---

50 Hard Pandas Questions & Answers (Q&A)

Basics & DataFrames

1. **Q:** How to create a Pandas Series?
   A: import pandas as pd; s = pd.Series([1,2,3]).
2. **Q:** How to create a DataFrame?
   A: df = pd.DataFrame({'A':[1,2],'B':[3,4]}).
3. **Q:** Difference between Series and DataFrame?
   A: Series → 1D; DataFrame → 2D labeled data.

   Here's a clear comparison of **Pandas Series vs DataFrame** 👇

Feature	Series	DataFrame
Definition	A **1-dimensional** labeled array that can hold data of any type (like a column in Excel).	A **2-dimensional** labeled data structure with rows and columns (like a full Excel sheet).
Structure	Single column with index and values.	Table with multiple columns (each can be a Series).
Dimensionality	1D	2D
Data Type	Homogeneous (one data type per Series)	Heterogeneous (different data types per column)
Creation	pd.Series([1,2,3])	pd.DataFrame({'A':[1,2], 'B':[3,4]})
Access Elements	By **index label** or **position**	By **row and column labels** (df.loc[], df.iloc[])
Example	python\nimport pandas as pd\ns = pd.Series([10,20,30])\nprint(s)\n	python\ndf = pd.DataFrame({'A':[10,20], 'B':[30,40]})\nprint(df)\n

---

Key Points

1. A **DataFrame** is a **collection of Series** sharing the same index.
2. **Series → one column**, **DataFrame → multiple columns**.
3. You can convert a Series to DataFrame using s.to_frame().

Interview-Friendly 1-Line

A Series is a one-dimensional labeled array, while a DataFrame is a two-dimensional table made up of multiple Series. ✅

4. **Q:** How to read CSV file?
   A: pd.read_csv('file.csv').

5. **Q:** How to write DataFrame to CSV?
   A: df.to_csv('file.csv', index=False).
6. **Q:** How to get DataFrame shape?
   A: df.shape.
7. **Q:** How to get DataFrame info and column types?
   A: df.info().
8. **Q:** How to get summary statistics?
   A: df.describe().
9. **Q:** How to select a column?
   A: df['A'] or df.A.
10. **Q:** How to select multiple columns?
    A: df[['A','B']].

Indexing & Selection

11. **Q:** Difference between loc and iloc?
    A: loc → label-based; iloc → integer-based.
12. **Q:** Select row by index label?
    A: df.loc[2].
13. **Q:** Select row by integer position?
    A: df.iloc[2].
14. **Q:** Select specific rows and columns?
    A: df.loc[0:2, ['A','B']].
15. **Q:** Conditional selection?
    A: df[df['A']>5].
16. **Q:** Boolean indexing with multiple conditions?
    A: df[(df['A']>5) & (df['B']<10)].
17. **Q:** How to set index column?
    A: df.set_index('A', inplace=True).
18. **Q:** How to reset index?
    A: df.reset_index(inplace=True).
19. **Q:** Select first n rows?
    A: df.head(n).
20. **Q:** Select last n rows?
    A: df.tail(n).

Data Manipulation

21. **Q:** Add a new column?
    A: df['C'] = df['A'] + df['B'].
22. **Q:** Delete a column?
    A: df.drop('C', axis=1, inplace=True).

23. **Q:** Rename columns?
    A: df.rename(columns={'A':'Col1'}, inplace=True).
24. **Q:** Sort DataFrame by column?
    A: df.sort_values('A').
25. **Q:** Sort by multiple columns?
    A: df.sort_values(['A','B'], ascending=[True,False]).
26. **Q:** Fill missing values?
    A: df['A'].fillna(0, inplace=True).
27. **Q:** Drop missing values?
    A: df.dropna(inplace=True).
28. **Q:** Check for missing values?
    A: df.isnull().sum().
29. **Q:** Replace values?
    A: df.replace({'A':1}, 100, inplace=True).
30. **Q:** Apply function to column?
    A: df['A'].apply(lambda x:x*2).

Aggregation & Grouping

31. **Q:** Group by column and aggregate?
    A: df.groupby('A')['B'].sum().
32. **Q:** Group by multiple columns?
    A: df.groupby(['A','B']).mean().
33. **Q:** Pivot table?
    A: df.pivot_table(values='B', index='A', columns='C', aggfunc='sum').
34. **Q:** Count unique values in a column?
    A: df['A'].value_counts().
35. **Q:** Drop duplicate rows?
    A: df.drop_duplicates(inplace=True).
36. **Q:** Keep first occurrence of duplicates?
    A: df.drop_duplicates(keep='first').
37. **Q:** How to get correlation?
    A: df.corr().
38. **Q:** How to get covariance?
    A: df.cov().
39. **Q:** How to get cumulative sum?
    A: df['A'].cumsum().
40. **Q:** How to get cumulative product?
    A: df['A'].cumprod().

Merging, Joining & Reshaping

41. **Q:** Concatenate DataFrames?
    A: pd.concat([df1, df2], axis=0).

42. **Q:** Merge DataFrames like SQL join?
    A: pd.merge(df1, df2, on='key', how='inner').

43. **Q:** Left, right, outer join differences?
    A: left → keep df1 rows; right → df2 rows; outer → all rows.

44. **Q:** Stack and unstack?
    A: df.stack() → columns→rows; df.unstack() → rows→columns.

45. **Q:** Melt DataFrame?
    A: pd.melt(df, id_vars=['A'], value_vars=['B','C']).

46. **Q:** Reshape using pivot?
    A: df.pivot(index='A', columns='B', values='C').

47. **Q:** How to get top n rows per group?
    A: df.groupby('A').head(n).

48. **Q:** How to sample rows randomly?
    A: df.sample(n=5) or frac=0.2 for 20%.

49. **Q:** Convert DataFrame column type?
    A: df['A'] = df['A'].astype(float).

50. **Q:** Apply function across DataFrame rows or columns?
    A: df.apply(func, axis=0) → columns, axis=1 → rows.
