Name : Aditi Devdhe

Topic : Ques/Ans based on Python Frameworks

Question 1: What is Pandas, and why is it commonly used in data cleaning tasks?

Answer : Pandas is a Python library for data manipulation and analysis. It's commonly used in data cleaning due to its powerful data structures, like DataFrames, which simplify tasks like handling missing values and transforming datasets.

Question 2: Given a DataFrame with missing values, how would you check for missing values in each column and count the total number of missing values?

Answer : To check for missing values, use df.isnull() to get a boolean DataFrame. To count, use df.isnull().sum() for column-wise counts and df.isnull().sum().sum() for the total count.

Question 3: How can you remove duplicates from a DataFrame while retaining the first occurrence of each unique row?

Answer : We can remove duplicates by using df.drop_duplicates(keep='first') while keeping the first occurrence in the DataFrame.

Question 4: If you have a DataFrame with a column containing string values, how can you convert all the values in that column to lowercase?

Answer : We can use df['column_name'] = df['column_name'].str.lower() to convert all string values in a column to lowercase.

Question 5: How do you replace missing values in a DataFrame with a specific value, like 0, for a particular column?

Answer : By using df['column_name'].fillna(0, inplace=True) missing values can be replaced in a specific column with the value 0.

Question 6: If you have a DataFrame with a datetime column, how can you extract the year, month, and day into separate columns?

Answer : Using df['year'] = df['datetime_column'].dt.year, df['month'] = df['datetime_column'].dt.month, and df['day'] = df['datetime_column'].dt.day we can extract year, month, and day into separate columns.

Question 7: How can you filter rows in a DataFrame where a specific column's values meet a certain condition (e.g., all rows where 'age' is greater than 30)?

Answer : By Using df_filtered = df[df['age'] > 30] to filter rows where the 'age' column values are greater than 30.

Question 8: What is the purpose of the .apply() function in Pandas, and how would you use it to create a new column based on values from existing columns?

Answer : The .apply() function is used to apply a function along the axis of a DataFrame. To create a new column based on existing columns, use df['new_column'] = df.apply(lambda row: desired_function(row['col1'], row['col2']), axis=1).

Question 9: Suppose you want to merge two DataFrames, 'df1' and 'df2,' on a common column 'key.' How would you perform this merge operation in Pandas?

Answer : We can perform specified merge operation in Pandas by using merged_df = pd.merge(df1, df2, on='key') to merge 'df1' and 'df2' on the common column 'key'.

Question 10: You have a DataFrame with a column containing messy text data. How can you clean and standardize the text data (e.g., remove punctuation and convert to lowercase) in that column?

Answer : By using df['text_column'] = df['text_column'].str.replace('[^\w\s]', '').str.lower() for removing punctuation and converting text data in a column to lowercase.