Ex No : 1
23-07-2024

IMPORTING PANDAS LIBRARY AND EXPLORING THE FUNCTIONS IN IT

Aim:

To import the pandas library and exploring the functions in it for data analysis in Google Colab.

Given Dataset: Employee dataset

data = {'EmployeeID': [101, 102, 103, 104, 105,106,107,108,109,110],

'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve', ‘Cathey’, ‘Darth’, ‘John’, ‘Peter’, ’Alex’],

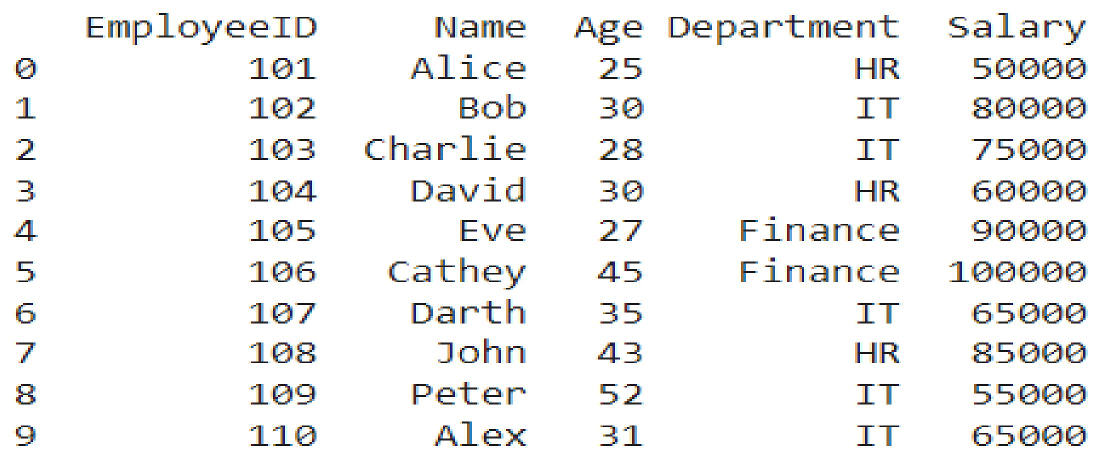
'Age': [25, 30, 28, 30, 27, 45, 35, 43, 52, 31],

'Department': ['HR', 'IT', 'IT', 'HR', 'Finance', ‘Finance', ’IT’, 'HR', ‘IT’, ‘IT’],

'Salary': [50000, 80000, 75000, 60000, 90000,100000,65000,85000, 55000, 65000]}

DataFrame:

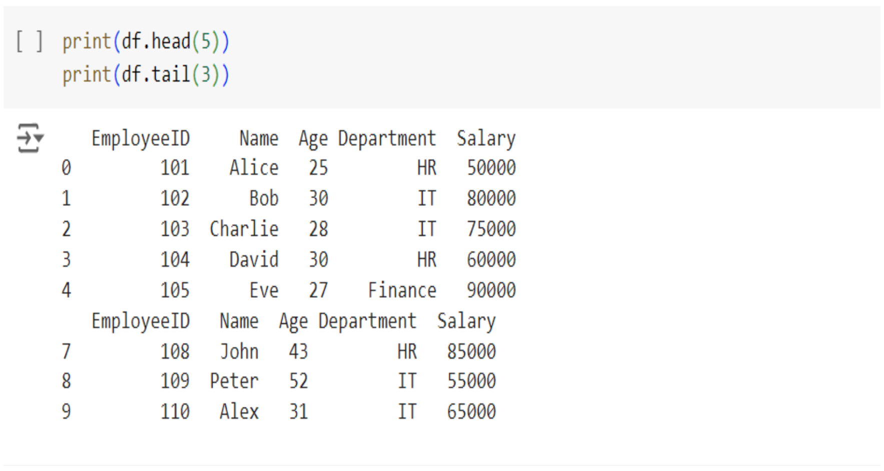
```
import pandas as pd
data = { 'EmployeeID': [101, 102, 103, 104, 105,106,107,108,109,110],
 'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve', 'Cathey', 'Darth', 'John', 'Peter', 'Alex'],
  'Age': [25, 30, 28, 30, 27, 45, 35, 43, 52, 31],
  'Department': ['HR', 'IT', 'IT', 'HR', 'Finance', 'Finance', 'IT', 'HR', 'IT', 'IT'],
  'Salary': [50000, 80000, 75000, 60000, 90000,100000,65000,85000, 55000, 65000]}
df = pd.DataFrame(data)
print(df)
```



```
   EmployeeID     Name  Age Department  Salary
0         101    Alice   25         HR   50000
1         102      Bob   30         IT   80000
2         103  Charlie   28         IT   75000
3         104    David   30         HR   60000
4         105      Eve   27    Finance   90000
5         106   Cathey   45    Finance  100000
6         107    Darth   35         IT   65000
7         108     John   43         HR   85000
8         109    Peter   52         IT   55000
9         110     Alex   31         IT   65000
```

