Workshop 1 - Data analysis with Pandas

Student Number: 2302546

In [1]:
```
#importing necessary libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
# Loading dataset to a dataframe.
data = pd.read_csv('adult.data.csv')
```

In [3]:
```
data.head()
```

Out[3]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	s
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	M
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	M
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	M
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	M
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Fem

Q1. Use head(2), head(10), tail(2). Explain your observations, in no more than 2 to 3 lines.

Solution:

In [4]:
```
data.head(2)
```

Out[4]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male

In [5]:
```
data.head(10)
```

Out[5]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	s
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	M
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	M
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	M
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	M
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Fem
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Fem
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Fem
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	M
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Fem
9	42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	M

In [6]:
```
data.tail(2)
```

Out[6]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race
32559	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	White
32560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White

Using data.head(2) streamlines the displayed data to only the first two data on the dataset, a similar reaction happens with data.head(10) which indicates that the number in the parenthesis specifies the number of lines it needs to display. While the command next to 'data.' specifies whether the displayed data needs to start from. While the function .tail(2) displays the lasttwo rows of the dataset, these command however affects only the rows while showing all the columns.

In [7]:
```
data.shape
```

Out[7]:
```
(32561, 15)
```

Generating your unique dataset for this task

In [8]:
```
data = data.sample(n=30000, random_state = 46)
```

In [9]:
```
data.shape
```

Out[9]:
```
(30000, 15)
```

In [10]:
```
data.describe()
```

Out[10]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week
count	30000.000000	3.000000e+04	30000.000000	30000.000000	30000.000000	30000.000000
mean	38.580167	1.895246e+05	10.071033	1063.000233	87.530100	40.457667
std	13.650360	1.048394e+05	2.573482	7286.103159	403.745767	12.381434
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.180010e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.783155e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.367048e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.484705e+06	16.000000	99999.000000	4356.000000	99.000000

In [11]:
```
data['education-num'].value_counts()
```

Out[11]:
```
9     9712
10    6711
13    4912
14    1576
11    1264
7     1073
12     999
6      864
4      607
15     521
5      477
8      397
16     377
3      300
2      160
1       50
Name: education-num, dtype: int64
```

In [12]:
```
data['education'].value_counts()
```

Out[12]:
```
 HS-grad         9712
 Some-college    6711
 Bachelors       4912
 Masters         1576
 Assoc-voc       1264
 11th            1073
 Assoc-acdm       999
 10th             864
 7th-8th          607
 Prof-school      521
 9th              477
 12th             397
 Doctorate        377
 5th-6th          300
 1st-4th          160
 Preschool         50
Name: education, dtype: int64
```

In [13]:
```
data = data.drop(['fnlwgt'], axis=1)
```

In [14]:
```
data.shape
```

Out[14]:
```
(30000, 14)
```

In [15]:
```
data.describe(include='all')
```

Out[15]:

	age	workclass	education	education-num	marital-status	occupation	relationship	ra
count	30000.000000	30000	30000	30000.000000	30000	30000	30000	300
unique	NaN	9	16	NaN	7	15	6	
top	NaN	Private	HS-grad	NaN	Married-civ-spouse	Prof-specialty	Husband	Wh
freq	NaN	20917	9712	NaN	13817	3803	12181	256
mean	38.580167	NaN	NaN	10.071033	NaN	NaN	NaN	Na
std	13.650360	NaN	NaN	2.573482	NaN	NaN	NaN	Na
min	17.000000	NaN	NaN	1.000000	NaN	NaN	NaN	Na
25%	28.000000	NaN	NaN	9.000000	NaN	NaN	NaN	Na
50%	37.000000	NaN	NaN	10.000000	NaN	NaN	NaN	Na
75%	48.000000	NaN	NaN	12.000000	NaN	NaN	NaN	Na
max	90.000000	NaN	NaN	16.000000	NaN	NaN	NaN	Na

In [16]:
```
data['education'].value_counts()
```

Out[16]:
```
HS-grad         9712
Some-college    6711
Bachelors       4912
Masters         1576
Assoc-voc       1264
11th            1073
Assoc-acdm       999
10th             864
7th-8th          607
Prof-school      521
9th              477
12th             397
Doctorate        377
5th-6th          300
1st-4th          160
Preschool         50
Name: education, dtype: int64
```

In [17]:
```
data['education'].nunique()
```

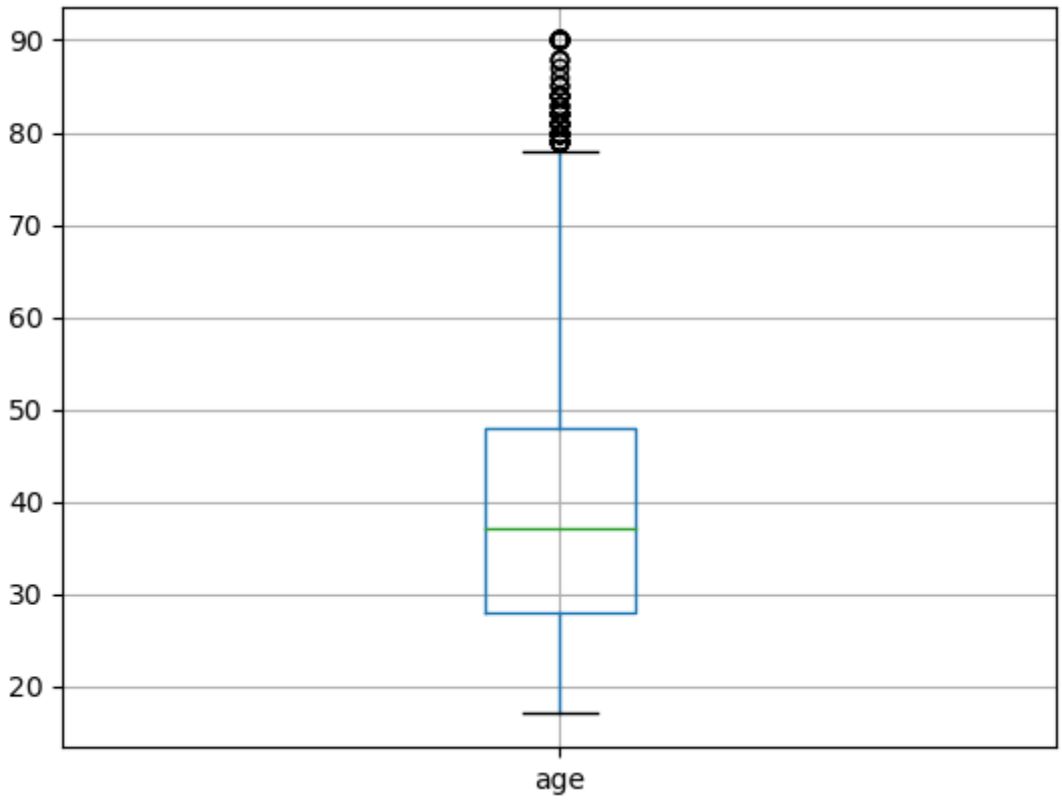
Out[17]:
```
16
```

In [18]:
```
data['age'].value_counts()
```

Out[18]:
```
36    830
34    825
31    818
33    813
35    810
     ...
83      5
85      3
88      3
86      1
87      1
Name: age, Length: 73, dtype: int64
```

