ASSIGNMENT-5

```
# Creating a new dataset
data = {
    "Employee_ID": [101, 102, 103, 104, 105, 106],
    "Name": ["Rajesh", "Meena", "Suresh", "Anita", "Vijay", "Neeta"],
    "Department": ["HR", "IT", "Finance", "IT", "Finance", "HR"],
    "Age": [29, 35, 45, 32, 50, 28],
    "Salary": [70000, 85000, 95000, 64000, 120000, 72000],
    "City": ["Delhi", "Mumbai", "Bangalore", "Chennai", "Delhi", "Mumbai"]
}
df = pd.DataFrame(data)
print(df)
```

Exercise 1: Rename Columns

Rename the "Salary" column to "Annual Salary" and "City" to "Location".

Print the updated DataFrame.

Program:

```
df.rename(columns={"Salary": "Annual Salary", "City": "Location"})
print(df)
```

Exercise 2: Drop Columns

Drop the "Location" column from the DataFrame.

Print the DataFrame after dropping the column.

Program:

```
df.drop(columns=["Location"])
print(df)
```

Exercise 3: Drop Rows

Drop the row where "Name" is "Suresh".

Print the updated DataFrame.

Program:

```
df = df[df["Name"] != "Suresh"]
print(df)
```

Exercise 4: Handle Missing Data

Assign None to the "Salary" of "Meena".

Fill the missing "Salary" value with the mean salary of the existing employees.

Print the cleaned DataFrame.

Program:

```
df.loc[df["Name"] == "Meena", "Annual Salary"] = None
df["Annual Salary"].fillna(df["Annual Salary"].mean())
print(df)
```

Exercise 5: Create Conditional Columns

Create a new column "Seniority" that assigns "Senior" to employees aged 40 or above and "Junior" to employees younger than 40.

Print the updated DataFrame.

Program:

```
df["Seniority"] = df["Age"].apply(lambda x: "Senior" if x >= 40 else "Junior")
print(df)
```

Exercise 6: Grouping and Aggregation

Group the DataFrame by "Department" and calculate the average salary in each department.

Print the grouped DataFrame.

Program:

```
grouped_df = df.groupby("Department")["Annual Salary"].mean().reset_index()
```

```
print(grouped_df)
```