Income Analysis using Python Pandas

dataset = https://www.kaggle.com/datasets/wenruliu/adult-income-dataset?resource=download (https://www.kaggle.com/datasets/wenruliu/adult-income-dataset?resource=download)

In [ ]:

```
import pandas as pd
```

In [2]:

```
import matplotlib.pyplot as plt
```

In [8]:

```
import csv
```

In [23]:

```
import seaborn as sns
```

In [11]:

```
data=pd.read_csv(r'C:\Users\hp\Desktop\pandas projecrs\Adult Income Project\adult.csv')
```

In [12]:

```
data
```

Out[12]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship	
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	E
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	V
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	V
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	E
4	18	?	103497	Some-college	10	Never-married	?	Own-child	V
...	...	...	...	...	...	...	...	...	
48837	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	V
48838	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	V
48839	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarried	V
48840	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	V
48841	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	V

48842 rows × 15 columns

1. Display Top 10 rows of the DataSet .

In [13]:

```
data.head(10)
```

Out[13]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship	race
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	Black
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black
4	18	?	103497	Some-college	10	Never-married	?	Own-child	White
5	34	Private	198693	10th	6	Never-married	Other-service	Not-in-family	White
6	29	?	227026	HS-grad	9	Never-married	?	Unmarried	Black
7	63	Self-emp-not-inc	104626	Prof-school	15	Married-civ-spouse	Prof-specialty	Husband	White
8	24	Private	369667	Some-college	10	Never-married	Other-service	Unmarried	White
9	55	Private	104996	7th-8th	4	Married-civ-spouse	Craft-repair	Husband	White

2. Display last 10 rows of the Dataset .

In [14]:

```
data.tail(10)
```

Out[14]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship
48832	32	Private	34066	10th	6	Married-civ-spouse	Handlers-cleaners	Husband
48833	43	Private	84661	Assoc-voc	11	Married-civ-spouse	Sales	Husband
48834	32	Private	116138	Masters	14	Never-married	Tech-support	Not-in-family
48835	53	Private	321865	Masters	14	Married-civ-spouse	Exec-managerial	Husband
48836	22	Private	310152	Some-college	10	Never-married	Protective-serv	Not-in-family
48837	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife
48838	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband
48839	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarried
48840	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child
48841	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife

3. Find shape of the Dataset (Number of rows and number of columns)

In [15]:

```
data.shape
```

Out[15]:

```
(48842, 15)
```

In [16]:

```
print("Number of Rows :",data.shape[0])
print("Number of Columns :",data.shape[1])
```

```
Number of Rows : 48842
Number of Columns : 15
```

4. Getting Information About our Dataset like total number of Rows,total number of columns,datatypes of each column and memory requiement .

In [17]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48842 entries, 0 to 48841
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   age              48842 non-null  int64 
 1   workclass        48842 non-null  object
 2   fnlwgt           48842 non-null  int64 
 3   education        48842 non-null  object
 4   educational-num  48842 non-null  int64 
 5   marital-status   48842 non-null  object
 6   occupation       48842 non-null  object
 7   relationship     48842 non-null  object
 8   race             48842 non-null  object
 9   gender           48842 non-null  object
 10  capital-gain     48842 non-null  int64 
 11  capital-loss     48842 non-null  int64 
 12  hours-per-week   48842 non-null  int64 
 13  native-country   48842 non-null  object
 14  income           48842 non-null  object
dtypes: int64(6), object(9)
memory usage: 5.6+ MB
```

5. Fetch Random Samples from the Dataset(50 %).

In [19]:

```
data.sample(frac=0.50)
```

Out[19]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship
39411	48	Private	233802	Masters	14	Married-civ-spouse	Exec-managerial	Husband
1584	36	Private	209993	5th-6th	3	Married-civ-spouse	Priv-house-serv	Wife
31149	19	Private	184737	HS-grad	9	Never-married	Other-service	Own-child
39387	54	Self-emp-inc	162439	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband
28462	27	Private	134152	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband
...	...	...	...	...	...	...	...	...
38435	42	Private	336643	Assoc-voc	11	Separated	Prof-specialty	Unmarried
29805	40	Private	169885	HS-grad	9	Separated	Other-service	Not-in-family
2514	18	Private	183274	11th	7	Never-married	Other-service	Own-child
32806	44	Self-emp-inc	64632	Doctorate	16	Married-civ-spouse	Prof-specialty	Husband
8072	63	Self-emp-not-inc	246124	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband

24421 rows × 15 columns

6. Check Null Values in the Dataset.

In [21]:

```
data.isnull().sum()
```

Out[21]:

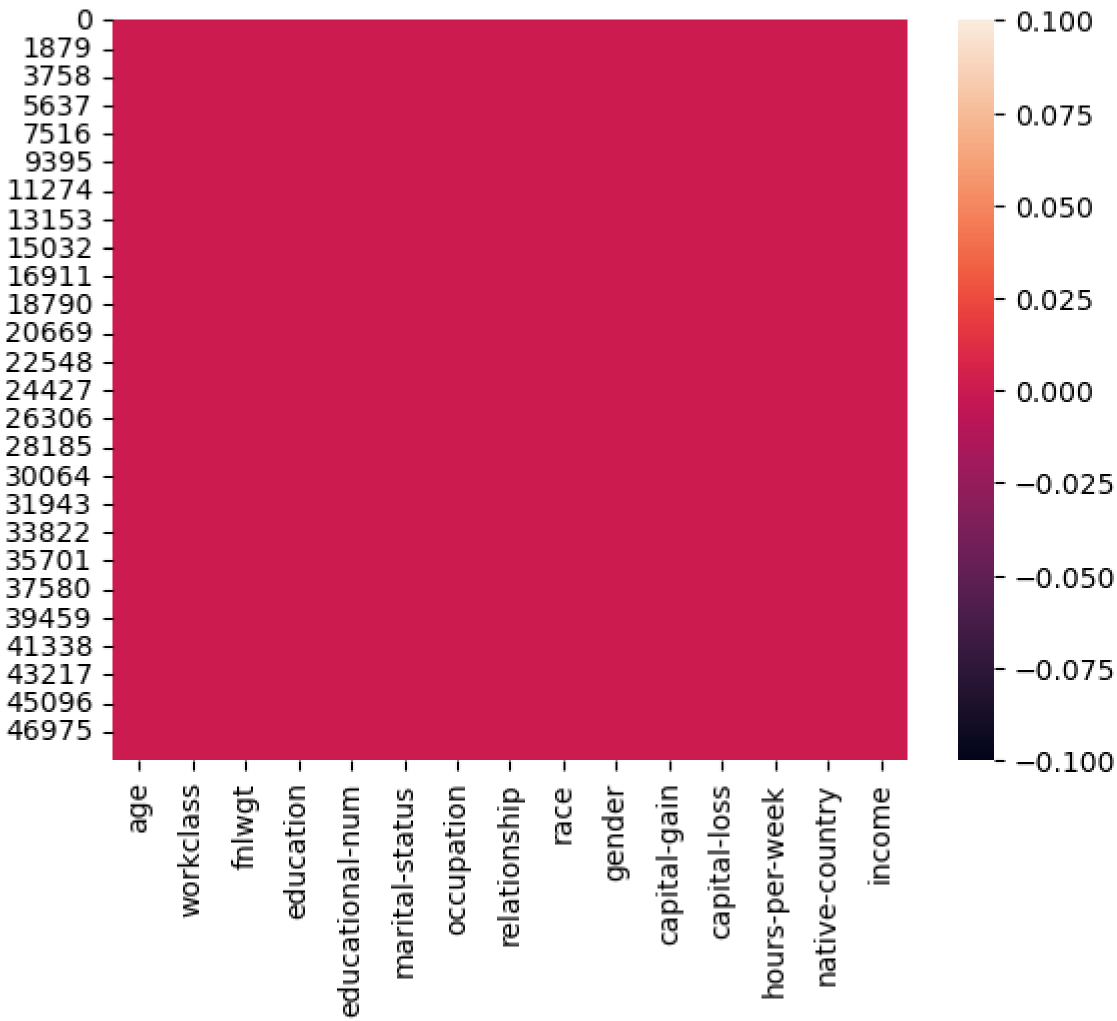
```
age                0
workclass          0
fnlwgt             0
education          0
educational-num    0
marital-status     0
occupation         0
relationship       0
race               0
gender             0
capital-gain       0
capital-loss       0
hours-per-week     0
native-country     0
income             0
dtype: int64
```

In [25]:

```
sns.heatmap(data.isnull())
```

Out[25]:

```
<Axes: >
```



7. Perform Data cleaning (Replace '?' with NaN)

In [26]:

```
data.tail(20)
```

Out[26]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship
48822	41	?	202822	HS-grad	9	Separated	?	Not-in-family
48823	72	?	129912	HS-grad	9	Married-civ-spouse	?	Husband
48824	45	Local-gov	119199	Assoc-acdm	12	Divorced	Prof-specialty	Unmarried
48825	31	Private	199655	Masters	14	Divorced	Other-service	Not-in-family
48826	39	Local-gov	111499	Assoc-acdm	12	Married-civ-spouse	Adm-clerical	Wife
48827	37	Private	198216	Assoc-acdm	12	Divorced	Tech-support	Not-in-family
48828	43	Private	260761	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband
48829	65	Self-emp-not-inc	99359	Prof-school	15	Never-married	Prof-specialty	Not-in-family
48830	43	State-gov	255835	Some-college	10	Divorced	Adm-clerical	Other-relative
48831	43	Self-emp-not-inc	27242	Some-college	10	Married-civ-spouse	Craft-repair	Husband
48832	32	Private	34066	10th	6	Married-civ-spouse	Handlers-cleaners	Husband
48833	43	Private	84661	Assoc-voc	11	Married-civ-spouse	Sales	Husband
48834	32	Private	116138	Masters	14	Never-married	Tech-support	Not-in-family
48835	53	Private	321865	Masters	14	Married-civ-spouse	Exec-managerial	Husband
48836	22	Private	310152	Some-college	10	Never-married	Protective-serv	Not-in-family
48837	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife
48838	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband
48839	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarried
48840	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship
48841	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife

In [27]:

```
data.isin(['?']).sum()
```

Out[27]:

```
age                  0
workclass         2799
fnlwgt               0
education            0
educational-num      0
marital-status       0
occupation        2809
relationship         0
race                 0
gender               0
capital-gain         0
capital-loss         0
hours-per-week       0
native-country     857
income               0
dtype: int64
```

In [28]:

```
import numpy as np
```

In [29]:

```
data.columns
```

Out[29]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'educational-num',
       'marital-status', 'occupation', 'relationship', 'race', 'gender',
       'capital-gain', 'capital-loss', 'hours-per-week', 'native-country',
       'income'],
      dtype='object')
```

In [31]:

```
data['workclass'] = data['workclass'].replace('?',np.nan)
data['occupation'] = data['occupation'].replace('?',np.nan)
data['native-country'] = data['native-country'].replace('?',np.nan)
```

In [32]:

```
data.isin(['?']).sum()
```

Out[32]:

```
age                0
workclass          0
fnlwgt             0
education          0
educational-num    0
marital-status     0
occupation         0
relationship       0
race               0
gender             0
capital-gain       0
capital-loss       0
hours-per-week     0
native-country     0
income             0
dtype: int64
```

In [33]:

```
data.isnull().sum()
```

Out[33]:

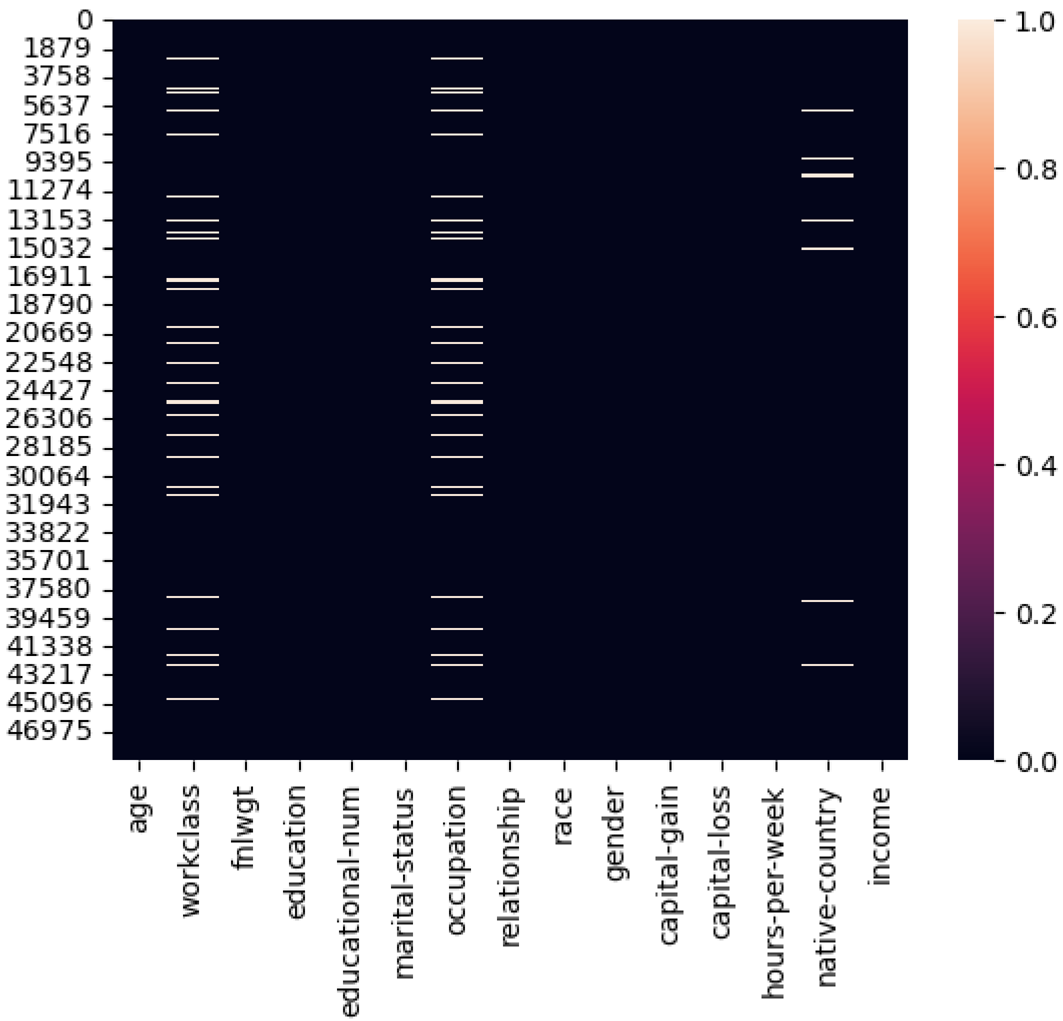
```
age                   0
workclass          2799
fnlwgt                0
education             0
educational-num       0
marital-status        0
occupation         2809
relationship          0
race                  0
gender                0
capital-gain          0
capital-loss          0
hours-per-week        0
native-country      857
income                0
dtype: int64
```

In [34]:

```
sns.heatmap(data.isnull())
```

Out[34]:

```
<Axes: >
```



8. Drop all the Missing Values .

In [35]:

```
per_missing = data.isnull().sum()*100 / len(data)
```

In [36]:

```
per_missing
```

Out[36]:

```
age                0.000000
workclass          5.730724
fnlwgt             0.000000
education          0.000000
educational-num    0.000000
marital-status     0.000000
occupation         5.751198
relationship       0.000000
race               0.000000
gender             0.000000
capital-gain       0.000000
capital-loss       0.000000
hours-per-week     0.000000
native-country     1.754637
income             0.000000
dtype: float64
```

In [37]:

```
data.dropna(how='any',inplace=True)
```

In [38]:

```
data.shape
```

Out[38]:

```
(45222, 15)
```

9. Check For Duplicate Data and Drop Them .

In [39]:

```
dup = data.duplicated().any()
```

In [40]:

```
print("Are there any duplicated values in data : ",dup)
```

```
Are there any duplicated values in data :  True
```

In [42]:

```
data=data.drop_duplicates()
```

In [43]:

```
data.shape
```

Out[43]:

```
(45175, 15)
```

10. Get Overall Statistics of the datset .

In [45]:

```
data.describe()
```

Out[45]:

	age	fnlwgt	educational-num	capital-gain	capital-loss	hours-per-week
count	45175.000000	4.517500e+04	45175.000000	45175.000000	45175.000000	45175.000000
mean	38.556170	1.897388e+05	10.119314	1102.576270	88.687593	40.942512
std	13.215349	1.056524e+05	2.551740	7510.249876	405.156611	12.007730
min	17.000000	1.349200e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.173925e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.783120e+05	10.000000	0.000000	0.000000	40.000000
75%	47.000000	2.379030e+05	13.000000	0.000000	0.000000	45.000000
max	90.000000	1.490400e+06	16.000000	99999.000000	4356.000000	99.000000

11. Drop the columns education-num,capital-gain and capital loss

In [46]:

```
data.columns
```

Out[46]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'educational-num',
       'marital-status', 'occupation', 'relationship', 'race', 'gender',
       'capital-gain', 'capital-loss', 'hours-per-week', 'native-country',
       'income'],
      dtype='object')
```

In [48]:

```
data = data.drop(['educational-num','capital-gain','capital-loss'],axis=1)
```

In [49]:

```
data.columns
```

Out[49]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'marital-status',
       'occupation', 'relationship', 'race', 'gender', 'hours-per-week',
       'native-country', 'income'],
      dtype='object')
```

Univariate Analysis

Univariate analysis. The prefix 'Uni' means one, meaning 'univariate analysis' is the analysis of one variable at a time. For numeric features, we want to know the range of values present and how often these values (or groups of values) occur

12 . What is the Distribution of Age Column ?

