**TASK – 1(30.08.2024)**

**BY AMITA**

**### \*\*Exercise 5: Handling Missing Values\*\***

**1. Create a DataFrame with missing values:**

**```python**

**data = {**

**"Name": ["Amit", "Neha", "Raj", "Priya"],**

**"Age": [28, None, 35, 29],**

**"City": ["Delhi", "Mumbai", None, "Chennai"]**

**}**

**df = pd.DataFrame(data)**

**```**

import pandas as pd

data = {

"Name": ["Amit", "Neha", "Raj", "Priya"],

"Age": [28, None, 35, 29],

"City": ["Delhi", "Mumbai", None, "Chennai"]

}

df = pd.DataFrame(data)

print("Original DataFrame:\n", df)

**2. Fill missing values in the `"Age"` column with the average age.**

df['Age'].fillna(df['Age'].mean(), inplace=True)

print("\nDataFrame after filling missing 'Age' values with the average:\n", df)

**3. Drop rows where any column has missing data.**

df\_cleaned = df.dropna()

print("\nDataFrame after dropping rows with missing data:\n", df\_cleaned)

**### \*\*Exercise 6: Adding and Removing Columns\*\***

1. **Add a new column `"Salary"` with the following values: `[50000, 60000, 70000, 65000]`.**

df['Salary'] = [50000, 60000, 70000, 65000]

print("\nDataFrame after adding 'Salary' column:\n", df)

1. **Remove the `"City"` column from the DataFrame.**

df = df.drop(columns=['City'])

print("\nDataFrame after removing 'City' column:\n", df)

**### \*\*Exercise 7: Sorting Data\*\***

1. **Sort the DataFrame by `"Age"` in ascending order.**

df\_sorted\_age = df.sort\_values(by='Age')

print("\nDataFrame sorted by 'Age' in ascending order:\n", df\_sorted\_age)

1. **Sort the DataFrame first by `"City"` and then by `"Age"` in descending order.**

df['City'] = ["Delhi", "Mumbai", "Chennai", "Bangalore"]

df\_sorted\_city\_age = df.sort\_values(by=['City', 'Age'], ascending=[True, False])

print("\nDataFrame sorted first by 'City' and then by 'Age' in descending order:\n", df\_sorted\_city\_age)

**### \*\*Exercise 8: Grouping and Aggregation\*\***

1. **Group the DataFrame by `"City"` and calculate the average `"Age"` for each city.**

avg\_age\_by\_city = df.groupby('City')['Age'].mean().reset\_index()

print("\nAverage 'Age' by 'City':\n", avg\_age\_by\_city)

1. **Group the DataFrame by `"City"` and `"Age"`, and count the number of occurrences for each group.**

count\_by\_city\_age = df.groupby(['City', 'Age']).size().reset\_index(name='Count')

print("\nCount of occurrences by 'City' and 'Age':\n", count\_by\_city\_age)

**### \*\*Exercise 9: Merging DataFrames\*\***

**1. Create two DataFrames:A**

**```python**

**df1 = pd.DataFrame({**

**"Name": ["Amit", "Neha", "Raj"],**

**"Department": ["HR", "IT", "Finance"]**

**})**

**df2 = pd.DataFrame({**

**"Name": ["Neha", "Raj", "Priya"],**

**"Salary": [60000, 70000, 65000]**

**})**

**```**

df1 = pd.DataFrame({

"Name": ["Amit", "Neha", "Raj"],

"Department": ["HR", "IT", "Finance"]

})

df2 = pd.DataFrame({

"Name": ["Neha", "Raj", "Priya"],

"Salary": [60000, 70000, 65000]

})

print("DataFrame 1:\n", df1)

print("\nDataFrame 2:\n", df2)

1. **Merge `df1` and `df2` on the `"Name"` column using an inner join.**

df\_inner = pd.merge(df1, df2, on='Name', how='inner')

print("\nDataFrame after inner join on 'Name':\n", df\_inner)

1. **Merge the same DataFrames using a left join.**

df\_left = pd.merge(df1, df2, on='Name', how='left')

print("\nDataFrame after left join on 'Name':\n", df\_left)