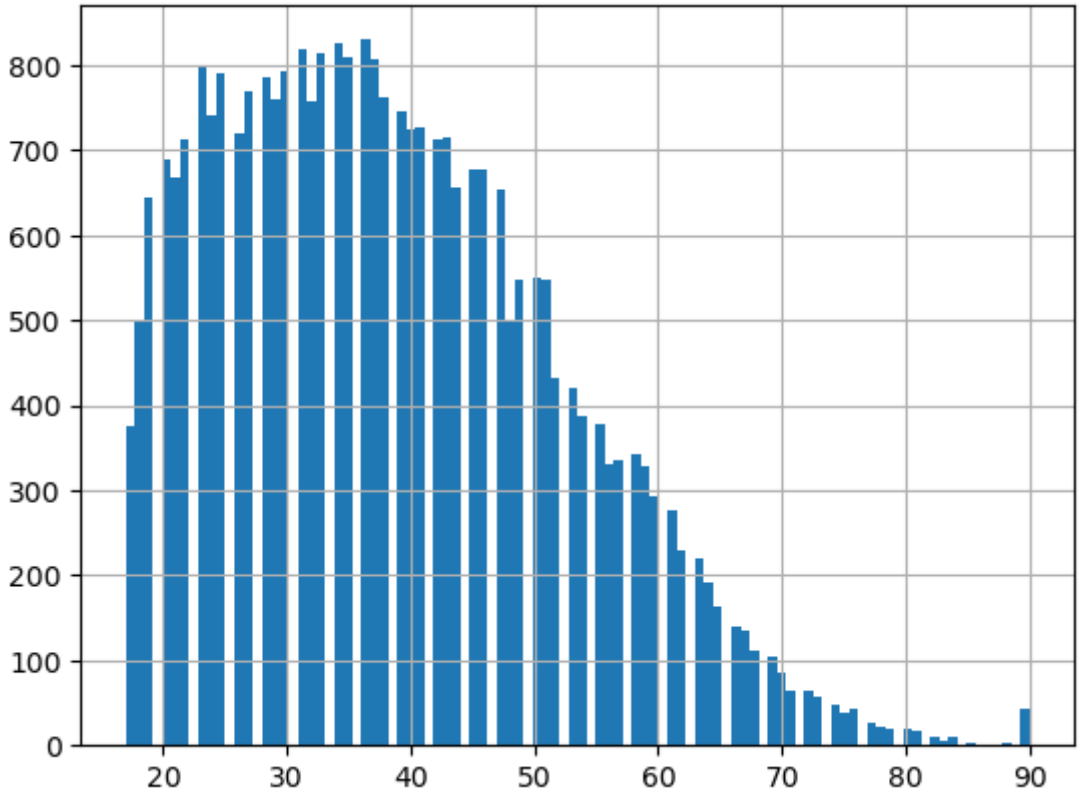
In [19]:
```
data.boxplot(column='age')
```

Out[19]:
```
<AxesSubplot:>
```



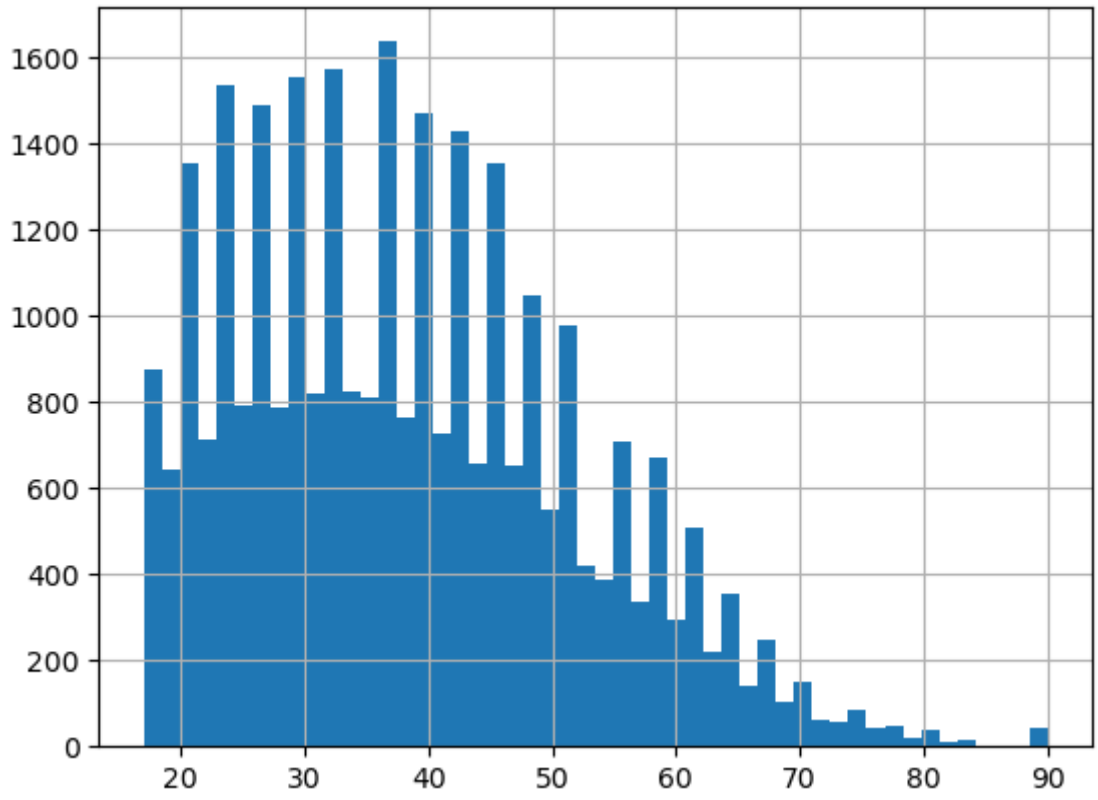
In [20]:
```
data['age'].hist(bins=100)
```

