```
pip install pandas
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: numpy>=1.26.0 in /usr/local/lib/python3.12
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/p
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.1
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist
```

```
import pandas as pd
```

```
data = {
    "Employee_ID": [101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112
    "Name": ["Amit", "Sneha", "Ravi", "Priya", "Arjun", "Neha", "Karan", "Isha"
    "Department": ["HR", "Finance", "IT", "IT", "Sales", "HR", "Finance", "IT"
    "Experience_Years": [2, 5, 3, 8, 4, 1, 6, 7, 5, 4, 2, 9, 10, 3, 1],
    "Salary": [35000, 55000, 48000, 80000, 60000, 32000, 70000, 75000, 65000, 5
}
```

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

	Employee_ID	Name	Department	Experience_Years	Salary
0	101	Amit	HR	2	35000
1	102	Sneha	Finance	5	55000
2	103	Ravi	IT	3	48000
3	104	Priya	IT	8	80000
4	105	Arjun	Sales	4	60000
5	106	Neha	HR	1	32000
6	107	Karan	Finance	6	70000
7	108	Isha	IT	7	75000
8	109	Vikram	Sales	5	65000
9	110	Tanya	Finance	4	58000
10	111	Rohit	HR	2	36000
11	112	Simran	IT	9	90000
12	113	Raj	Sales	10	95000
13	114	Pooja	Finance	3	50000
14	115	Nikhil	HR	1	33000

```
#Show first 5 rows.

df.head()
```

	Employee_ID	Name	Department	Experience_Years	Salary
0	101	Amit	HR	2	35000
1	102	Sneha	Finance	5	55000
2	103	Ravi	IT	3	48000
3	104	Priya	IT	8	80000
4	105	Arjun	Sales	4	60000

```
#Show last 5 rows:

df.tail()
```

	Employee_ID	Name	Department	Experience_Years	Salary
10	111	Rohit	HR	2	36000
11	112	Simran	IT	9	90000
12	113	Raj	Sales	10	95000
13	114	Pooja	Finance	3	50000
14	115	Nikhil	HR	1	33000

```
#Show first 2 rows
df.head(2)
```

	Employee_ID	Name	Department	Experience_Years	Salary
0	101	Amit	HR	2	35000
1	102	Sneha	Finance	5	55000

```
#Show last 3 rows

df.tail(3)
```

	Employee_ID	Name	Department	Experience_Years	Salary
12	113	Raj	Sales	10	95000
13	114	Pooja	Finance	3	50000
14	115	Nikhil	HR	1	33000

```
#Show data type of rows
df.dtypes
```

	0
Employee_ID	int64
Name	object
Department	object
Experience_Years	int64
Salary	int64

dtype: object

```
#Show the number of rows and columns.

df.shape
```

```
(15, 5)
```

```
#Show stats for the data

df.describe()
```

	Employee_ID	Experience_Years	Salary
count	15.000000	15.000000	15.000000
mean	108.000000	4.666667	58800.000000
std	4.472136	2.845213	20424.774872
min	101.000000	1.000000	32000.000000
25%	104.500000	2.500000	42000.000000
50%	108.000000	4.000000	58000.000000
75%	111.500000	6.500000	72500.000000
max	115.000000	10.000000	95000.000000

```
#Find out mean of experience years or Find out the avg experience years.

Exp_mean=df['Experience_Years'].mean()
Exp_mean
```

```
np.float64(4.666666666666667)
```

```
#Find the median and mode also.

Exp_median=df['Experience_Years'].mode()
Exp_median
```