In [50]:

```
data.columns
```

Out[50]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'marital-status',
       'occupation', 'relationship', 'race', 'gender', 'hours-per-week',
       'native-country', 'income'],
      dtype='object')
```

In [51]:

```
data['age'].describe()
```

Out[51]:

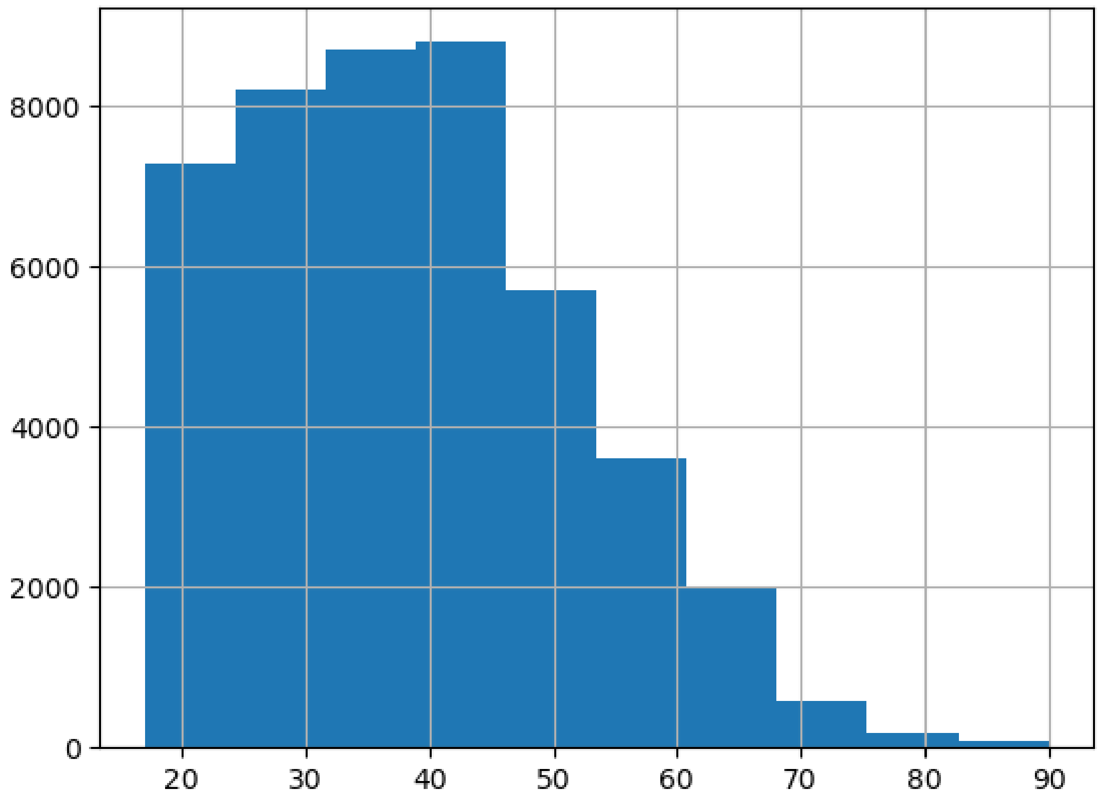
```
count    45175.000000
mean        38.556170
std         13.215349
min         17.000000
25%         28.000000
50%         37.000000
75%         47.000000
max         90.000000
Name: age, dtype: float64
```

In [52]:

```
data['age'].hist()
```

Out[52]:

```
<Axes: >
```



13. Find Total Number of persons having age between 17 to 48(inclusive) using the between method .

In [54]:

```
a=sum((data['age'] >= 17) & (data['age'] <=48))
```

In [55]:

```
a
```

Out[55]:

```
34858
```

In [56]:

```
sum(data['age'].between(17,48))
```

Out[56]:

```
34858
```

13. What is the distribution of workclass column ?

In [57]:

```
data.columns
```

Out[57]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'marital-status',
       'occupation', 'relationship', 'race', 'gender', 'hours-per-week',
       'native-country', 'income'],
      dtype='object')
```

