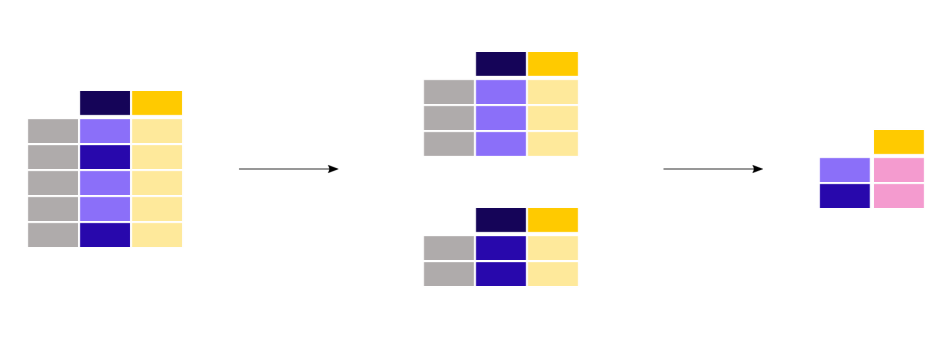
Aggregating statistics grouped by category



The groupby method is used to support this type of operations. This fits in the more general split-apply-combine pattern:

* **Split** the data into groups
* **Apply** a function to each group independently
* **Combine** the results into a data structure

Output according to the command

**titanic.groupby("Sex")["Age"].mean()**

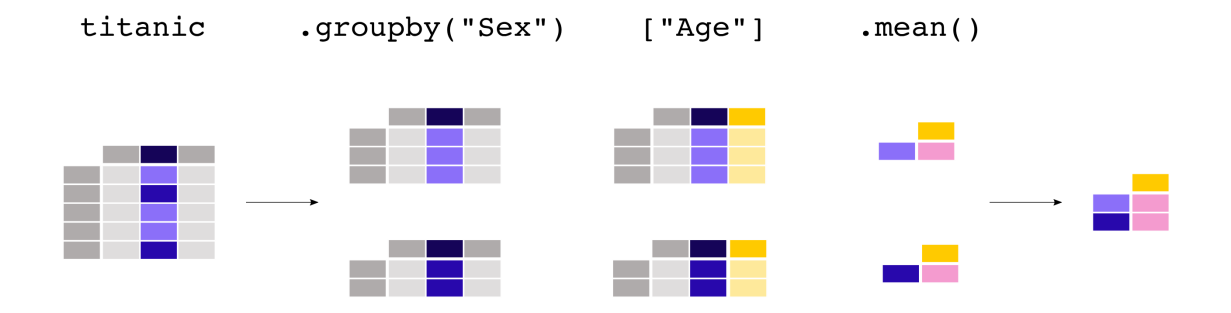
Out[10]:

Sex

female 27.915709

male 30.726645

Name: Age, dtype: float64



Grouping can be done by multiple columns at the same time. Provide the column names as a list to the **[groupby()](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html" \l "pandas.DataFrame.groupby" \o "pandas.DataFrame.groupby)** method.

**Key points for groupby() function**

* groupby() is used to split data into groups based on one or more keys, allowing for efficient analysis and aggregation of grouped data.
* It supports various aggregation functions like sum, mean, count, min, and max, which can be applied to each group.
* You can apply multiple aggregations on different columns using .agg(), offering more flexibility in analysis.
* The result of groupby() often returns a DataFrame with a MultiIndex, where each level represents a grouping key.
* You can filter groups based on specific conditions by using .filter() after groupby().
* groupby() allows iteration over groups, enabling customized operations on each subset of data.

**Other important points**

1. groupby() is a powerful function in pandas that is used for grouping data based on some criteria. It enables you to split a DataFrame into groups based on one or more columns and then apply a function (such as aggregation, transformation, or filtering) to each group independently.
2. The basic syntax of groupby() involves selecting a column or columns to group by and then applying an aggregation function.
3. To apply custom functions with groupby() in pandas, you can use either the apply() method or the agg() method.
4. You can perform transformations within each group using the transform() method with groupby() in pandas. The transform() method is particularly useful when you want to maintain the original shape of the DataFrame but apply a transformation to each group independently.
5. You can sort the results of groupby() in pandas using the sort\_values() method. For example, grouped\_data contains the mean value for each category. The sort\_values() method is then used to sort the result in descending order based on the mean values.