Questions:

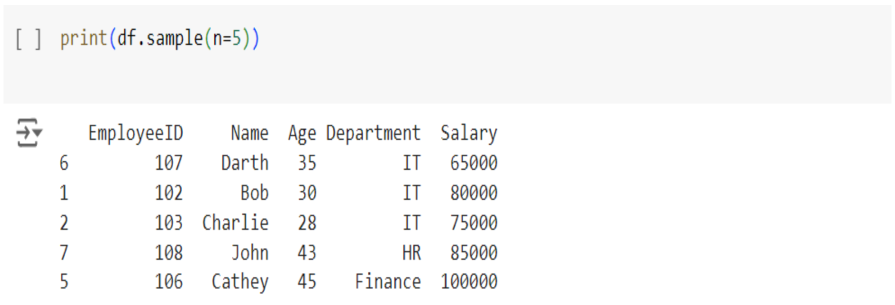
1.How can you display the first 5 rows and last 3 rows of employee dataset?



```
[ ] print(df.head(5))
    print(df.tail(3))
```

```
   EmployeeID     Name  Age Department  Salary
0         101    Alice   25         HR   50000
1         102      Bob   30         IT   80000
2         103  Charlie   28         IT   75000
3         104    David   30         HR   60000
4         105      Eve   27    Finance   90000
   EmployeeID   Name  Age Department  Salary
7         108   John   43         HR   85000
8         109  Peter   52         IT   55000
9         110   Alex   31         IT   65000
```

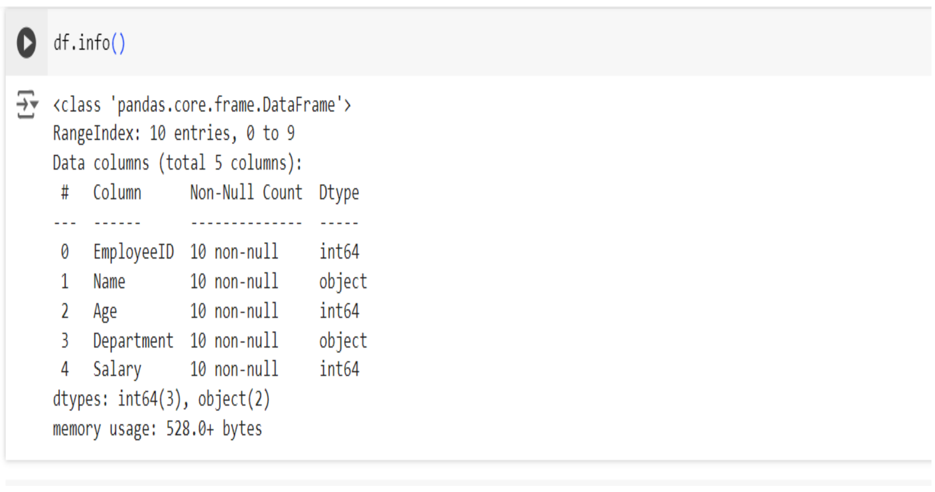
2. Retrieve a random sample of 5 rows from the employee dataset.



```
[ ] print(df.sample(n=5))
```

```
   EmployeeID     Name  Age Department  Salary
6         107    Darth   35         IT   65000
1         102      Bob   30         IT   80000
2         103  Charlie   28         IT   75000
7         108     John   43         HR   85000
5         106   Cathey   45    Finance  100000
```

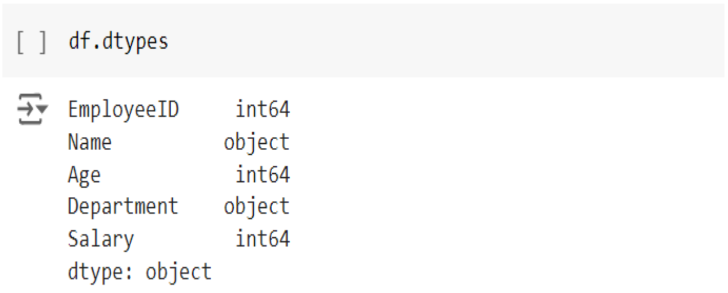
3. How can you get a concise summary of employee dataset including data typesand non-null values?