In [60]:

```
data['workclass'].describe()
```

Out[60]:

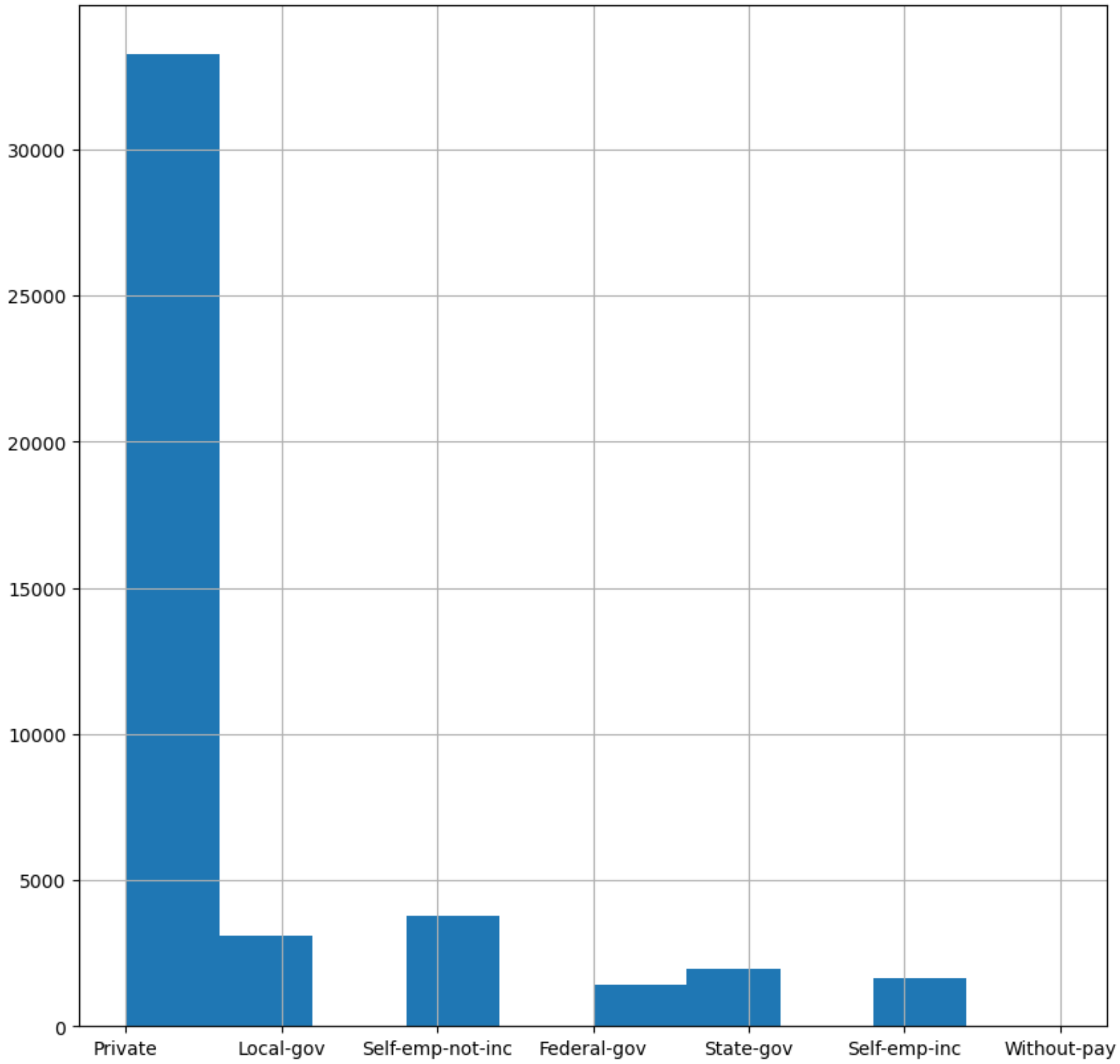
```
count       45175
unique          7
top       Private
freq        33262
Name: workclass, dtype: object
```

In [62]:

```
plt.figure(figsize=(10,10))
data['workclass'].hist()
```

Out[62]:

```
<Axes: >
```



14 .How many persons having Bachelors or Masters Degree.