Out[20]:
```
<AxesSubplot:>
```



In [21]:
```
data.age.hist(bins=50)
```

Out[21]:
```
<AxesSubplot:>
```



In [22]:
```
data['sex'].value_counts()
```

Out[22]:
```
 Male      20091
 Female     9909
Name: sex, dtype: int64
```

In [23]:
```
data.columns
```

Out[23]:
```
Index(['age', 'workclass', 'education', 'education-num', 'marital-status',
       'occupation', 'relationship', 'race', 'sex', 'capital-gain',
       'capital-loss', 'hours-per-week', 'native-country', 'class-label'],
      dtype='object')
```

In [24]:
```
data['workclass'].value_counts()
```

Out[24]:
```
 Private             20917
 Self-emp-not-inc     2341
 Local-gov            1924
 ?                    1693
 State-gov            1184
 Self-emp-inc         1036
 Federal-gov           885
 Without-pay            13
 Never-worked            7
Name: workclass, dtype: int64
```

Q2. How many males and females exist in the dataset? In a new cell, use a correct command to answer the question and write your answer.

Solution:

In [25]:
```
data['sex'].value_counts()
```

Out[25]:
```
 Male      20091
 Female     9909
Name: sex, dtype: int64
```

data['sex'].value_counts()

We have 20,091 Males and 9,909 females in the dataset according to Out[25]

Applying groupby functions in order to summarise the data.

In [26]:
```
data['age'].groupby([data['sex']]).mean()
```

Out[26]:
```
sex
 Female    36.827531
 Male      39.444577
Name: age, dtype: float64
```

In [27]:
```
data['age'].groupby([data['sex'],data['education']]).mean()
```

Out[27]:
```
sex      education
 Female   10th            35.109091
          11th            30.311224
          12th            30.257576
          1st-4th         48.577778
          5th-6th         43.972603
          7th-8th         49.925676
          9th             42.132353
          Assoc-acdm      36.275689
          Assoc-voc       37.594421
          Bachelors       35.649038
          Doctorate       45.038961
          HS-grad         38.597771
          Masters         43.103659
          Preschool       41.750000
          Prof-school     40.244186
          Some-college    33.765140
 Male     10th            38.371817
          11th            33.493392
          12th            32.845283
          1st-4th         45.495652
          5th-6th         42.352423
          7th-8th         48.045752
          9th             40.519062
          Assoc-acdm      38.048333
          Assoc-voc       38.691729
          Bachelors       40.307870
          Doctorate       48.550000
          HS-grad         39.170268
          Masters         44.459410
          Preschool       43.323529
          Prof-school     45.487356
          Some-college    37.048609
Name: age, dtype: float64
```

Q3. What is the average contribution to capital-gain of each sex and occupation category?

In [28]:
```
data['capital-gain'].groupby([data['sex'],data['occupation']]).mean()
```

Out[28]:
```
sex      occupation
 Female   ?                   326.553966
          Adm-clerical        474.735745
          Craft-repair        796.308057
          Exec-managerial    1058.988837
          Farming-fishing     691.385965
          Handlers-cleaners   105.798701
          Machine-op-inspct   177.585170
          Other-service       154.563603
          Priv-house-serv     300.323077
          Prof-specialty     1295.553791
          Protective-serv    1734.301370
          Sales               288.785775
          Tech-support        702.964968
          Transport-moving    438.146341
 Male     ?                   855.050483
          Adm-clerical        494.631255
          Armed-Forces          0.000000
          Craft-repair        591.091339
          Exec-managerial    2761.695231
          Farming-fishing     569.687427
          Handlers-cleaners   285.102210
          Machine-op-inspct   405.017319
          Other-service       247.460260
          Priv-house-serv      74.250000
          Prof-specialty     3442.656741
          Protective-serv     549.540698
          Sales              1872.088116
          Tech-support        697.527002
          Transport-moving    408.084527
Name: capital-gain, dtype: float64
```

Utilizing the "groupby" function, I grouped the sex and occupation by the capital gain data with the mean in consideration.

In [29]:
```
data['occupation'].groupby([data['sex']]).value_counts()
```

Out[29]:
```
sex      occupation
 Female   Adm-clerical         2350
          Other-service        1643
          Prof-specialty       1385
          Sales                1167
          Exec-managerial      1075
          ?                     769
          Machine-op-inspct     499
          Tech-support          314
          Craft-repair          211
          Handlers-cleaners     154
          Priv-house-serv       130
          Transport-moving       82
          Protective-serv        73
          Farming-fishing        57
 Male     Craft-repair         3591
          Exec-managerial      2684
          Prof-specialty       2418
          Sales                2213
          Transport-moving     1396
          Other-service        1384
          Machine-op-inspct    1328
          Adm-clerical         1139
          Handlers-cleaners    1086
          ?                     931
          Farming-fishing       851
          Tech-support          537
          Protective-serv       516
          Armed-Forces            9
          Priv-house-serv         8
Name: occupation, dtype: int64
```

Q4. Identify the average capital-gain by males and females accross different marital-status.

In [30]:
```
data['capital-gain'].groupby([data['sex'],data['marital-status']]).mean()
```

Out[30]:
```
sex      marital-status
 Female   Divorced                 416.625254
          Married-AF-spouse          0.000000
          Married-civ-spouse      1618.297760
          Married-spouse-absent    245.388298
          Never-married            345.180100
          Separated                357.830743
          Widowed                  492.782034
 Male     Divorced                1140.543385
          Married-AF-spouse        912.250000
          Married-civ-spouse      1746.369542
          Married-spouse-absent    987.148515
          Never-married            420.263332
          Separated                870.424658
          Widowed                  799.467532
Name: capital-gain, dtype: float64
```

Capital gain data grouped by the sex and marital status with mean in consideration shown in my input above to give the classification as requested by the question.

In [31]:
```
data['race'].value_counts()
```

Out[31]:
```
 White                25624
 Black                 2901
 Asian-Pac-Islander     945
 Amer-Indian-Eskimo     285
 Other                  245
Name: race, dtype: int64
```

Question. What is the maximum age accross differnt races?

In [32]:
```
data['age'].groupby([data['race']]).max()
```

Out[32]:
```
race
 Amer-Indian-Eskimo    82
 Asian-Pac-Islander    90
 Black                 90
 Other                 77
 White                 90
Name: age, dtype: int64
```

Q5. Are minimum and maximum age by sex same?

In [33]:
```
#Minimum age by sex:
data['age'].groupby([data['sex']]).min()
```

Out[33]:
```
sex
 Female    17
 Male      17
Name: age, dtype: int64
```

In [34]:
```
#Maximum age by sex
data['age'].groupby([data['sex']]).max()
```

Out[34]:
```
sex
 Female    90
 Male      90
Name: age, dtype: int64
```

- Maximum age for male and female is the same (90)
- Minimum age for male and female is also the same (17)
- However, there is a difference between the maximum and the minimum age of the sexes.