	Experience_Years
0	1
1	2
2	3
3	4
4	5

dtype: int64

```
Exp_mode=df["Experience_Years"].mode()
Exp_mode
```

	Experience_Years
0	1
1	2
2	3
3	4
4	5

dtype: int64

```
#Find average salary given.

Salary_mean=df['Salary'].mean()
Salary_mean
```

```
np.float64(58800.0)
```

```
#Show all column names

df.columns
```

```
Index(['Employee_ID', 'Name', 'Department', 'Experience_Years',
'Salary'], dtype='object')
```

```
#Show all values

df.values
```

```
array([[101, 'Amit', 'HR', 2, 35000],
       [102, 'Sneha', 'Finance', 5, 55000],
       [103, 'Ravi', 'IT', 3, 48000],
       [104, 'Priya', 'IT', 8, 80000],
```

```
        [105, 'Arjun', 'Sales', 4, 60000],
        [106, 'Neha', 'HR', 1, 32000],
        [107, 'Karan', 'Finance', 6, 70000],
        [108, 'Isha', 'IT', 7, 75000],
        [109, 'Vikram', 'Sales', 5, 65000],
        [110, 'Tanya', 'Finance', 4, 58000],
        [111, 'Rohit', 'HR', 2, 36000],
        [112, 'Simran', 'IT', 9, 90000],
        [113, 'Raj', 'Sales', 10, 95000],
        [114, 'Pooja', 'Finance', 3, 50000],
        [115, 'Nikhil', 'HR', 1, 33000]], dtype=object)
```

```
#Show the data briefly (summary/content)
#Or
#Display information of dataframe.

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 5 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   Employee_ID       15 non-null     int64 
 1   Name              15 non-null     object
 2   Department        15 non-null     object
 3   Experience_Years  15 non-null     int64 
 4   Salary            15 non-null     int64 
dtypes: int64(3), object(2)
memory usage: 732.0+ bytes
```

```
#Access info with integer index.

df.iloc[5,1]
```

```
'Neha'
```

```
#Access info with column label.

df.loc[0,'Name']
```

```
'Amit'
```

```
#Access info about Neha.

df.loc[5,'Name']
```

```
'HR'
```

```
df.iloc[5,1]
```

```
'Neha'
```

```
#Filter the rows for experience years more than 2.

Filter_exp=df.query('Experience_Years > 2')
Filter_exp
```

	Employee_ID	Name	Department	Experience_Years	Salary
1	102	Sneha	Finance	5	55000
2	103	Ravi	IT	3	48000
3	104	Priya	IT	8	80000
4	105	Arjun	Sales	4	60000
6	107	Karan	Finance	6	70000
7	108	Isha	IT	7	75000
8	109	Vikram	Sales	5	65000
9	110	Tanya	Finance	4	58000
11	112	Simran	IT	9	90000
12	113	Raj	Sales	10	95000
13	114	Pooja	Finance	3	50000

```
#Filter for salary >35,000.

Filter_salary=df.query('Salary > 35000')
Filter_salary
```

	Employee_ID	Name	Department	Experience_Years	Salary
1	102	Sneha	Finance	5	55000
2	103	Ravi	IT	3	48000
3	104	Priya	IT	8	80000
4	105	Arjun	Sales	4	60000
6	107	Karan	Finance	6	70000
7	108	Isha	IT	7	75000
8	109	Vikram	Sales	5	65000
9	110	Tanya	Finance	4	58000
10	111	Rohit	HR	2	36000
11	112	Simran	IT	9	90000
12	113	Raj	Sales	10	95000
13	114	Pooja	Finance	3	50000

```
#Remove employee id column AND name together


drop_id=df.drop(columns=['Employee_ID'] and ['Name'])
drop_id
```

	Employee_ID	Department	Experience_Years	Salary
0	101	HR	2	35000
1	102	Finance	5	55000
2	103	IT	3	48000
3	104	IT	8	80000
4	105	Sales	4	60000
5	106	HR	1	32000
6	107	Finance	6	70000
7	108	IT	7	75000
8	109	Sales	5	65000
9	110	Finance	4	58000
10	111	HR	2	36000
11	112	IT	9	90000
12	113	Sales	10	95000
13	114	Finance	3	50000
14	115	HR	1	33000

```
#df.drop(columns=['Age'], inplace=True) , this will permanently drop the age c
```

```
#Change Experience_Years to Years of work.

Rename_Exp=df.rename(columns={'Experience_Years' : 'Years of work'})
Rename_Exp
```

	Employee_ID	Name	Department	Years of work	Salary
0	101	Amit	HR	2	35000
1	102	Sneha	Finance	5	55000
2	103	Ravi	IT	3	48000
3	104	Priya	IT	8	80000
4	105	Arjun	Sales	4	60000
5	106	Neha	HR	1	32000
6	107	Karan	Finance	6	70000
7	108	Isha	IT	7	75000
8	109	Vikram	Sales	5	65000
9	110	Tanya	Finance	4	58000
10	111	Rohit	HR	2	36000
11	112	Simran	IT	9	90000
12	113	Raj	Sales	10	95000
13	114	Pooja	Finance	3	50000
14	115	Nikhil	HR	1	33000

```
#Same goes for rename: to permanently rename the column name,
#use df_renamed = df.rename(columns={'Name': 'Full Name'}, inplace=True
```

```
#Sort out the experience years of employees in ascending order.
#Sort is used for making things in ascending order.

Sort_exp= df.sort_values(by='Experience_Years')
Sort_exp
```