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   EmployeeID  10 non-null     int64
 1   Name        10 non-null     object
 2   Age         10 non-null     int64
 3   Department  10 non-null     object
 4   Salary      10 non-null     int64
dtypes: int64(3), object(2)
memory usage: 528.0+ bytes
```

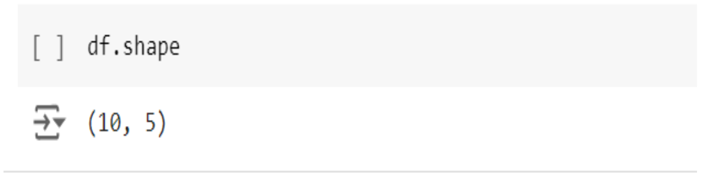
4. **Display the data types of each column in employee dataset?**



```
[ ] df.dtypes
```

```
EmployeeID     int64
Name          object
Age            int64
Department    object
Salary         int64
dtype: object
```

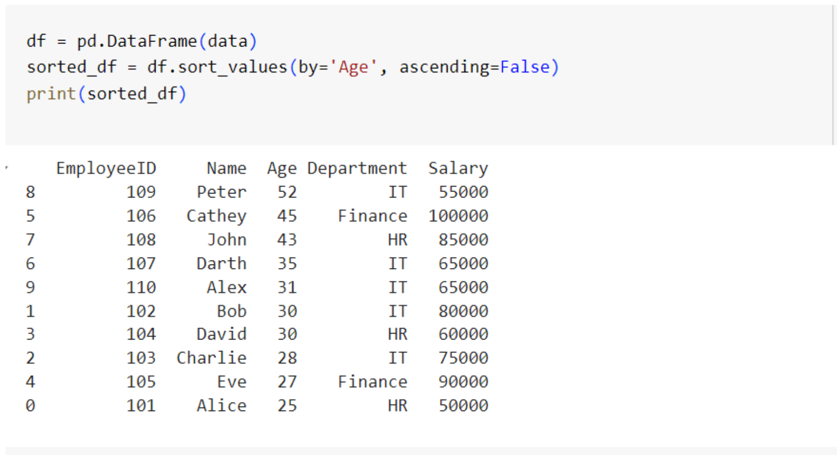
5. **Show the number of rows and columns in the dataset.**



```
[ ] df.shape
```

```
(10, 5)
```

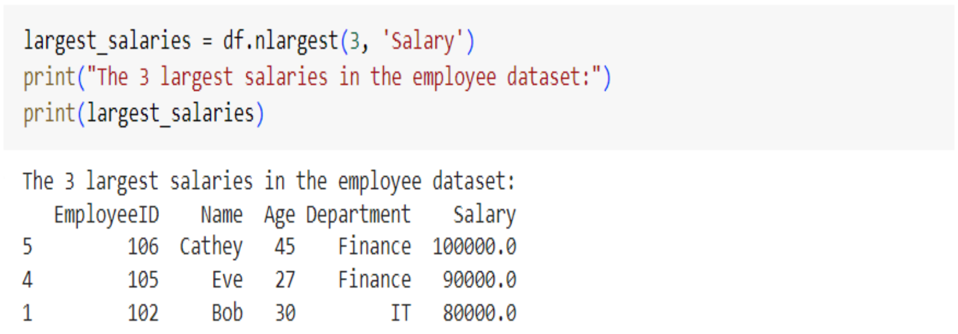
6. **How can you sort your employee dataset by 'Age' in descending order?**



```
df = pd.DataFrame(data)
sorted_df = df.sort_values(by='Age', ascending=False)
print(sorted_df)
```

```
   EmployeeID     Name  Age Department  Salary
8         109    Peter   52         IT   55000
5         106   Cathey   45    Finance  100000
7         108     John   43         HR   85000
6         107    Darth   35         IT   65000
9         110     Alex   31         IT   65000
1         102      Bob   30         IT   80000
3         104    David   30         HR   60000
2         103  Charlie   28         IT   75000
4         105      Eve   27    Finance   90000
0         101    Alice   25         HR   50000
```

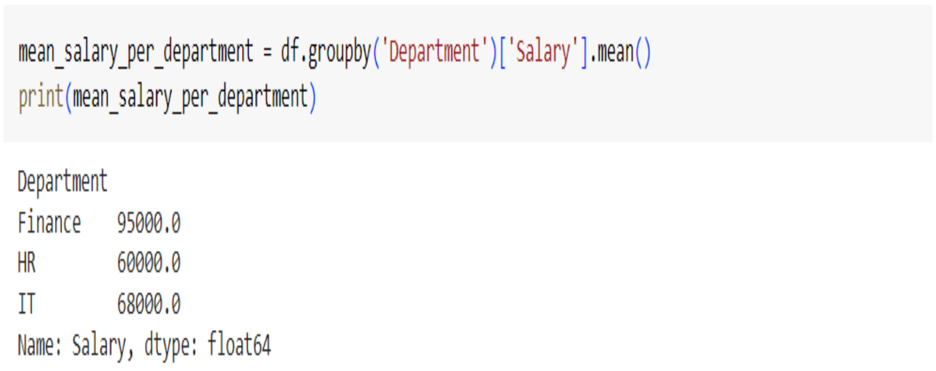
7. **Show the 3 largest salaries in employee dataset.**



