# BIG DATA ANALYSIS USING DASK

## Project Title

Big Data Analysis on Social Network Ads Dataset using Dask

## Objective

To perform scalable analysis on a dataset using Dask and extract insights such as:

- Total rows and columns
- Missing values
- Purchased distribution
- Gender distribution
- Group-wise average salary and age
- Top purchased users

## Tool Used

**Dask DataFrame**

- Dask supports parallel processing using partitions.
- It is suitable for scalable big data processing.

In [8]:

```
import dask.dataframe as dd

# Load dataset
df = dd.read_csv("social.csv")   # keep your file name correct
df.compute()
```

Out[8]:

| | User ID | Gender | Age | EstimatedSalary | Purchased |
|---|---|---|---|---|---|
| **0** | 15624510 | Male | 19 | 19000 | 0 |
| **1** | 15810944 | Male | 35 | 20000 | 0 |
| **2** | 15668575 | Female | 26 | 43000 | 0 |
| **3** | 15603246 | Female | 27 | 57000 | 0 |
| **4** | 15804002 | Male | 19 | 76000 | 0 |
| **5** | 15728773 | Male | 27 | 58000 | 0 |
| **6** | 15598044 | Female | 27 | 84000 | 0 |
| **7** | 15694829 | Female | 32 | 150000 | 1 |
| **8** | 15600575 | Male | 25 | 33000 | 0 |
| **9** | 15727311 | Female | 35 | 65000 | 0 |
| **10** | 15570769 | Female | 26 | 80000 | 0 |
| **11** | 15606274 | Female | 26 | 52000 | 0 |
| **12** | 15746139 | Male | 20 | 86000 | 0 |
| **13** | 15704987 | Male | 32 | 18000 | 0 |
| **14** | 15628972 | Male | 18 | 82000 | 0 |
| **15** | 15697686 | Male | 29 | 80000 | 0 |
| **16** | 15733883 | Male | 47 | 25000 | 1 |
| **17** | 15617482 | Male | 45 | 26000 | 1 |
| **18** | 15704583 | Male | 46 | 28000 | 1 |
| **19** | 15621083 | Female | 48 | 29000 | 1 |
| **20** | 15649487 | Male | 45 | 22000 | 1 |
| **21** | 15736760 | Female | 47 | 49000 | 1 |
| **22** | 15714658 | Male | 48 | 41000 | 1 |
| **23** | 15599081 | Female | 45 | 22000 | 1 |
| **24** | 15705113 | Male | 46 | 23000 | 1 |
| **25** | 15631159 | Male | 47 | 20000 | 1 |
| **26** | 15792818 | Male | 49 | 28000 | 1 |
| **27** | 15633531 | Female | 47 | 30000 | 1 |
| **28** | 15744529 | Male | 29 | 43000 | 0 |
| **29** | 15669656 | Male | 31 | 18000 | 0 |
| **30** | 15581198 | Male | 31 | 74000 | 0 |
| **31** | 15729054 | Female | 27 | 137000 | 1 |
| **32** | 15573452 | Female | 21 | 16000 | 0 |
| **33** | 15776733 | Female | 28 | 44000 | 0 |
| **34** | 15724858 | Male | 27 | 90000 | 0 |
| **35** | 15713144 | Male | 35 | 27000 | 0 |

| | User ID | Gender | Age | EstimatedSalary | Purchased |
|---|---|---|---|---|---|
| **36** | 15690188 | Female | 33 | 28000 | 0 |
| **37** | 15689425 | Male | 30 | 49000 | 0 |
| **38** | 15671766 | Female | 26 | 72000 | 0 |
| **39** | 15782806 | Female | 27 | 31000 | 0 |
| **40** | 15764419 | Female | 27 | 17000 | 0 |
| **41** | 15591915 | Female | 33 | 51000 | 0 |
| **42** | 15772798 | Male | 35 | 108000 | 0 |
| **43** | 15792008 | Male | 30 | 15000 | 0 |
| **44** | 15715541 | Female | 28 | 84000 | 0 |
| **45** | 15639277 | Male | 23 | 20000 | 0 |
| **46** | 15798850 | Male | 25 | 79000 | 0 |
| **47** | 15776348 | Female | 27 | 54000 | 0 |
| **48** | 15727696 | Male | 30 | 135000 | 1 |
| **49** | 15793813 | Female | 31 | 89000 | 0 |
| **50** | 15694395 | Female | 24 | 32000 | 0 |
| **51** | 15764195 | Female | 18 | 44000 | 0 |