Data Visualisation

In [35]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

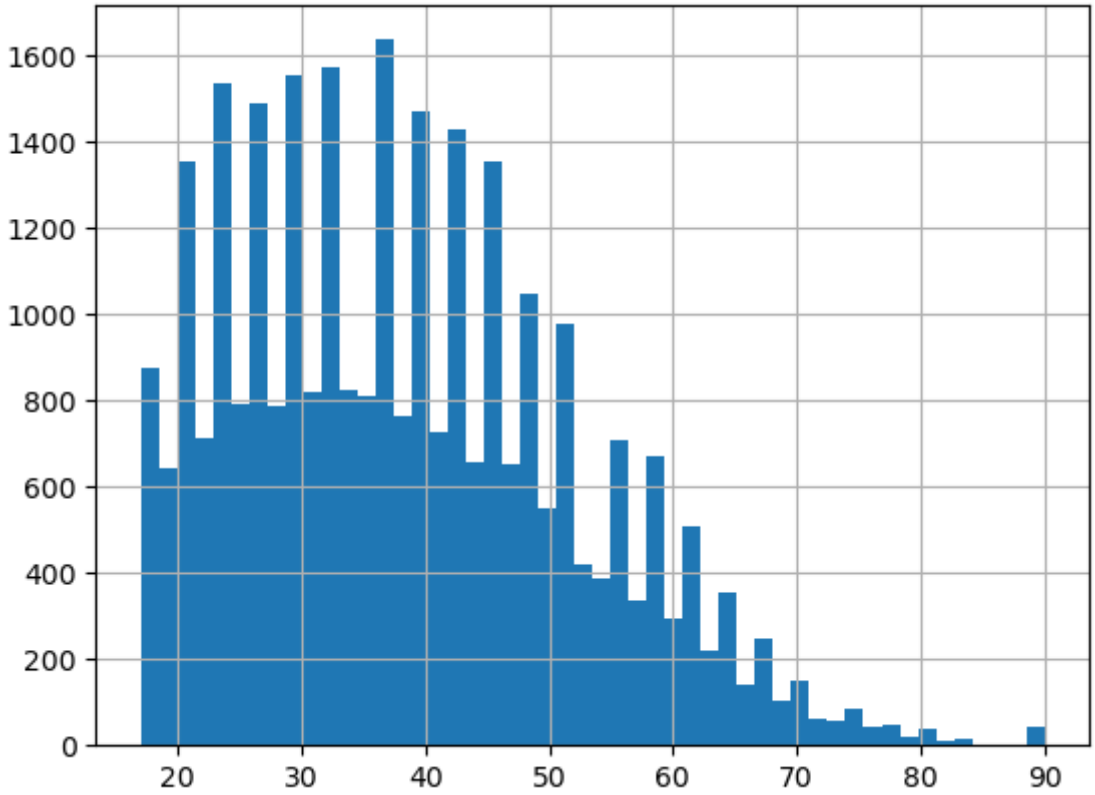
In [36]:
```
data.describe()
```

Out[36]:

	age	education-num	capital-gain	capital-loss	hours-per-week
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000
mean	38.580167	10.071033	1063.000233	87.530100	40.457667
std	13.650360	2.573482	7286.103159	403.745767	12.381434
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000
75%	48.000000	12.000000	0.000000	0.000000	45.000000
max	90.000000	16.000000	99999.000000	4356.000000	99.000000

