Day 13 of Training at Ansh Info Tech

Topics Covered

- **Pandas Library**
  - **DataFrames**
    - Setting Index - `set_index()`, `reset_index()`
    - Handling Missing Values - `dropna()`, `fillna()`
    - Grouping - `groupby()`, `grouped_df.min()` etc.
    - Custom Functions - `.apply(function_name)`
    - Joining - `concat([df, new_row])`
- **Worksheet For Practicing Pandas**
- **Python Exercises**

Summary

Pandas Library - DataFrames

DataFrames in Pandas offer powerful tools for data manipulation:

- **Setting Index**: `set_index()` assigns a column as the index, aiding in quick data lookup; `reset_index()` reverts the index to default numeric labels.
- **Handling Missing Values**: `dropna()` removes rows with missing data; `fillna()` replaces NaN values with specified alternatives, ensuring data completeness.
- **Grouping**: `groupby()` groups data based on specified criteria, facilitating aggregation operations like finding minimum values (`grouped_df.min()`), which is crucial for summarizing data.
- **Custom Functions**: `.apply(function_name)` applies custom functions to elements or columns of a DataFrame, allowing complex transformations or calculations.
- **Joining**: `concat([df, new_row])` concatenates DataFrames along rows or columns, useful for combining datasets or adding new data.

Worksheet For Practicing Pandas

The practice worksheet likely provided exercises to reinforce skills in using Pandas for tasks involving DataFrames, covering scenarios such as data cleaning, aggregation, and joining.

Python Exercises

Engaging in Python exercises enhances programming skills, reinforcing concepts learned in Pandas and enabling broader application of Python programming for data analysis and beyond.

```
import pandas as pd
import numpy as np
# to_csv converts df to csv file
df = pd.read_csv('https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.
print("df.head()")
print(df.head())
print("df.tail()")
print(df.tail())
print("df.info()")
print(df.info())
print("df.describe()")
print(df.describe())
print("df.sample()")
print(df.sample())
print("df.shape")
print(df.shape)
print("df.columns")
print(df.columns)
```

```
df.head()
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
df.tail()
     PassengerId  Survived  Pclass                                      Name  \
886          887         0       2                     Montvila, Rev. Juozas
887          888         1       1              Graham, Miss. Margaret Edith
888          889         0       3  Johnston, Miss. Catherine Helen "Carrie"
889          890         1       1                     Behr, Mr. Karl Howell
890          891         0       3                       Dooley, Mr. Patrick

        Sex   Age  SibSp  Parch      Ticket   Fare Cabin Embarked
886    male  27.0      0      0      211536  13.00   NaN        S
```

```
887  female  19.0      0      0      112053  30.00   B42        S
888  female   NaN      1      2  W./C. 6607  23.45   NaN        S
889    male  26.0      0      0      111369  30.00  C148        C
890    male  32.0      0      0      370376   7.75   NaN        Q
df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
None
df.describe()
       PassengerId    Survived      Pclass         Age       SibSp  \
```

```
df.reset_index() #Change the index names back to index numbers 0,1,2...
```

	index	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticke
0	0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 2117
1	1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 1759
2	2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2 310128
3	3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	11380
4	4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	37345
...	...	...	...	...	...	...	...	...	...	..
886	886	887	0	2	Montvila, Rev.	male	27.0	0	0	21153

```
df.set_index("PassengerId")
```

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
PassengerId									
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500

```
df.isnull()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
0	False	False	False	False	False	False	False	False	False	False	True
1	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	True
3	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	True
...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	True
887	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	False	False	True	False	False	False	False	True
889	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	True

891 rows × 12 columns

```
df.isnull().sum()
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

```
df.dropna(axis=1)
df.set_index('Fare')
df.index.values
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-37-d8793a10f75e> in <cell line: 2>()
      1 df.dropna(axis=1)
----> 2 df.set_index('Fare')
      3 df.index.values

/usr/local/lib/python3.10/dist-packages/pandas/core/frame.py in set_index(self, keys,
drop, append, inplace, verify_integrity)
   5857
   5858         if missing:
-> 5859             raise KeyError(f"None of {missing} are in the columns")
   5860
   5861         if inplace:

KeyError: "None of ['Fare'] are in the columns"
```

```
df2 = df.iloc[5:15,5:15]
df2
```