```
largest_salaries = df.nlargest(3, 'Salary')
print("The 3 largest salaries in the employee dataset:")
print(largest_salaries)
```

```
The 3 largest salaries in the employee dataset:
   EmployeeID    Name  Age Department    Salary
5         106  Cathey   45    Finance  100000.0
4         105     Eve   27    Finance   90000.0
1         102     Bob   30         IT   80000.0
```

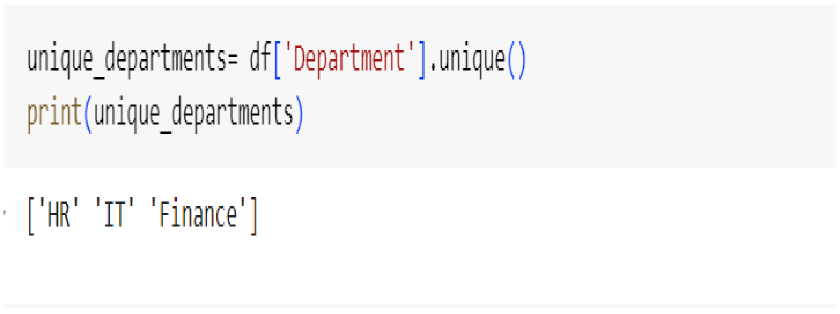
8. Calculate the mean salary of employees in each department.



```
mean_salary_per_department = df.groupby('Department')['Salary'].mean()
print(mean_salary_per_department)
```

```
Department
Finance    95000.0
HR         60000.0
IT         68000.0
Name: Salary, dtype: float64
```

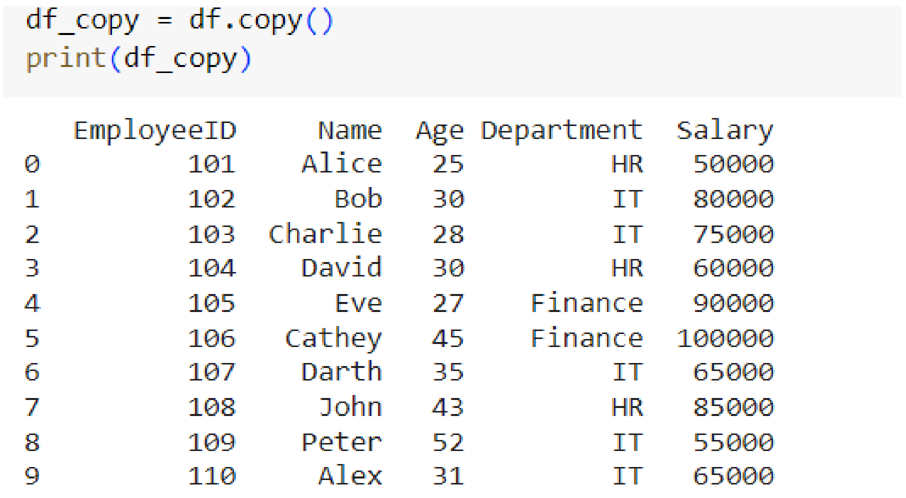
9. How many unique departments are there in employee dataset?



```
unique_departments= df['Department'].unique()
print(unique_departments)
```

```
['HR' 'IT' 'Finance']
```

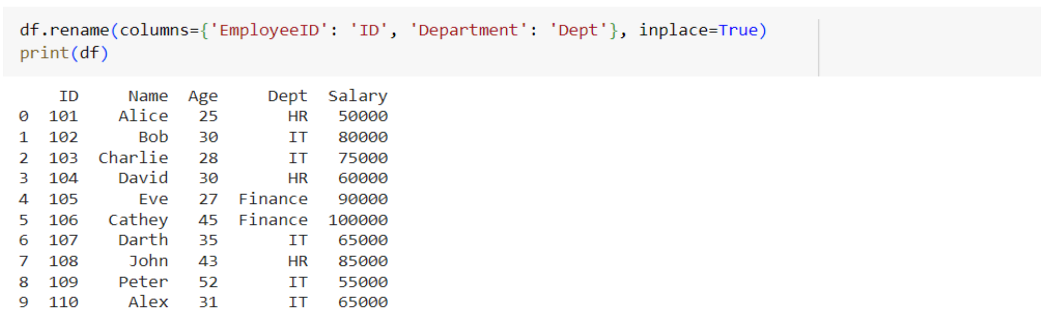
10. Create a copy of your employee dataset.