In [37]:
```
data['age'].hist(bins=50)
```

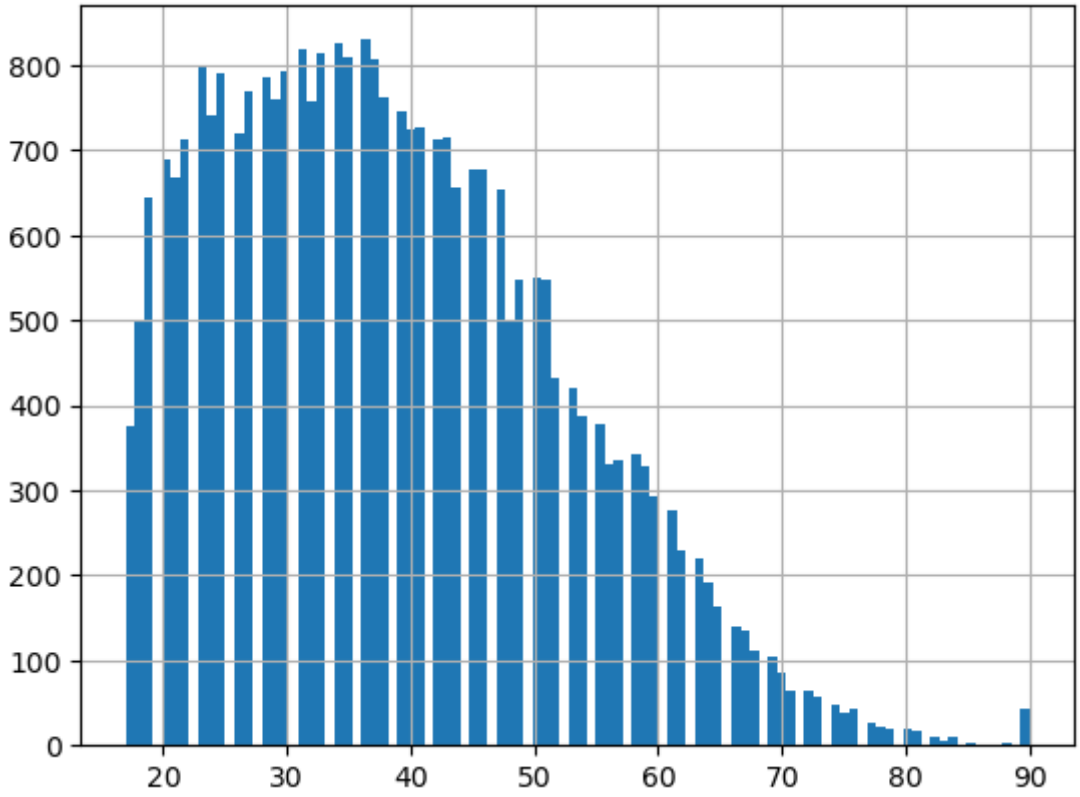
Out[37]:
```
<AxesSubplot:>
```



Try-it-yourself

In [38]:
```
data['age'].hist(bins=100)
```

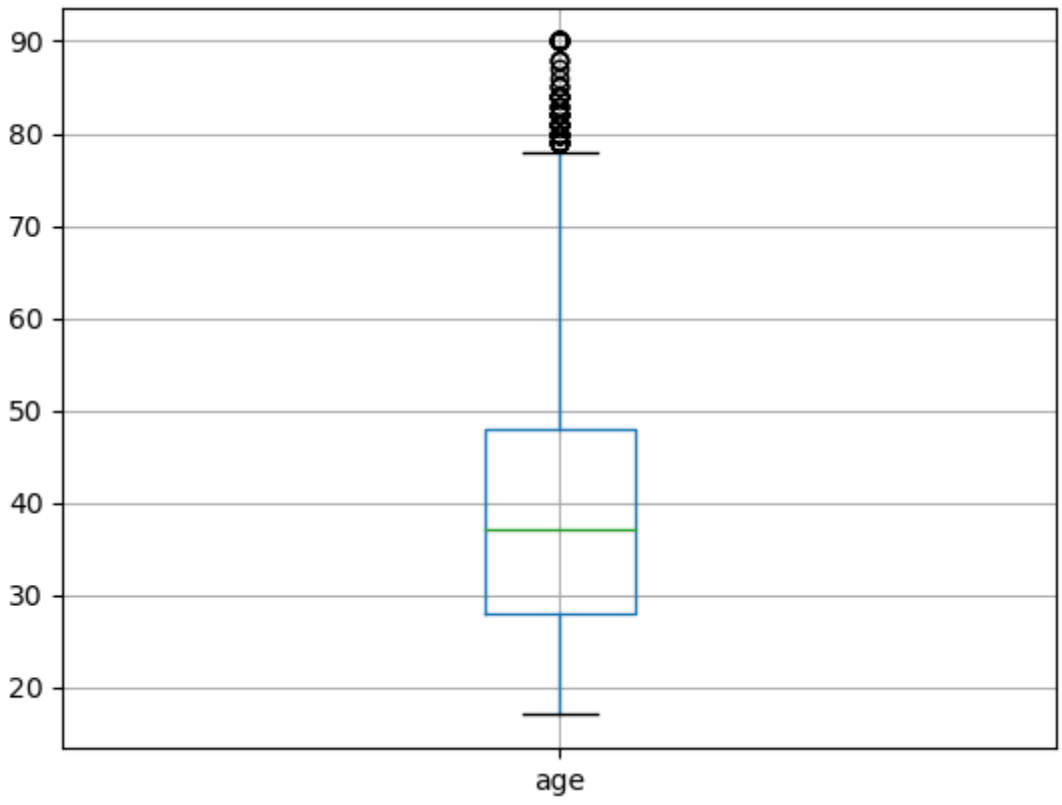
Out[38]:
```
<AxesSubplot:>
```



There are differences between the two graphs. The width of the bars differs as well the bar's fusion.

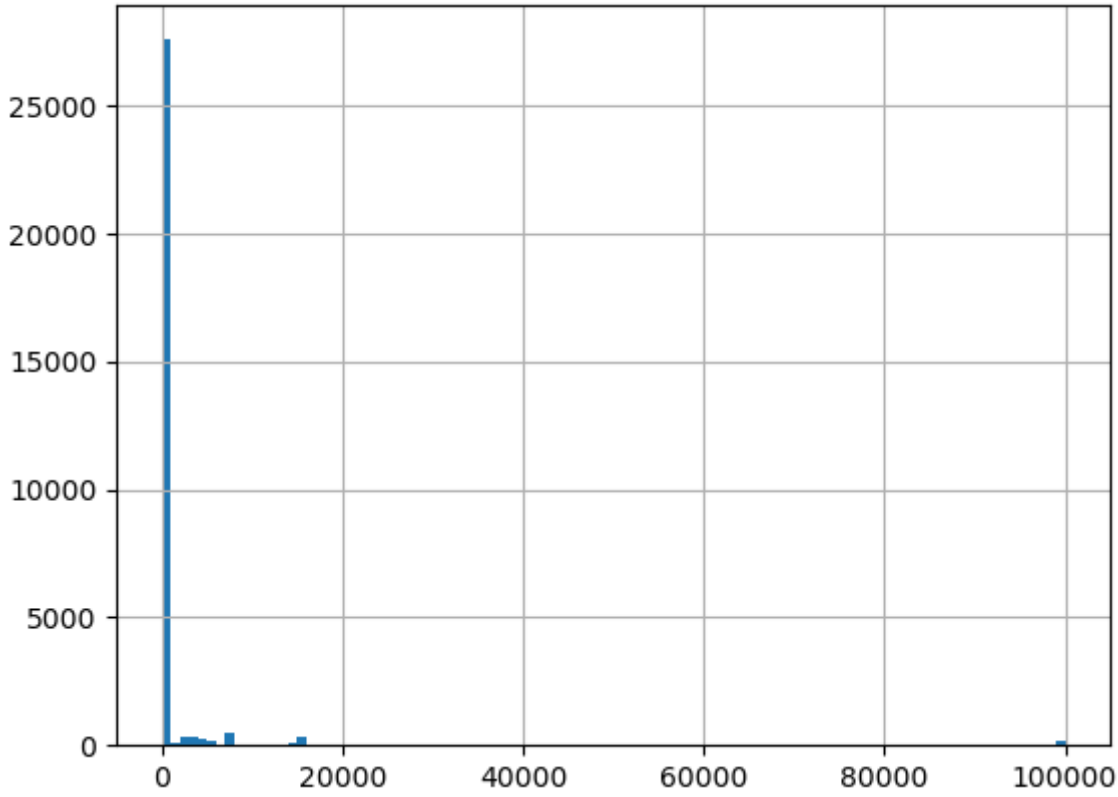
In [39]:
```
data.boxplot(column='age')
```