	Employee_ID	Name	Department	Experience_Years	Salary
5	106	Neha	HR	1	32000
14	115	Nikhil	HR	1	33000
10	111	Rohit	HR	2	36000
0	101	Amit	HR	2	35000
2	103	Ravi	IT	3	48000
13	114	Pooja	Finance	3	50000
9	110	Tanya	Finance	4	58000
4	105	Arjun	Sales	4	60000
8	109	Vikram	Sales	5	65000
1	102	Sneha	Finance	5	55000
6	107	Karan	Finance	6	70000
7	108	Isha	IT	7	75000
3	104	Priya	IT	8	80000
11	112	Simran	IT	9	90000
12	113	Raj	Sales	10	95000

```
#Fill missing values with 0

Fillna_0=df.fillna(0)
Fillna_0
```

	Employee_ID	Name	Department	Experience_Years	Salary
0	101	Amit	HR	2	35000
1	102	Sneha	Finance	5	55000
2	103	Ravi	IT	3	48000
3	104	Priya	IT	8	80000
4	105	Arjun	Sales	4	60000
5	106	Neha	HR	1	32000
6	107	Karan	Finance	6	70000
7	108	Isha	IT	7	75000
8	109	Vikram	Sales	5	65000
9	110	Tanya	Finance	4	58000
10	111	Rohit	HR	2	36000
11	112	Simran	IT	9	90000
12	113	Raj	Sales	10	95000
13	114	Pooja	Finance	3	50000
14	115	Nikhil	HR	1	33000

```
#Remove duplicate objects if any.

Drop_dupl=df.drop_duplicates()
Drop_dupl
```

	Employee_ID	Name	Department	Experience_Years	Salary
0	101	Amit	HR	2	35000
1	102	Sneha	Finance	5	55000
2	103	Ravi	IT	3	48000
3	104	Priya	IT	8	80000
4	105	Arjun	Sales	4	60000
5	106	Neha	HR	1	32000
6	107	Karan	Finance	6	70000
7	108	Isha	IT	7	75000
8	109	Vikram	Sales	5	65000
9	110	Tanya	Finance	4	58000
10	111	Rohit	HR	2	36000
11	112	Simran	IT	9	90000
12	113	Raj	Sales	10	95000
13	114	Pooja	Finance	3	50000
14	115	Nikhil	HR	1	33000

```
#Replace and rename have a difference:
#Rename: to change label, ex: City to location.
#Replace: to change value, ex: David to davidoff.

#Replace Neha with Nehita in the dataset.

Replace_Neha=df.replace({'Neha' : 'Nehita'})
Replace_Neha
```

	Employee_ID	Name	Department	Experience_Years	Salary
0	101	Amit	HR	2	35000
1	102	Sneha	Finance	5	55000
2	103	Ravi	IT	3	48000
3	104	Priya	IT	8	80000
4	105	Arjun	Sales	4	60000
5	106	Nehita	HR	1	32000
6	107	Karan	Finance	6	70000
7	108	Isha	IT	7	75000
8	109	Vikram	Sales	5	65000
9	110	Tanya	Finance	4	58000
10	111	Rohit	HR	2	36000
11	112	Simran	IT	9	90000
12	113	Raj	Sales	10	95000
13	114	Pooja	Finance	3	50000
14	115	Nikhil	HR	1	33000

```
#Use group by for Salary and Name:

Group_df=df.groupby('Name')['Salary'].sum()
Group_df
```

Name	Salary
Amit	35000
Arjun	60000
Isha	75000
Karan	70000
Neha	32000
Nikhil	33000
Pooja	50000
Priya	80000
Raj	95000
Ravi	48000
Rohit	36000
Simran	90000
Sneha	55000
Tanya	58000
Vikram	65000

dtype: int64

```
#Now write upto 5 names in grouped data.

Group_df.head(5)
```

Name	Salary
Amit	35000
Arjun	60000
Isha	75000
Karan	70000
Neha	32000

dtype: int64

```
#Find out aggregate for salary with names.

Agg_df=df.groupby('Name').agg({'Salary':'mean'})
Agg_df
```

Name	Salary
Amit	35000.0
Arjun	60000.0
Isha	75000.0
Karan	70000.0
Neha	32000.0
Nikhil	33000.0
Pooja	50000.0
Priya	80000.0
Raj	95000.0
Ravi	48000.0
Rohit	36000.0
Simran	90000.0
Sneha	55000.0
Tanya	58000.0
Vikram	65000.0

```
#Find mean of salary.(Avg)

a = df.agg({'Salary':'mean'})
a
```

	0
Salary	58800.0

dtype: float64

```
#Use print for the same.

#(The table doesn't appear using print)

print(a)
```