```
df_copy = df.copy()
print(df_copy)
```

```
   EmployeeID     Name  Age Department  Salary
0         101    Alice   25         HR   50000
1         102      Bob   30         IT   80000
2         103  Charlie   28         IT   75000
3         104    David   30         HR   60000
4         105      Eve   27    Finance   90000
5         106   Cathey   45    Finance  100000
6         107    Darth   35         IT   65000
7         108     John   43         HR   85000
8         109    Peter   52         IT   55000
9         110     Alex   31         IT   65000
```

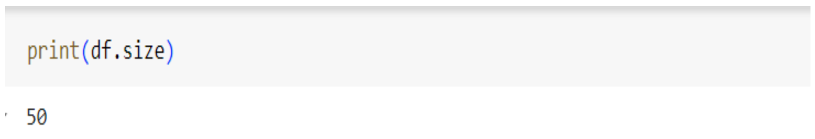
11. Rename the column ‘EmployeeID' as'ID', 'Department' as'Dept ' in employee dataset



```
df.rename(columns={'EmployeeID': 'ID', 'Department': 'Dept'}, inplace=True)
print(df)
```

```
    ID     Name  Age     Dept  Salary
0  101    Alice   25       HR   50000
1  102      Bob   30       IT   80000
2  103  Charlie   28       IT   75000
3  104    David   30       HR   60000
4  105      Eve   27  Finance   90000
5  106   Cathey   45  Finance  100000
6  107    Darth   35       IT   65000
7  108     John   43       HR   85000
8  109    Peter   52       IT   55000
9  110     Alex   31       IT   65000
```

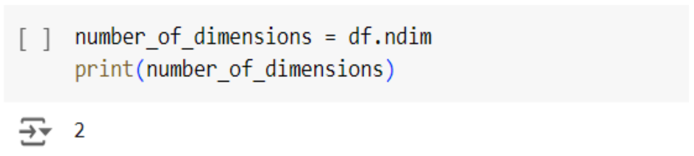
12. Show the total number of elements in the DataFrame



```
print(df.size)
```

```
50
```

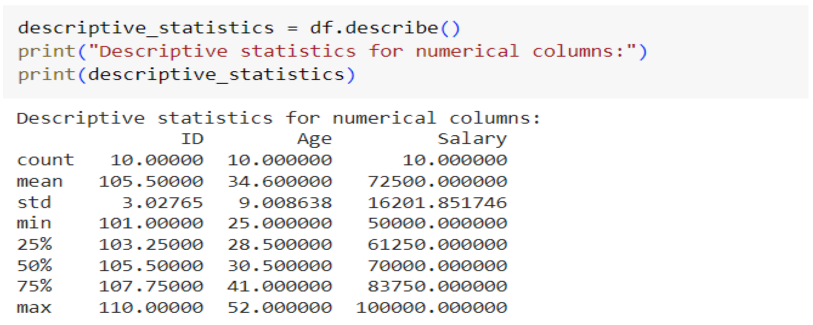
13. Display the number of dimensions (axes) of the DataFrame.



```
[ ]  number_of_dimensions = df.ndim
     print(number_of_dimensions)
```

```
2
```

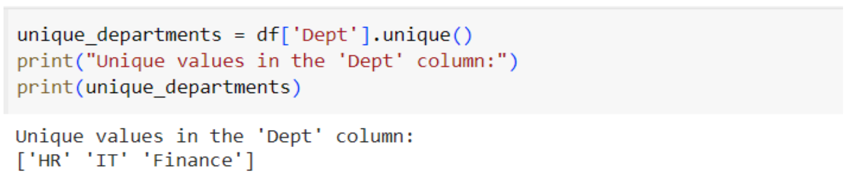
14. Generate descriptive statistics for numerical columns.



```
descriptive_statistics = df.describe()
print("Descriptive statistics for numerical columns:")
print(descriptive_statistics)
```

```
Descriptive statistics for numerical columns:
              ID        Age         Salary
count   10.00000  10.000000      10.000000
mean   105.50000  34.600000   72500.000000
std      3.02765   9.008638   16201.851746
min    101.00000  25.000000   50000.000000
25%    103.25000  28.500000   61250.000000
50%    105.50000  30.500000   70000.000000
75%    107.75000  41.000000   83750.000000
max    110.00000  52.000000  100000.000000
```

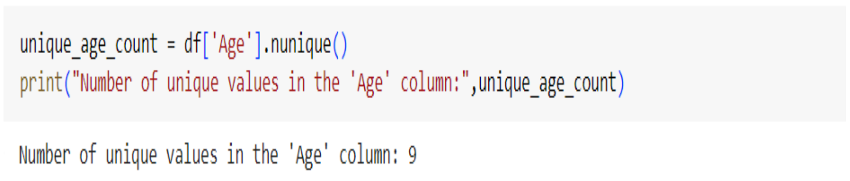
15. Display unique values in a ‘Department’ column.