In [63]:

```
data.columns
```

Out[63]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'marital-status',
       'occupation', 'relationship', 'race', 'gender', 'hours-per-week',
       'native-country', 'income'],
      dtype='object')
```

In [66]:

```
filter1 = data['education'] =='Bachelors'
filter2= data['education'] == 'Masters'
```

In [67]:

```
data[filter1 | filter2]
```

Out[67]:

	age	workclass	fnlwgt	education	marital-status	occupation	relationship	race	gende
11	36	Federal-gov	212465	Bachelors	Married-civ-spouse	Adm-clerical	Husband	White	Male
15	43	Private	346189	Masters	Married-civ-spouse	Exec-managerial	Husband	White	Male
20	34	Private	107914	Bachelors	Married-civ-spouse	Tech-support	Husband	White	Male
23	25	Private	220931	Bachelors	Never-married	Prof-specialty	Not-in-family	White	Male
24	25	Private	205947	Bachelors	Married-civ-spouse	Prof-specialty	Husband	White	Male
...	...	...	...	...	...	...	...	...	..
48817	34	Private	160216	Bachelors	Never-married	Exec-managerial	Not-in-family	White	Female
48819	38	Private	139180	Bachelors	Divorced	Prof-specialty	Unmarried	Black	Female
48825	31	Private	199655	Masters	Divorced	Other-service	Not-in-family	Other	Female
48834	32	Private	116138	Masters	Never-married	Tech-support	Not-in-family	Asian-Pac-Islander	Male
48835	53	Private	321865	Masters	Married-civ-spouse	Exec-managerial	Husband	White	Male

10072 rows × 12 columns

In [68]:

```
len(data[filter1 | filter2] )
```

Out[68]:

```
10072
```

In [69]:

```
sum(data['education'].isin(['Bachelors','Masters']))
```

Out[69]:

```
10072
```

Bivariate Analysis

It is a methodical statistical technique applied to a pair of variables (features/ attributes) of data to determine the empirical relationship between them. In order words, it is meant to determine any concurrent relations (usually over and above a simple correlation analysis).

In [70]:

```
data.columns
```

Out[70]:

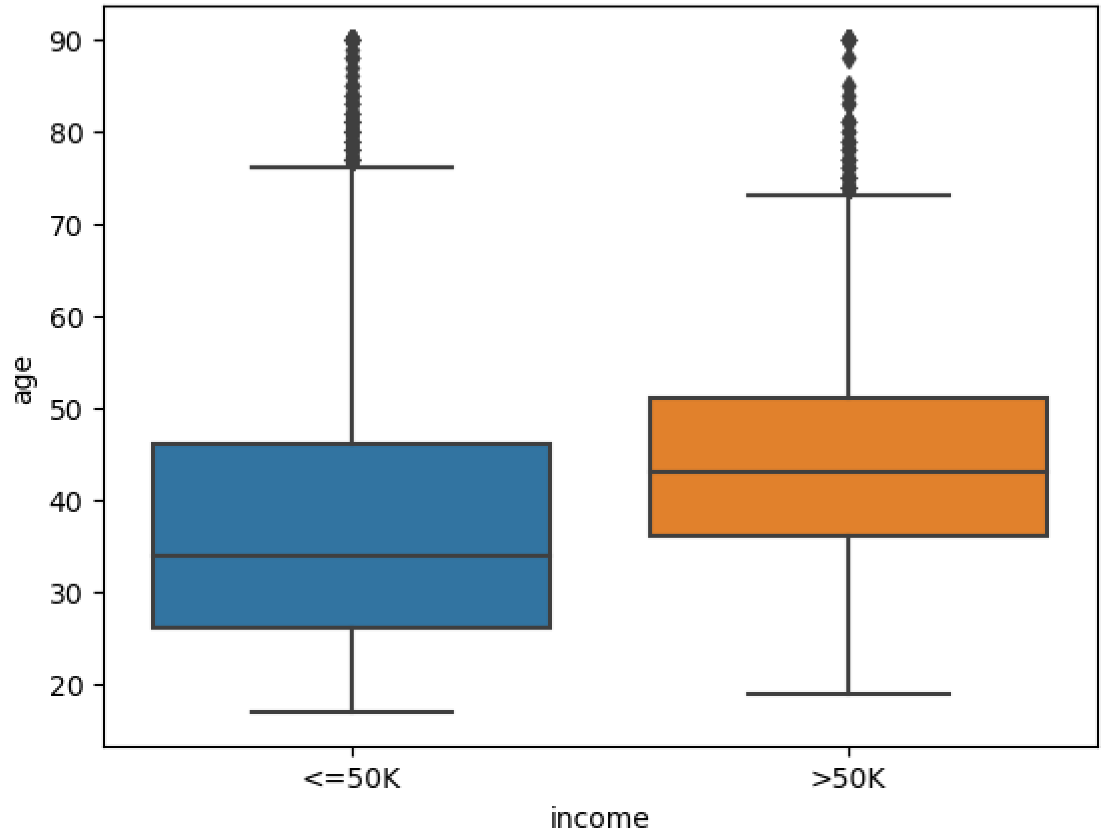
```
Index(['age', 'workclass', 'fnlwgt', 'education', 'marital-status',
       'occupation', 'relationship', 'race', 'gender', 'hours-per-week',
       'native-country', 'income'],
      dtype='object')
```

In [72]:

```
sns.boxplot(x='income',y='age',data=data)
```

Out[72]:

```
<Axes: xlabel='income', ylabel='age'>
```



15 . Replace Salary values ['<= 50k','>50k'] with 0 and 1

In [73]:

```
data.columns
```

Out[73]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'marital-status',
       'occupation', 'relationship', 'race', 'gender', 'hours-per-week',
       'native-country', 'income'],
      dtype='object')
```

In [75]:

```
data['income'].unique()
```

Out[75]:

```
array(['<=50K', '>50K'], dtype=object)
```

In [76]:

```
data['income'].value_counts()
```

Out[76]:

```
<=50K    33973
>50K     11202
Name: income, dtype: int64
```

In [83]:

```
def salary_data(sal):
    if sal=='<=50K':
        return 0
    else:
        return 1
```

In [84]:

```
data['encoded_income']=data['income'].apply(salary_data)
```

In [85]:

```
data.head(1)
```

Out[85]:

	age	workclass	fnlwgt	education	marital-status	occupation	relationship	race	gender	hours pe wee
0	25	Private	226802	11th	Never-married	Machine-op-inspct	Own-child	Black	Male	4

In [86]:

```
data.replace(to_replace=['<=50K', '>50K'],value=[0,1],inplace=True)
```

In [87]:

```
data.head(1)
```

Out[87]:

	age	workclass	fnlwgt	education	marital-status	occupation	relationship	race	gender	hours pe wee
0	25	Private	226802	11th	Never-married	Machine-op-inspct	Own-child	Black	Male	4

16. Which Workclass Getting the highest salary ?

In [89]:

```
data.groupby('workclass')['income'].mean().sort_values(ascending =False)
```

Out[89]:

```
workclass
Self-emp-inc        0.554407
Federal-gov         0.390469
Local-gov           0.295161
Self-emp-not-inc    0.279051
State-gov           0.267215
Private             0.217816
Without-pay         0.095238
Name: income, dtype: float64
```

17. Who has better chance to get salary > 50k Male or Females ?

In [92]:

```
data.groupby('gender')['income'].mean().sort_values(ascending=False)
```

Out[92]:

```
gender
Male      0.312609
Female    0.113692
Name: income, dtype: float64
```

18 Convert workclass columns Datatype to Category Dattype

In [93]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 45175 entries, 0 to 48841
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   age             45175 non-null  int64 
 1   workclass       45175 non-null  object
 2   fnlwgt          45175 non-null  int64 
 3   education       45175 non-null  object
 4   marital-status  45175 non-null  object
 5   occupation      45175 non-null  object
 6   relationship    45175 non-null  object
 7   race            45175 non-null  object
 8   gender          45175 non-null  object
 9   hours-per-week  45175 non-null  int64 
 10  native-country  45175 non-null  object
 11  income          45175 non-null  int64 
 12  encoded_income  45175 non-null  int64 
dtypes: int64(5), object(8)
memory usage: 4.8+ MB
```

In [95]:

```
data['workclass'] = data['workclass'].astype('category')
```

In [96]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 45175 entries, 0 to 48841
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype   
---  ------          --------------  -----   
 0   age             45175 non-null  int64   
 1   workclass       45175 non-null  category
 2   fnlwgt          45175 non-null  int64   
 3   education       45175 non-null  object  
 4   marital-status  45175 non-null  object  
 5   occupation      45175 non-null  object  
 6   relationship    45175 non-null  object  
 7   race            45175 non-null  object  
 8   gender          45175 non-null  object  
 9   hours-per-week  45175 non-null  int64   
 10  native-country  45175 non-null  object  
 11  income          45175 non-null  int64   
 12  encoded_income  45175 non-null  int64   
dtypes: category(1), int64(5), object(7)
memory usage: 4.5+ MB
```

In [ ]:

```

```