**Pandas Assignment-**

Q1. How do you load a CSV file into a Pandas DataFrame?

Q2. How do you check the data type of a column in a Pandas DataFrame?

Q3. How do you select rows from a Pandas DataFrame based on a condition?

Q4. How do you rename columns in a Pandas DataFrame?

Q5. How do you drop columns in a Pandas DataFrame?

Q6. How do you find the unique values in a column of a Pandas DataFrame?

Q7. How do you find the number of missing values in each column of a Pandas DataFrame?

Q8. How do you fill missing values in a Pandas DataFrame with a specific value?

Q9. How do you concatenate two Pandas DataFrames?

Q10. How do you merge two Pandas DataFrames on a specific column?

Q11. How do you group data in a Pandas DataFrame by a specific column and apply an aggregation function?

Q12. How do you pivot a Pandas DataFrame?

Q13. How do you change the data type of a column in a Pandas DataFrame?

Q14. How do you sort a Pandas DataFrame by a specific column?

Q15. How do you create a copy of a Pandas DataFrame?

Q16. How do you filter rows of a Pandas DataFrame by multiple conditions?

Q17. How do you calculate the mean of a column in a Pandas DataFrame?

Q18. How do you calculate the standard deviation of a column in a Pandas DataFrame?

Q19. How do you calculate the correlation between two columns in a Pandas DataFrame?

Q20. How do you select specific columns in a DataFrame using their labels?

Q21. How do you select specific rows in a DataFrame using their indexes?

Q22. How do you sort a DataFrame by a specific column?

Q23. How do you create a new column in a DataFrame based on the values of another column?

Q24. How do you remove duplicates from a DataFrame?

Q25. What is the difference between .loc and .iloc in Pandas?

**Answers:**

Ans 1. To load a CSV file into a Pandas DataFrame, we can use the **pd.read\_csv()** function, like this:

.

import pandas as pd df = pd.read\_csv('file.csv')

Ans 2. To check the data type of a column in a Pandas DataFrame, we can use the **.dtypes** attribute, like this:

.

df['column\_name'].dtypes

Ans 3. To select rows from a Pandas DataFrame based on a condition, we can use boolean indexing, like this:

.

df[df['column\_name'] == 'value']

Ans 4. To rename columns in a Pandas DataFrame, we can use the **.rename()** function, like this:

.

df = df.rename(columns={'old\_name': 'new\_name'})

Ans 5. To drop columns in a Pandas DataFrame, we can use the **.drop()** function, like this:

.

df = df.drop(columns=['column\_name'])

Ans 6. To find the unique values in a column of a Pandas DataFrame, we can use the **.unique()** function, like this:

.

df['column\_name'].unique()

Ans 7. To find the number of missing values in each column of a Pandas DataFrame, we can use the **.isnull()** function and the **.sum()** function, like this:

.

df.isnull().sum()

Ans 8. To fill missing values in a Pandas DataFrame with a specific value, we can use the **.fillna()** function, like this:

.

df = df.fillna(value)

Ans 9. To concatenate two Pandas DataFrames, we can use the **pd.concat()** function, like this:

.

pd.concat([df1, df2])

Ans 10. To merge two Pandas DataFrames on a specific column, we can use the **pd.merge()** function, like this:

.

pd.merge(df1, df2, on='column\_name')

Ans 11. To group data in a Pandas DataFrame by a specific column and apply an aggregation function, we can use the **.groupby()** function, like this:

.

df.groupby('column\_name').agg({'column\_name': 'function'})

Ans 12. To pivot a Pandas DataFrame, we can use the **.pivot()** function, like this:

.

df.pivot(index='column\_name1', columns='column\_name2', values='column\_name3')

Ans 13. To change the data type of a column in a Pandas DataFrame, we can use the **.astype()** function, like this:

.

df['column\_name'] = df['column\_name'].astype(data\_type)

Ans 14. To sort a Pandas DataFrame by a specific column, we can use the **sort\_values()** function and specify the column name as the parameter. For example:

.

df = df.sort\_values(by='column\_name')

Ans 15. To create a copy of a Pandas DataFrame, we can use the **copy()** function. For example:

.

df\_copy = df.copy()

Ans 16. To filter rows of a Pandas DataFrame by multiple conditions, we can use the **&** (and) or **|** (or) operator to combine conditions. For example:

.

df[(df['column1'] > value1) & (df['column2'] < value2)]

Ans 17. To calculate the mean of a column in a Pandas DataFrame, we can use the **mean()** function. For example:

.

mean\_value = df['column\_name'].mean()

Ans 18. To calculate the standard deviation of a column in a Pandas DataFrame, we can use the **std()** function. For example:

.

std\_value = df['column\_name'].std()

Ans 19. To calculate the correlation between two columns in a Pandas DataFrame, we can use the **corr()** function. For example:

.

corr = df['column1'].corr(df['column2'])

Ans 20. To select specific columns in a DataFrame using their labels, we can use the **[]** operator. For example:

.

df[['column1', 'column2']]

Ans 21. To select specific rows in a DataFrame using their indexes, we can use the **loc[]** or **iloc[]** functions. For example:

.

df.loc[index\_value] df.iloc[index\_position]

Ans 22. To sort a DataFrame by a specific column, we can use the **sort\_values()** function and specify the column name as the parameter. For example:

.

df = df.sort\_values(by='column\_name')

Ans 23. To create a new column in a DataFrame based on the values of another column, we can assign a new value to a new column name. For example:

.

df['new\_column'] = df['column\_name'] \* 2

Ans 24. To remove duplicates from a DataFrame, we can use the **drop\_duplicates()** function. For example:

.

df = df.drop\_duplicates()

Ans 25. What is the difference between .loc and .iloc in Pandas?

The difference between .loc and .iloc in Pandas is that .loc uses the label of the rows and columns, while .iloc uses the integer-based location. For example, df.loc[:, 'column\_name'] will select all rows for a specific column using the column name, while df.iloc[:, 0] will select all rows for the first column. Additionally, .loc can also accept a boolean array for filtering the DataFrame and it can also handle missing values while .iloc cannot.