Generay example

print('general example of groupby function apply at two columns and sum of salry column')

# import pandas

import pandas as pd

#create a dictinory contain the dataset

data={

    'Name': ['Alice', 'Bob', 'Charlie', 'David','sonu', 'Rahul','Ravi','Bob'],

    'Age': [24, 30, 22, 35,24,30, 22, 35],

    'City': ['New York', ' Houston', 'Chicago', 'Houston', 'New York', 'Los Angeles', 'Chicago', 'Houston'],

    'Salary': [70000, 80000, 60000, 90000, 70000, 80000, 60000, 90000],

    'Department': ['HR', 'Finance', 'IT', 'Marketing', 'HR', 'Finance', 'IT', 'Marketing'],

    'Experience': [2, 5, 1, 8, 2, 5, 1, 8],

    'Country': ['USA', 'USA', 'USA', 'USA', 'India', 'India', 'India', 'India'],

    'Job Title': ['Manager', 'Senior Developer', 'Software Engineer', 'Marketing Manager', 'Manager', 'Senior Developer', 'Software Engineer', 'Marketing Manager']

}

# Create a DataFrame from the dictionary

df = pd.DataFrame(data)

# Display the DataFrame

df

#group the dataframe by 'Name' and 'City' and calculate the sum of  'Salary'

grouped= df.groupby(['Name', 'City'])['Salary'].sum().reset\_index()

# Display the grouped DataFrame

print(grouped)

output

Name City Salary

0 Alice New York 70000

1 Bob Houston 80000

2 Bob Houston 90000

3 Charlie Chicago 60000

4 David Houston 90000

5 Rahul Los Angeles 80000

6 Ravi Chicago 60000

7 sonu New York 70000

Example of groupby() a single column

print('create a datframe and group by a single column')

# import pandas

import pandas as pd

# create a dictionary containing the dataset

data = {

    'category':['Electronis','cloths','furniture','Electronis','cloths','furniture','woodwork','cement','construction'],

    'product':['laptop','shirt','table','mobile','jeans','chair','wooden chair','cement bag','bricks'],

    'price':[1000,50,200,800,60,150,100,120,10],

    'quantity':[5,10,2,3,8,4,6,12,15],

    'sales':[5000,500,400,2400,480,600,600,1440,150],

    'profit':[1000,200,50,800,120,150,60, 240,30]

    }

# Create a DataFrame from the dictionary

df = pd.DataFrame(data)

# Display the DataFrame

df

#group by datframe according to category and sum of sales column

grouped=df.groupby('category') ['sales'].sum()

#display the grouped dataframe

print(grouped)

output

create a datframe and group by a single column

category

Electronis 7400

cement 1440

cloths 980

construction 150

furniture 1000

woodwork 600

Name: sales, dtype: int64

**Group by a Multiple Column in Pandas**

We can also group multiple columns and calculate multiple aggregates in Pandas.

#import pandsa

import pandas as pd

#crete a dictinory of students dataset

data={

    'gender':['Male','Female','Male','Female','Male','Female','Male','Female','Male','Female'],

    'name':['John','Alice','Bob','Emma','Mike','Sophia','David','Olivia','James','Isabella'],

    'age':[20,21,22,23,24,25,26,27,28,29],

    'grade':[85,90,78,88,92,95,80,87,89,91],

    'city':['New York','Los Angeles','Chicago','Houston','Phoenix','San Diego','Dallas','San Jose','Austin','Seattle'],

    'country':['USA','USA','USA','USA','USA','USA','USA','USA','USA','USA'],

    'major':['Computer Science','Mathematics','Physics','Chemistry','Biology','Engineering','Economics','Psychology','History','Sociology']

}

# Create a DataFrame from the dictionary

df = pd.DataFrame(data)

# Display the DataFrame

df

#defined aggregate funvction on the grade column

agg\_function={

    'grade':['mean','max','min']

}

#groupby at two columns and min/max grade of students

#grouped=df.groupby(['gender','name']).agg({'grade':['mean','max']}).reset\_index() #this is also correct method

grouped=df.groupby(['gender','age']).agg(agg\_function).reset\_index()

# Display the grouped DataFrame

print(grouped)

Output

gender age grade

mean max min

0 Female 21 90.0 90 90

1 Female 23 88.0 88 88

2 Female 25 95.0 95 95

3 Female 27 87.0 87 87

4 Female 29 91.0 91 91

5 Male 20 85.0 85 85

6 Male 22 78.0 78 78

7 Male 24 92.0 92 92

8 Male 26 80.0 80 80

9 Male 28 89.0 89 89

**Group With Categorical Data**

We group with [*categorical data*](https://www.programiz.com/python-programming/pandas/categorical) where we want to analyze data based on specific categories.

Pandas provides powerful tools to work with categorical data efficiently using the groupby() function.

print('groupby function used in the categorical data')

#import pandas

import pandas as pd

#create a dictionary containing the dataset

data={

    'catagory':['a','b','c','a','b','c','a','b','c'],

    'Values':[10,20,30,40,50,60,70,80,90]

}

#craeet dataframe from the dictionary

df=pd.DataFrame(data)

#convert the 'catagory' column to categorical type

df['catagory'] = df['catagory'].astype('category')

#group by category column and calculate the sum of values

grouped=df.groupby('catagory') ['Values'].sum().reset\_index()

#display the grouped dataframe

print(grouped)

output

groupby function used in the categorical data

catagory Values

0 a 120

1 b 150

2 c 180

Here, first the Category column is converted to a categorical data type using pd.Categorical().

The data is then grouped by the Category column using the groupby() function. And the total values for each category are calculated using the sum() aggregation function.