): combina DataFrames usando el índice.
df.index.name: asigna o muestra el nombre del índice.
df.columns.name: asigna o muestra el nombre de las columnas.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com