```
unique_departments = df['Dept'].unique()
print("Unique values in the 'Dept' column:")
print(unique_departments)
```

```
Unique values in the 'Dept' column:
['HR' 'IT' 'Finance']
```

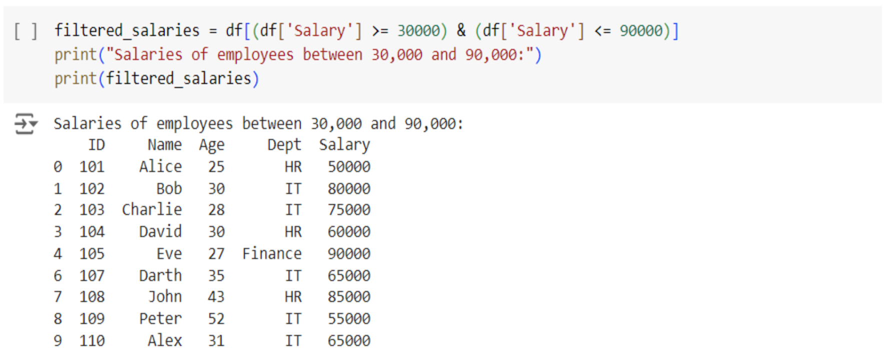
16. Count the number of unique values in an ‘Age’ column.



```
unique_age_count = df['Age'].nunique()
print("Number of unique values in the 'Age' column:",unique_age_count)
```

```
Number of unique values in the 'Age' column: 9
```

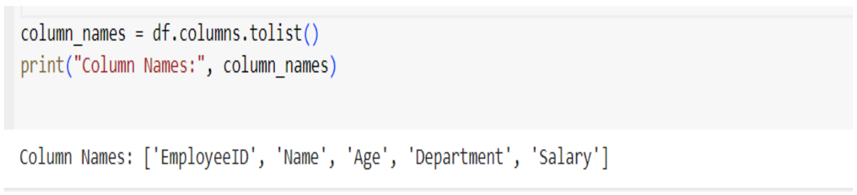
17. Display the salaries of employees between 30,000 and 90,000.



```
[ ] filtered_salaries = df[(df['Salary'] >= 30000) & (df['Salary'] <= 90000)]
    print("Salaries of employees between 30,000 and 90,000:")
    print(filtered_salaries)
```

```
Salaries of employees between 30,000 and 90,000:
    ID     Name  Age     Dept  Salary
0  101    Alice   25       HR   50000
1  102      Bob   30       IT   80000
2  103  Charlie   28       IT   75000
3  104    David   30       HR   60000
4  105      Eve   27  Finance   90000
6  107    Darth   35       IT   65000
7  108     John   43       HR   85000
8  109    Peter   52       IT   55000
9  110     Alex   31       IT   65000
```

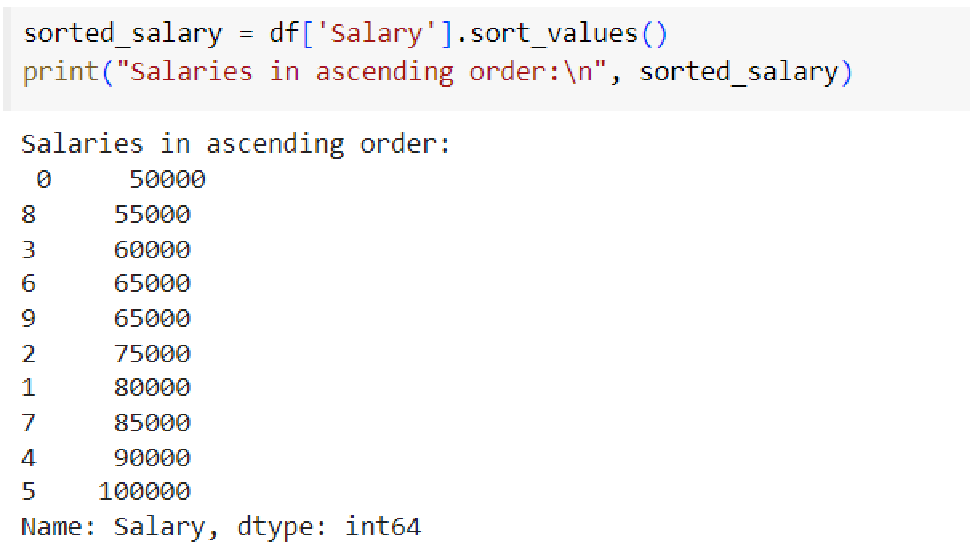
18. Get the list of column names in the DataFrame.



```
column_names = df.columns.tolist()
print("Column Names:", column_names)
```

```
Column Names: ['EmployeeID', 'Name', 'Age', 'Department', 'Salary']
```

19. Display the salary details in ascending order.



```
sorted_salary = df['Salary'].sort_values()
print("Salaries in ascending order:\n", sorted_salary)
```

```
Salaries in ascending order:
 0     50000
8     55000
3     60000
6     65000
9     65000
2     75000
1     80000
7     85000
4     90000
5    100000
Name: Salary, dtype: int64
```

20. Count occurrences of unique values in a ‘Department’.