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
5	NaN	0	0	330877	8.4583	NaN	Q
6	54.0	0	0	17463	51.8625	E46	S
7	2.0	3	1	349909	21.0750	NaN	S
8	27.0	0	2	347742	11.1333	NaN	S
9	14.0	1	0	237736	30.0708	NaN	C
10	4.0	1	1	PP 9549	16.7000	G6	S
11	58.0	0	0	113783	26.5500	C103	S
12	20.0	0	0	A/5. 2151	8.0500	NaN	S
13	39.0	1	5	347082	31.2750	NaN	S
14	14.0	0	0	350406	7.8542	NaN	S

```
df2.dropna(axis=1, thresh = 2)
```

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
5	NaN	0	0	330877	8.4583	NaN	Q
6	54.0	0	0	17463	51.8625	E46	S
7	2.0	3	1	349909	21.0750	NaN	S
8	27.0	0	2	347742	11.1333	NaN	S
9	14.0	1	0	237736	30.0708	NaN	C
10	4.0	1	1	PP 9549	16.7000	G6	S
11	58.0	0	0	113783	26.5500	C103	S
12	20.0	0	0	A/5. 2151	8.0500	NaN	S
13	39.0	1	5	347082	31.2750	NaN	S
14	14.0	0	0	350406	7.8542	NaN	S

```
df2.fillna(method='pad')
```

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
5	NaN	0	0	330877	8.4583	NaN	Q
6	54.0	0	0	17463	51.8625	E46	S
7	2.0	3	1	349909	21.0750	E46	S
8	27.0	0	2	347742	11.1333	E46	S
9	14.0	1	0	237736	30.0708	E46	C
10	4.0	1	1	PP 9549	16.7000	G6	S
11	58.0	0	0	113783	26.5500	C103	S
12	20.0	0	0	A/5. 2151	8.0500	C103	S
13	39.0	1	5	347082	31.2750	C103	S
14	14.0	0	0	350406	7.8542	C103	S

```
df2.fillna(method='bfill')
```

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
5	54.0	0	0	330877	8.4583	E46	Q
6	54.0	0	0	17463	51.8625	E46	S
7	2.0	3	1	349909	21.0750	G6	S
8	27.0	0	2	347742	11.1333	G6	S
9	14.0	1	0	237736	30.0708	G6	C
10	4.0	1	1	PP 9549	16.7000	G6	S
11	58.0	0	0	113783	26.5500	C103	S
12	20.0	0	0	A/5. 2151	8.0500	NaN	S
13	39.0	1	5	347082	31.2750	NaN	S
14	14.0	0	0	350406	7.8542	NaN	S

```
df2.fillna(method='ffill')
```

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
5	NaN	0	0	330877	8.4583	NaN	Q
6	54.0	0	0	17463	51.8625	E46	S
7	2.0	3	1	349909	21.0750	E46	S
8	27.0	0	2	347742	11.1333	E46	S
9	14.0	1	0	237736	30.0708	E46	C
10	4.0	1	1	PP 9549	16.7000	G6	S
11	58.0	0	0	113783	26.5500	C103	S
12	20.0	0	0	A/5. 2151	8.0500	C103	S
13	39.0	1	5	347082	31.2750	C103	S
14	14.0	0	0	350406	7.8542	C103	S

```
df2['Age'].fillna(df2['Age'].mean().astype(int))
```

```
5     25.0
6     54.0
7      2.0
8     27.0
9     14.0
10     4.0
11    58.0
12    20.0
```

```
     13    39.0
     14    14.0
     Name: Age, dtype: float64
```

```
grouped_df = df.groupby('Age')
```

```
grouped_df['Fare'].max()
```

```
     Age
     0.42       8.5167
     0.67      14.5000
     0.75      19.2583
     0.83      29.0000
     0.92     151.5500
                ...
     70.00     71.0000
     70.50      7.7500
     71.00     49.5042
     74.00      7.7750
     80.00     30.0000
     Name: Fare, Length: 88, dtype: float64
```

```
def give_tip(fare):
  return fare+100
grouped_df['Fare'].apply(give_tip)
```

```
     Age
     0.42   803    108.5167
     0.67   755    114.5000
     0.75   469    119.2583
            644    119.2583
     0.83   78     129.0000
                     ...
     70.50  116    107.7500
     71.00  96     134.6542
            493    149.5042
     74.00  851    107.7750
     80.00  630    130.0000
     Name: Fare, Length: 714, dtype: float64
```

Start coding or generate with AI.

```
df.fillna(5, inplace = True)
df
df.set_index('Company')
print(df.index.values)
```

```
     [0 1 2 3 4 5 6 7]
```

```
df.fillna(df.mean())
```

	A	B	C	D
0	1.0	5.0	10	NaN
1	2.0	5.0	20	NaN
2	3.0	5.0	30	NaN
3	2.0	5.0	40	NaN

```
df.fillna(0)
```