Out[39]:
```
<AxesSubplot:>
```



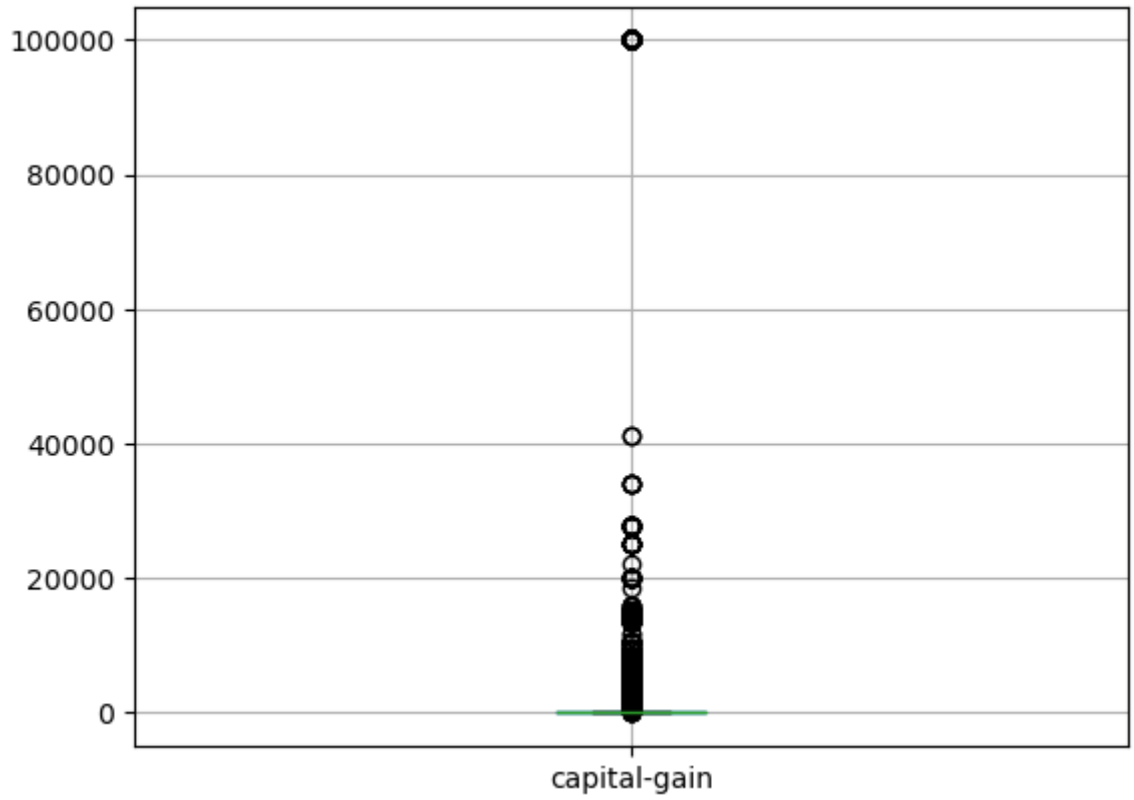
In [40]:
```
data['capital-gain'].hist(bins=100)
```

Out[40]:
```
<AxesSubplot:>
```



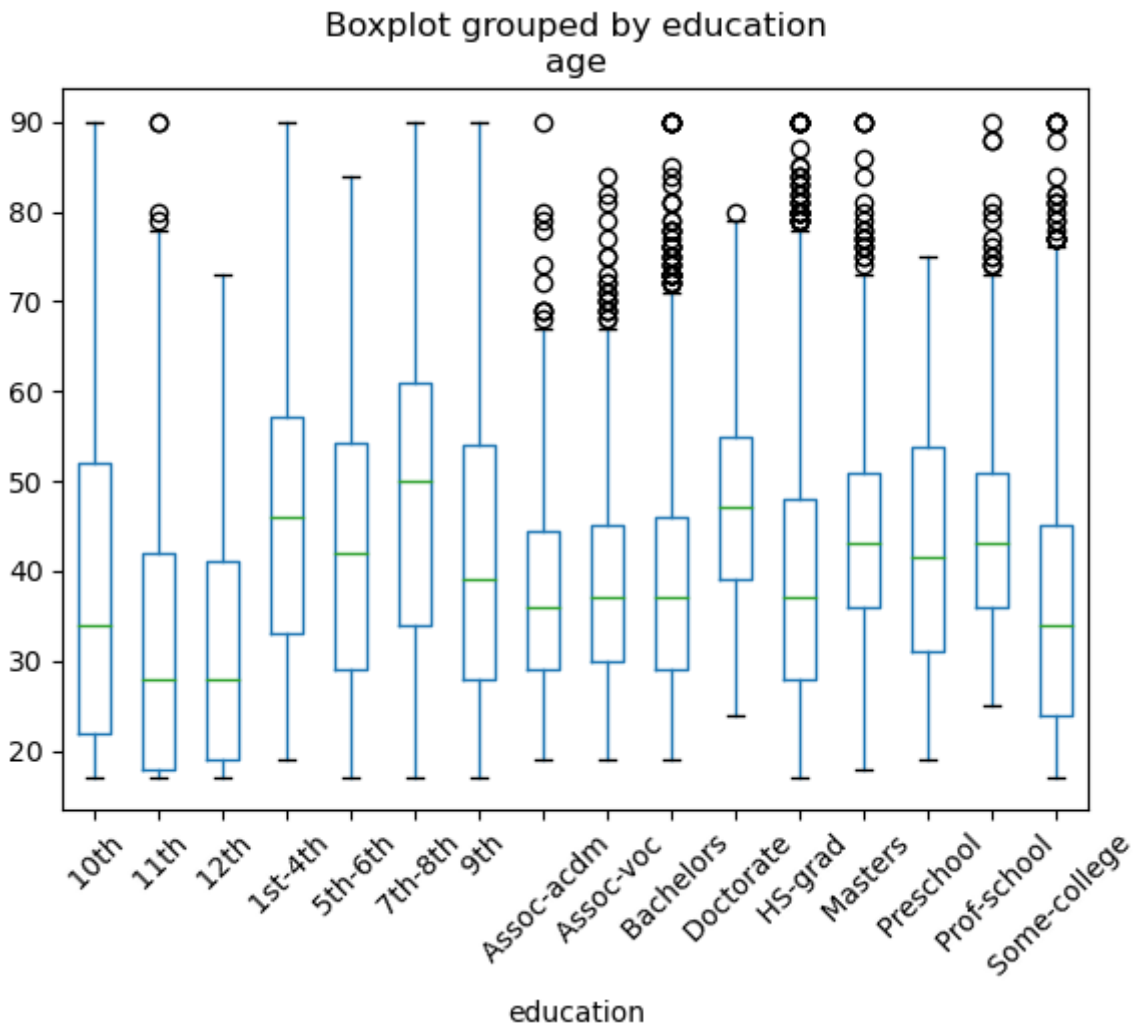
In [41]:
```
data.boxplot(column='capital-gain')
```

Out[41]:
```
<AxesSubplot:>
```



In [42]:
```
data.boxplot(column='age',by = 'education', grid=False, rot = 45, fontsize = 10)
```