In [13]:
```
print("Total Rows:", df.shape[0].compute())
print("Total Columns:", len(df.columns))
print("Partitions (Parallelism):", df.npartitions)

print(df.dtypes)
```

```
Total Rows: 52
Total Columns: 5
Partitions (Parallelism): 1
User ID             int64
Gender             string
Age                 int64
EstimatedSalary     int64
Purchased           int64
dtype: object
```

# Missing Values Analysis

We check how many missing values are present in each column.

In [10]:
```
missing = df.isnull().sum().compute()
print("Missing Values:\n", missing)
```

```
Missing Values:
 User ID           0
Gender            0
Age               0
EstimatedSalary   0
Purchased         0
dtype: int64
```

# Purchased Distribution

This shows how many users purchased (1) and not purchased (0).

In [12]:
```
df[df["Purchased"] == 1].compute()
```

Out[12]:

| | User ID | Gender | Age | EstimatedSalary | Purchased |
|---|---|---|---|---|---|
| **7** | 15694829 | Female | 32 | 150000 | 1 |
| **16** | 15733883 | Male | 47 | 25000 | 1 |
| **17** | 15617482 | Male | 45 | 26000 | 1 |
| **18** | 15704583 | Male | 46 | 28000 | 1 |
| **19** | 15621083 | Female | 48 | 29000 | 1 |
| **20** | 15649487 | Male | 45 | 22000 | 1 |
| **21** | 15736760 | Female | 47 | 49000 | 1 |
| **22** | 15714658 | Male | 48 | 41000 | 1 |
| **23** | 15599081 | Female | 45 | 22000 | 1 |
| **24** | 15705113 | Male | 46 | 23000 | 1 |
| **25** | 15631159 | Male | 47 | 20000 | 1 |
| **26** | 15792818 | Male | 49 | 28000 | 1 |
| **27** | 15633531 | Female | 47 | 30000 | 1 |
| **31** | 15729054 | Female | 27 | 137000 | 1 |
| **48** | 15727696 | Male | 30 | 135000 | 1 |

In [14]:
```
df[df["Purchased"] == 0].compute()
```

Out[14]:

| | User ID | Gender | Age | EstimatedSalary | Purchased |
|---|---|---|---|---|---|
| **0** | 15624510 | Male | 19 | 19000 | 0 |
| **1** | 15810944 | Male | 35 | 20000 | 0 |
| **2** | 15668575 | Female | 26 | 43000 | 0 |
| **3** | 15603246 | Female | 27 | 57000 | 0 |
| **4** | 15804002 | Male | 19 | 76000 | 0 |
| **5** | 15728773 | Male | 27 | 58000 | 0 |
| **6** | 15598044 | Female | 27 | 84000 | 0 |
| **8** | 15600575 | Male | 25 | 33000 | 0 |
| **9** | 15727311 | Female | 35 | 65000 | 0 |
| **10** | 15570769 | Female | 26 | 80000 | 0 |
| **11** | 15606274 | Female | 26 | 52000 | 0 |
| **12** | 15746139 | Male | 20 | 86000 | 0 |
| **13** | 15704987 | Male | 32 | 18000 | 0 |
| **14** | 15628972 | Male | 18 | 82000 | 0 |
| **15** | 15697686 | Male | 29 | 80000 | 0 |
| **28** | 15744529 | Male | 29 | 43000 | 0 |
| **29** | 15669656 | Male | 31 | 18000 | 0 |
| **30** | 15581198 | Male | 31 | 74000 | 0 |
| **32** | 15573452 | Female | 21 | 16000 | 0 |
| **33** | 15776733 | Female | 28 | 44000 | 0 |
| **34** | 15724858 | Male | 27 | 90000 | 0 |
| **35** | 15713144 | Male | 35 | 27000 | 0 |
| **36** | 15690188 | Female | 33 | 28000 | 0 |
| **37** | 15689425 | Male | 30 | 49000 | 0 |
| **38** | 15671766 | Female | 26 | 72000 | 0 |
| **39** | 15782806 | Female | 27 | 31000 | 0 |
| **40** | 15764419 | Female | 27 | 17000 | 0 |
| **41** | 15591915 | Female | 33 | 51000 | 0 |
| **42** | 15772798 | Male | 35 | 108000 | 0 |
| **43** | 15792008 | Male | 30 | 15000 | 0 |
| **44** | 15715541 | Female | 28 | 84000 | 0 |
| **45** | 15639277 | Male | 23 | 20000 | 0 |
| **46** | 15798850 | Male | 25 | 79000 | 0 |
| **47** | 15776348 | Female | 27 | 54000 | 0 |
| **49** | 15793813 | Female | 31 | 89000 | 0 |
| **50** | 15694395 | Female | 24 | 32000 | 0 |

| | User ID | Gender | Age | EstimatedSalary | Purchased |
|---|---|---|---|---|---|
| **51** | 15764195 | Female | 18 | 44000 | 0 |

# Gender Distribution

This shows Male vs Female count.

In [15]:
```
gender_count = df["Gender"].value_counts().compute()
print("Gender Distribution:\n", gender_count)
```

```
Gender Distribution:
 Gender
Female    24
Male      28
Name: count, dtype: int64[pyarrow]
```

# Group-wise Analysis

We find average Age and Estimated Salary for Purchased=0 and Purchased=1.

In [16]:
```
grouped = df.groupby("Purchased")[["Age", "EstimatedSalary"]].mean().compute()
print("Average Age & Salary by Purchased:\n", grouped)
```

```
Average Age & Salary by Purchased:
                   Age  EstimatedSalary
Purchased
0           27.297297     52378.378378
1           43.266667     51000.000000
```

# Top Purchased Users

We extract top 5 users with highest salary among Purchased = 1.

In [17]:
```
top_salary = df[df["Purchased"] == 1][["User ID","Gender","Age","EstimatedSalary"]] \
    .nlargest(5, "EstimatedSalary") \
    .compute()

print("Top 5 highest salary purchased users:\n")
print(top_salary)
```

```
Top 5 highest salary purchased users:

     User ID  Gender  Age  EstimatedSalary
7   15694829  Female   32           150000
31  15729054  Female   27           137000
48  15727696    Male   30           135000
21  15736760  Female   47            49000
22  15714658    Male   48            41000
```

# Final Insights & Conclusion

- Majority of users did not purchase.
- Purchased users generally have higher salary and higher age.
- Dask processed the dataset efficiently using partitions, showing scalability for big data.

In [18]:
```
print("---------- FINAL INSIGHTS ----------")
print("1) Purchased Distribution:\n", purchase_count)
```

```python
print("\n2) Gender Distribution:\n", gender_count)
print("\n3) Avg Age & Salary by Purchased:\n", grouped)
print("\nScalability proof - Partitions:", df.npartitions)
```

```
---------- FINAL INSIGHTS ----------
1) Purchased Distribution:
 Purchased
0    37
1    15
Name: count, dtype: int64

2) Gender Distribution:
 Gender
Female    24
Male      28
Name: count, dtype: int64[pyarrow]

3) Avg Age & Salary by Purchased:
                  Age  EstimatedSalary
Purchased
0          27.297297     52378.378378
1          43.266667     51000.000000

Scalability proof - Partitions: 1
```