	A	B	C	D
0	1.0	5.0	10	0.0
1	2.0	0.0	20	0.0
2	3.0	0.0	30	0.0
3	0.0	0.0	40	0.0

Grouping

```
d = {"Company":["FB", "GOOGLE", "MICROSOFT", "FB", "GOOGLE", "FB", "MICROSOFT", "FB"],
    "Employee":["Sam", "Rachel", "Maddy", "Joe", "Srishti", "Shivay", "Pushpa", "Kirti"],
    "Sales":[1000, 500, 550, 2000, 890, 500, 350, 350]}

df = pd.DataFrame(d)
df
```

	Company	Employee	Sales
0	FB	Sam	1000
1	GOOGLE	Rachel	500
2	MICROSOFT	Maddy	550
3	FB	Joe	2000
4	GOOGLE	Srishti	890
5	FB	Shivay	500
6	MICROSOFT	Pushpa	350
7	FB	Kirti	350

```
df.min()
```

```
Company       FB
Employee     Joe
Sales        350
dtype: object
```

```
df.max()
```

```
Company     MICROSOFT
Employee      Srishti
Sales            2000
dtype: object
```

```
grouped_df = df.groupby('Company')
grouped_df
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7fee3c692df0>
```

```
grouped_df.min()
```

	Employee	Sales
Company		
FB	Joe	350
GOOGLE	Rachel	500
MICROSOFT	Maddy	350

```
grouped_df.max()
```

	Employee	Sales
Company		
FB	Shivay	2000
GOOGLE	Srishti	890
MICROSOFT	Pushpa	550

```
grouped_df.describe()
```

	Sales							
	count	mean	std	min	25%	50%	75%	max
Company								
FB	4.0	962.5	745.402576	350.0	462.5	750.0	1250.0	2000.0
GOOGLE	2.0	695.0	275.771645	500.0	597.5	695.0	792.5	890.0
MICROSOFT	2.0	450.0	141.421356	350.0	400.0	450.0	500.0	550.0

```
df.describe()
```

	Sales
count	8.000000
mean	767.500000
std	551.251822
min	350.000000
25%	462.500000
50%	525.000000
75%	917.500000
max	2000.000000

Custom Functions

```
def give_bonus(sales):
    return sales + 100
```

```
df['Sales'].apply(give_bonus)
```

```
0    1100
1     600
2     650
3    2100
4     990
5     600
6     450
7     450
Name: Sales, dtype: int64
```

```
df['Sales'] = df['Sales'].apply(lambda sales : sales + 100)
```

```
df
```

	Company	Employee	Sales
0	FB	Sam	1100
1	GOOGLE	Rachel	600
2	MICROSOFT	Maddy	650
3	FB	Joe	2100
4	GOOGLE	Srishti	990
5	FB	Shivay	600
6	MICROSOFT	Pushpa	450
7	FB	Kirti	450

Joining

```
new_employee = pd.DataFrame({'Company':['GOOGLE'], 'Employee':['Kriti'], 'Sales':[5000]})
new_employee
```

	Company	Employee	Sales
0	GOOGLE	Kriti	5000

```
df = pd.concat([df, new_employee])
df
```

	Company	Employee	Sales
0	FB	Sam	1100
1	GOOGLE	Rachel	600
2	MICROSOFT	Maddy	650
3	FB	Joe	2100
4	GOOGLE	Srishti	990
5	FB	Shivay	600
6	MICROSOFT	Pushpa	450
7	FB	Kirti	450
0	GOOGLE	Kriti	5000

```
df.index.values[-1] = 8
```

```
df
```

	Company	Employee	Sales
0	FB	Sam	1100
1	GOOGLE	Rachel	600
2	MICROSOFT	Maddy	650
3	FB	Joe	2100
4	GOOGLE	Srishti	990
5	FB	Shivay	600
6	MICROSOFT	Pushpa	450
7	FB	Kirti	450
8	GOOGLE	Kriti	5000

```
another_employee = pd.DataFrame({'Company':['INFOSYS'], 'Employee':['XYZ'], 'Gender':['M']})
another_employee
```

	Company	Employee	Gender
0	INFOSYS	XYZ	M

```
pd.concat([df, another_employee])
```

	Company	Employee	Sales	Gender
0	FB	Sam	1100.0	NaN
1	GOOGLE	Rachel	600.0	NaN
2	MICROSOFT	Maddy	650.0	NaN
3	FB	Joe	2100.0	NaN