Groupby , pivot and Cross table

Importing Employee Dataset for performing the Operations

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
# lets read the dataset from employee.csv
df = pd.read_csv("employee.csv")
```

In [4]:
```
# let's check the head of the dataset
df.head()
```

Out[4]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYea
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...	1	80	0	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...	4	80	1	1
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...	2	80	0	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...	3	80	0	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...	4	80	1	

5 rows × 35 columns

Groupby Function

The general syntax is `groupby(data, by, ...)` .

- `data` is a dataframe
- `by` columns on which grouping is to be done
- `agg` the aggregate function can be used separately

Let's see some examples.

In [6]:
```
# let's groupby the departments and their mean age

df[["Age", "Department"]].groupby(["Department"]).agg('mean')
```

Out[6]:

Department	Age
Human Resources	37.809524
Research & Development	37.042664
Sales	36.542601

In [8]:
```
# let's groupby the departments and their maximum age

df[["Age", "Department"]].groupby(["Department"]).agg('max')
```

Out[8]:

Department	Age
Human Resources	59
Research & Development	60
Sales	60

In [7]:
```
# let's groupby the departments and their minimum age

df[["Age", "Department"]].groupby(["Department"]).agg('min')
```

Out[7]:

Department	Age
Human Resources	19
Research & Development	18
Sales	18

In [9]:
```
# trying to check Different Departments and their Mean Salaries in each of the Education Fields.

df[["Department", "EducationField", "MonthlyRate"]].groupby(["Department", "EducationField"]).agg('mean')
```

Out[9]:

Department	EducationField	MonthlyRate
Human Resources	Human Resources	14810.740741
	Life Sciences	12813.875000
	Medical	12668.230769
	Other	9275.000000
	Technical Degree	13158.500000
Research & Development	Life Sciences	14594.704545
	Medical	14163.603306
	Other	13051.765625
	Technical Degree	14142.393617
Sales	Life Sciences	14523.786667
	Marketing	14076.943396
	Medical	15077.625000
	Other	15004.400000
	Technical Degree	14522.029412

Pivot Tables Function

- We can create a spreadsheet-style pivot table as a DataFrame. The levels in the pivot table will be stored in MultiIndex objects (hierarchical indexes) on the index and columns of the result DataFrame.

The general syntax is `pivot_table(data, values=None, index=None, columns=None, aggfunc='mean', ...)` .

- `data` is a dataframe
- `values` contains the column to aggregate
- `index` is the row in the pivot table
- `columns` contains the columns you want in the pivot table
- `aggfunc` is the aggregate function

Let's see some examples.

In [10]:
```
# let's make a pivot table for the department and their mean ages

df.pivot_table(values="Age",index ="Department", aggfunc="mean")
```

Out[10]:

Department	Age
Human Resources	37.809524
Research & Development	37.042664
Sales	36.542601

In [11]:
```
# let's try making a pivot table for department and their maximum ages

df.pivot_table(values="Age",index ="Department", aggfunc="max")
```

Out[11]:

Department	Age
Human Resources	59
Research & Development	60
Sales	60

Crosstab Function

- Compute a simple cross tabulation of two (or more) factors. By default computes a frequency table of the factors unless an array of values and an aggregation function are passed

The general syntax is `crosstab(data, values=None, index=None, columns=None, aggfunc='mean', ...)` .

- `data` is a dataframe
- `values` contains the column to aggregate
- `index` is the row in the pivot table
- `columns` contains the columns you want in the pivot table
- `aggfunc` is the aggregate function

Let's see some examples.

In [10]:
```
# make a cross table for the department and their mean ages
a =df["Age"]
age=df["Department"]
#c=df["Department"]
pd.crosstab( a, [age]).mean()
```

Out[10]:
```
Department
Human Resources            1.465116
Research & Development    22.348837
Sales                     10.372093
dtype: float64
```