Out[42]:
```
<AxesSubplot:title={'center':'age'}, xlabel='education'>
```

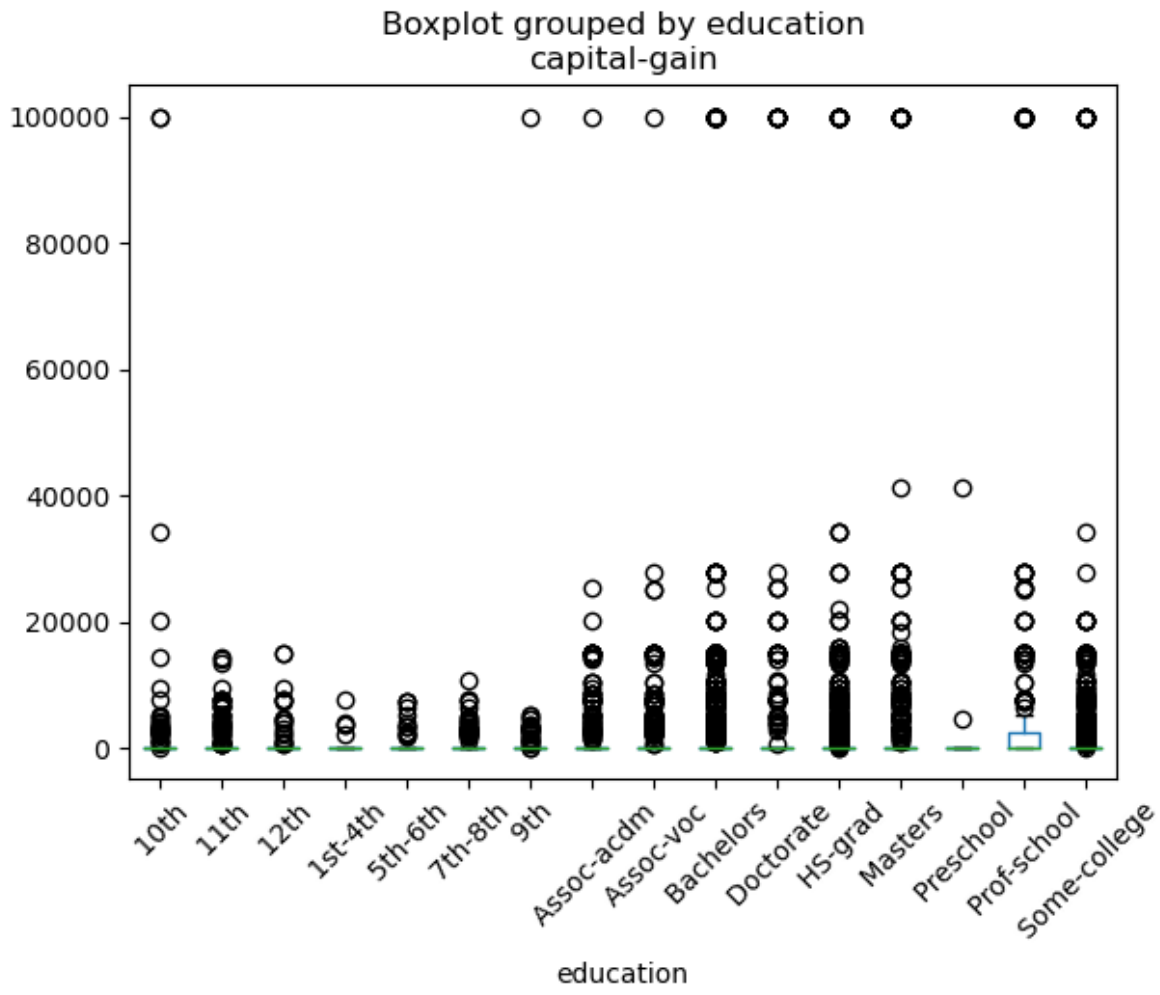


In [43]:
```
data['education'].value_counts()
```

Out[43]:
```
HS-grad         9712
Some-college    6711
Bachelors       4912
Masters         1576
Assoc-voc       1264
11th            1073
Assoc-acdm       999
10th             864
7th-8th          607
Prof-school      521
9th              477
12th             397
Doctorate        377
5th-6th          300
1st-4th          160
Preschool         50
Name: education, dtype: int64
```

In [44]:
```
data.boxplot(column='capital-gain', by = 'education', grid=False, rot = 45, fontsi
```

Out[44]:
```
<AxesSubplot:title={'center':'capital-gain'}, xlabel='education'>
```



In [45]:
```
data['marital-status'].value_counts()
```

Out[45]:
```
 Married-civ-spouse       13817
 Never-married             9830
 Divorced                  4088
 Separated                  944
 Widowed                    911
 Married-spouse-absent      390
 Married-AF-spouse           20
Name: marital-status, dtype: int64
```

Checking NULL values in the dataset

In [46]:
```
data.apply(lambda x: sum(x.isnull()), axis = 0)
```

Out[46]:
```
age              0
workclass        0
education        0
education-num    0
marital-status   0
occupation       0
relationship     0
race             0
sex              0
capital-gain     0
capital-loss     0
hours-per-week   0
native-country   0
class-label      0
dtype: int64
```

In [47]:
```
data['native-country'].value_counts()
```

Out[47]:
```
United-States                 26898
Mexico                          594
?                               536
Philippines                     176
Germany                         126
Canada                          111
Puerto-Rico                     106
El-Salvador                     100
India                            91
Cuba                             89
England                          86
South                            77
Jamaica                          72
China                            70
Italy                            65
Dominican-Republic               64
Vietnam                          62
Guatemala                        59
Japan                            53
Poland                           52
Columbia                         50
Taiwan                           44
Haiti                            41
Iran                             37
Portugal                         33
Nicaragua                        30
Peru                             29
France                           28
Ecuador                          27
Greece                           27
Ireland                          23
Trinadad&Tobago                  18
Laos                             17
Hong                             17
Cambodia                         16
Thailand                         15
Yugoslavia                       14
Honduras                         13
Outlying-US(Guam-USVI-etc)       13
Hungary                          12
Scotland                          9
Name: native-country, dtype: int64
```

In [48]:
```
data['occupation'].groupby([data['sex']]).value_counts()
```

Out[48]:
```
sex      occupation
 Female   Adm-clerical         2350
          Other-service        1643
          Prof-specialty       1385
          Sales                1167
          Exec-managerial      1075
          ?                     769
          Machine-op-inspct     499
          Tech-support          314
          Craft-repair          211
          Handlers-cleaners     154
          Priv-house-serv       130
          Transport-moving       82
          Protective-serv        73
          Farming-fishing        57
 Male     Craft-repair         3591
          Exec-managerial      2684
          Prof-specialty       2418
          Sales                2213
          Transport-moving     1396
          Other-service        1384
          Machine-op-inspct    1328
          Adm-clerical         1139
          Handlers-cleaners    1086
          ?                     931
          Farming-fishing       851
          Tech-support          537
          Protective-serv       516
          Armed-Forces            9
          Priv-house-serv         8
Name: occupation, dtype: int64
```