```
department_counts = df['Department'].value_counts()
print("Department counts:\n", department_counts)
```

```
Department counts:
 Department
IT          5
HR          3
Finance     2
Name: count, dtype: int64
```

21. Display the top 5 rows with the largest values in a ‘Salary’ column.

```
top_5_salaries = df.nlargest(5, 'Salary')
print("Top 5 salaries:\n", top_5_salaries)
```

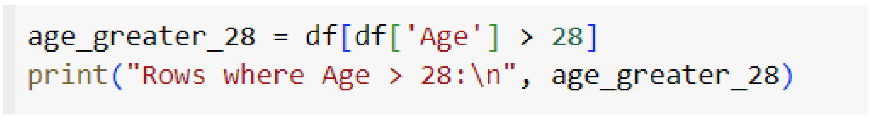
```
Top 5 salaries:
    EmployeeID     Name  Age Department  Salary
5         106   Cathey   45    Finance  100000
4         105      Eve   27    Finance   90000
7         108     John   43         HR   85000
1         102      Bob   30         IT   80000
2         103  Charlie   28         IT   75000
```

22. Create a deep copy of the DataFrame.

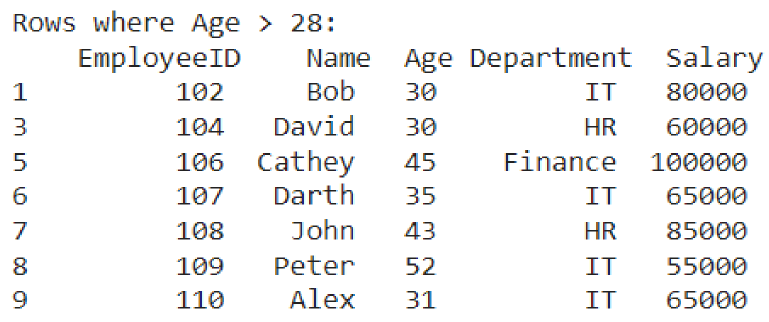
```
df_copy = df.copy(deep=True)
print(df_copy)
```

```
   EmployeeID     Name  Age Department  Salary
0         101    Alice   25         HR   50000
1         102      Bob   30         IT   80000
2         103  Charlie   28         IT   75000
3         104    David   30         HR   60000
4         105      Eve   27    Finance   90000
5         106   Cathey   45    Finance  100000
6         107    Darth   35         IT   65000
7         108     John   43         HR   85000
8         109    Peter   52         IT   55000
9         110     Alex   31         IT   65000
```

23. **How do you extract rows from a DataFrame where the values in the 'Age' column are greater than 28?**

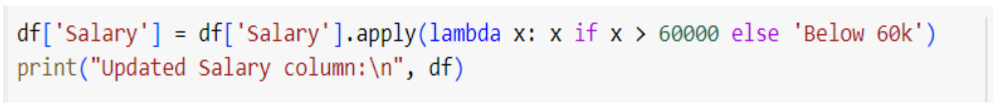


```
age_greater_28 = df[df['Age'] > 28]
print("Rows where Age > 28:\n", age_greater_28)
```

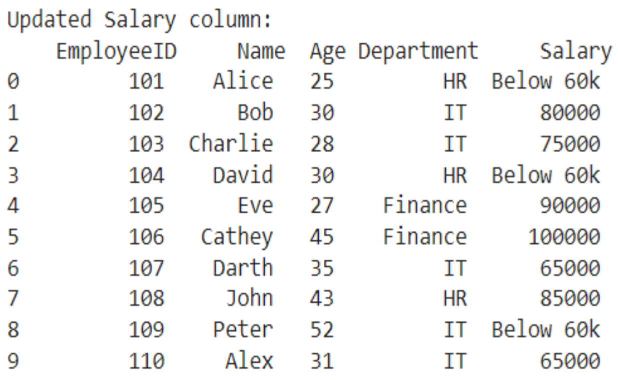


```
Rows where Age > 28:
    EmployeeID    Name  Age Department  Salary
1          102     Bob   30         IT   80000
3          104   David   30         HR   60000
5          106  Cathey   45    Finance  100000
6          107   Darth   35         IT   65000
7          108    John   43         HR   85000
8          109   Peter   52         IT   55000
9          110    Alex   31         IT   65000
```

24. **How would you update values in a DataFrame based on a condition, replacing values with 'Below 60k' where the condition ('Salary' > 60000) is not true?**

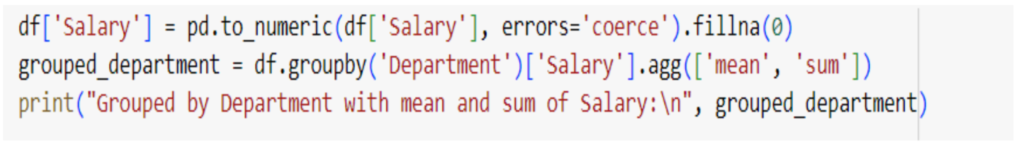


