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

Answer:- Pandas is a Python library, and it's like a magic tool for cleaning up and organizing data. It's commonly used in data cleaning tasks because it's really good at handling tables of information.

● Question 2: Given a Dataset 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 in each column and count the total number of missing values in a Dataset in Python, you would typically use the .isnull() method to identify missing values in each cell of the Dataset. Then, you would use the .sum() method to count the number of True values (which represent missing values) in each column. Finally, summing up these counts across all columns would give you the total number of missing values in the entire Dataset.

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

Answer:- To remove duplicates from a Dataset use the drop\_duplicates method in pandas.

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

Answer:- To convert all the values in a column of a Dataset to lowercase, use the str.lower() method.

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

Answer:- To replace missing values in a Dataset with a specific value, such as 0, for a particular column, use the fillna() method .

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

Answer:- Use the dt accessor, which helps access and work with datetime properties. Create a new column to store the year of each date entry.

Example:- df['Year'] = df['YourDatetimeColumn'].dt.year

Do the same as month and day, it will create different feature for it.

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

Answer:- To filter rows in a Dataset based on a specific column's values use boolean indexing.

Example:-

filtered\_df = df[df['age'] > 30]

code creates a boolean mask by checking the condition df['age'] > 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 in pandas is a powerful tool that allows you to apply a function along the axis of a Dataset.

Example:- def double\_age(x):

return x \* 2

df['double\_age'] = df['age'].apply(double\_age)

the double\_age function takes a value x and returns x \* 2. Then, use apply() to apply this function to each element in the 'age' column, creating a new column 'double\_age' with the doubled values.

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

Answer:- To merge two Datasets, 'df1' and 'df2,' on a common column 'key' in pandas, you can use the merge() function.

merges the two Datasets (df1 and df2) based on the common column 'key'. The resulting Dataset, merged\_df, will contain columns from both df1 and df2, where the rows are matched based on the values in the 'key' column.

Example:-

merged\_df = pd.merge(df1, df2, on='key', how='left')

merged\_df = pd.merge(df1, df2, on='key', how='right')

merged\_df = pd.merge(df1, df2, on='key', how='outer')

● Question 10: You have a Dataset 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:- Cleaning and standardizing text data in a pandas Dataset involves applying various transformations to a specific column containing text. Create a function that takes a text input and performs the necessary cleaning steps. Use the apply() function to apply your cleaning function to each element in the specified column. can check the results by looking at the new column.

Example:-

def clean\_text(text):

# Remove punctuation

text = ''.join([char for char in text if char not in string.punctuation])

# Convert to lowercase

text = text.lower()

return text

df['cleaned\_text'] = df['text\_column'].apply(clean\_text)

print(df[['text\_column', 'cleaned\_text']])