```
df['Salary'] = df['Salary'].apply(lambda x: x if x > 60000 else 'Below 60k')
print("Updated Salary column:\n", df)
```

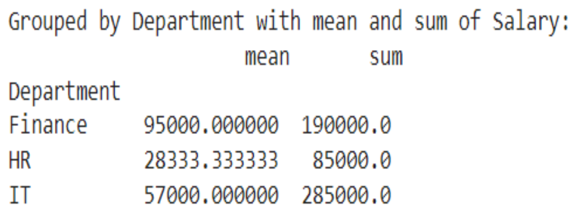


```
Updated Salary column:
    EmployeeID     Name  Age Department     Salary
0          101    Alice   25         HR  Below 60k
1          102      Bob   30         IT      80000
2          103  Charlie   28         IT      75000
3          104    David   30         HR  Below 60k
4          105      Eve   27    Finance      90000
5          106   Cathey   45    Finance     100000
6          107    Darth   35         IT      65000
7          108     John   43         HR      85000
8          109    Peter   52         IT  Below 60k
9          110     Alex   31         IT      65000
```

25. **Display the dataframe using Group by 'Department' and calculate mean, sum of values for salary.**

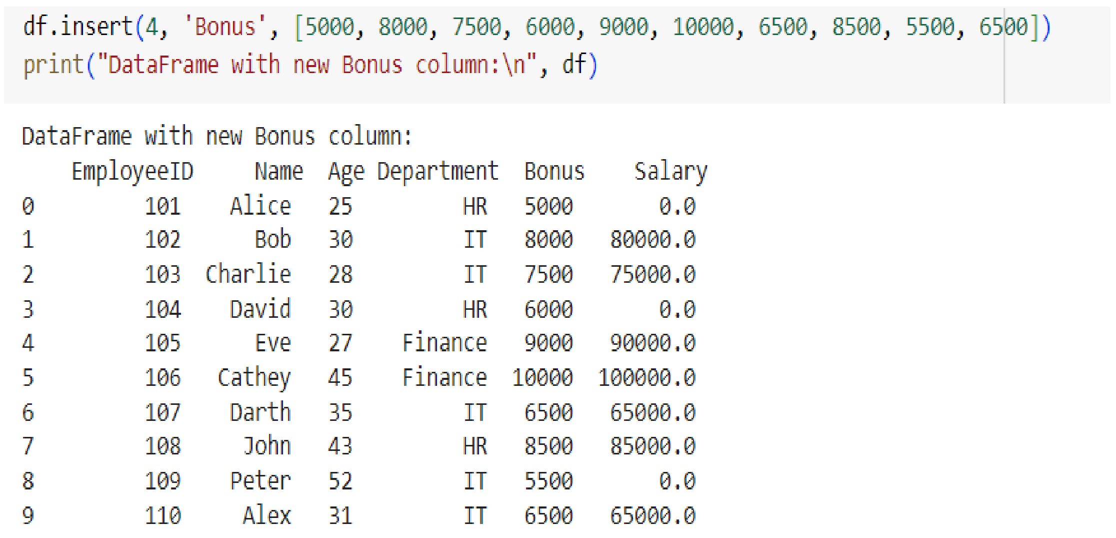


```
df['Salary'] = pd.to_numeric(df['Salary'], errors='coerce').fillna(0)
grouped_department = df.groupby('Department')['Salary'].agg(['mean', 'sum'])
print("Grouped by Department with mean and sum of Salary:\n", grouped_department)
```



```
Grouped by Department with mean and sum of Salary:
                    mean       sum
Department
Finance     95000.000000  190000.0
HR          28333.333333   85000.0
IT          57000.000000  285000.0
```

26. Insert a new column in location 4 into the DataFrame.



```python
df.insert(4, 'Bonus', [5000, 8000, 7500, 6000, 9000, 10000, 6500, 8500, 5500, 6500])
print("DataFrame with new Bonus column:\n", df)
```

```
DataFrame with new Bonus column:
    EmployeeID     Name  Age Department  Bonus    Salary
0         101    Alice   25         HR   5000       0.0
1         102      Bob   30         IT   8000   80000.0
2         103  Charlie   28         IT   7500   75000.0
3         104    David   30         HR   6000       0.0
4         105      Eve   27    Finance   9000   90000.0
5         106   Cathey   45    Finance  10000  100000.0
6         107    Darth   35         IT   6500   65000.0
7         108     John   43         HR   8500   85000.0
8         109    Peter   52         IT   5500       0.0
9         110     Alex   31         IT   6500   65000.0
```

Rubrics:

Problem Understanding (10)	Implementation (20)	Viva (10)	Time Management (10)	Total (50)

Result:

Thus the implementation of importing the pandas library and exploring the functions in it for data analysis in Google Colab was successfully executed and the output was verified.