Data transformation

Label Encoding

In [49]:
```
from sklearn.preprocessing import LabelEncoder
```

In [50]:
```
data.head()
```

Out[50]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex
11568	32	Private	HS-grad	9	Married-civ-spouse	Adm-clerical	Husband	White	Male
28705	59	Federal-gov	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Female
4451	19	Private	HS-grad	9	Never-married	Craft-repair	Not-in-family	White	Female
26294	27	Private	HS-grad	9	Never-married	Adm-clerical	Not-in-family	White	Male
10658	57	Private	Some-college	10	Married-civ-spouse	Sales	Husband	White	Male

In [51]:
```
data.dtypes
```

Out[51]:
```
age                int64
workclass         object
education         object
education-num      int64
marital-status    object
occupation        object
relationship      object
race              object
sex               object
capital-gain       int64
capital-loss       int64
hours-per-week     int64
native-country    object
class-label       object
dtype: object
```

In [52]:
```
columns = list(data.select_dtypes(exclude=['int64']))
```

In [53]:
```
columns
```

Out[53]:
```
['workclass',
 'education',
 'marital-status',
 'occupation',
 'relationship',
 'race',
 'sex',
 'native-country',
 'class-label']
```

In [54]:
```
data['class-label'].value_counts()
```

Out[54]:
```
 <=50K    22799
 >50K      7201
Name: class-label, dtype: int64
```

In [55]:
```
le = LabelEncoder()
for i in columns:
    #print(i)
    data[i] = le.fit_transform(data[i])
```

```
data.dtypes
```

Out[55]:
```
age              int64
workclass        int32
education        int32
education-num    int64
marital-status   int32
occupation       int32
relationship     int32
race             int32
sex              int32
capital-gain     int64
capital-loss     int64
hours-per-week   int64
native-country   int32
class-label      int32
dtype: object
```

In [56]:
```
data.head()
```

Out[56]:

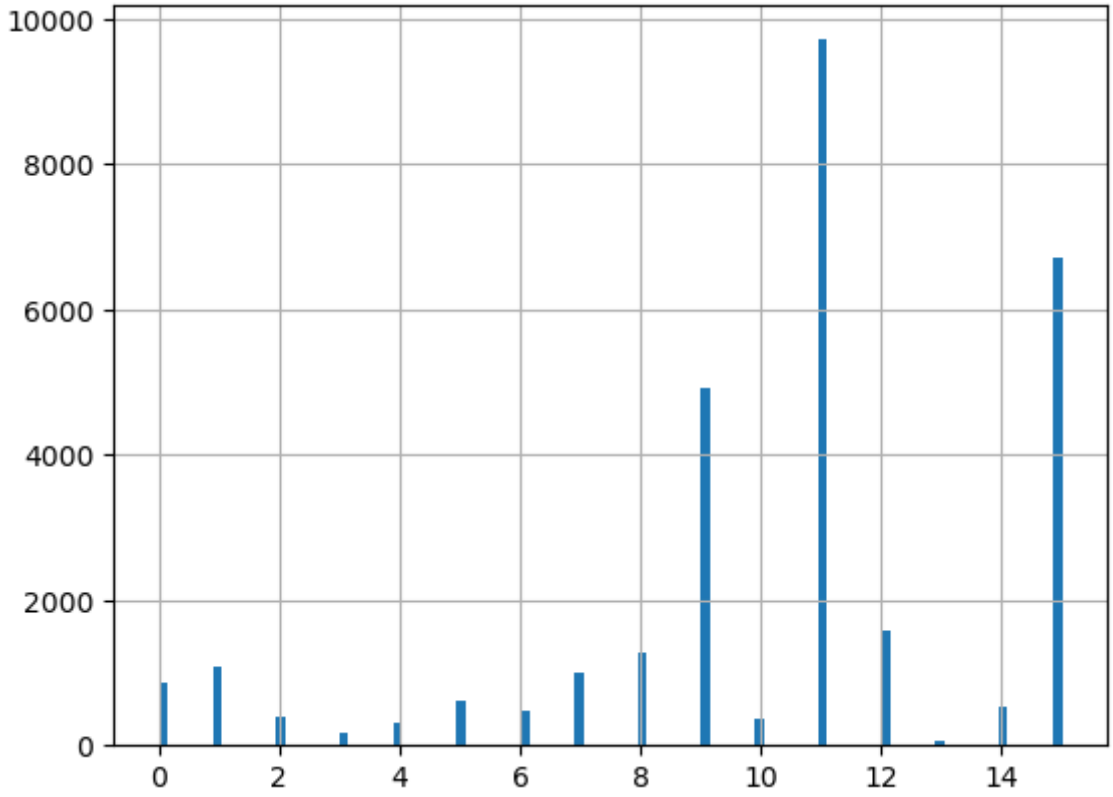
	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capit g
11568	32	4	11	9	2	1	0	4	1	
28705	59	1	12	14	4	10	1	4	0	
4451	19	4	11	9	4	3	1	4	0	
26294	27	4	11	9	4	1	1	4	1	
10658	57	4	15	10	2	12	0	4	1	

In [57]:
```
data['workclass'].value_counts()
```

Out[57]:
```
4    20917
6     2341
2     1924
0     1693
7     1184
5     1036
1      885
8       13
3        7
Name: workclass, dtype: int64
```

In [58]:
```
data['education'].hist(bins=100)
```

Out[58]:
```
<AxesSubplot:>
```



In [59]:
```
data.describe(include='all')
```

Out[59]:

	age	workclass	education	education-num	marital-status	occupation	relati
count	30000.000000	30000.00000	30000.000000	30000.000000	30000.000000	30000.000000	30000.(
mean	38.580167	3.86800	10.296000	10.071033	2.611000	6.566667	1.4
std	13.650360	1.45487	3.869067	2.573482	1.505065	4.234133	1.(
min	17.000000	0.00000	0.000000	1.000000	0.000000	0.000000	0.(
25%	28.000000	4.00000	9.000000	9.000000	2.000000	3.000000	0.(
50%	37.000000	4.00000	11.000000	10.000000	2.000000	7.000000	1.(
75%	48.000000	4.00000	12.000000	12.000000	4.000000	10.000000	3.(
max	90.000000	8.00000	15.000000	16.000000	6.000000	14.000000	5.(

Report:

Q6. What is the summary of the data.describe()?

Data.describe() outputs statistical summary of the dataset, which include count, mean value, the standard deviation, minimum, 1st quartile, the median, the third quartile and the maximum value. The count indicates the total number of values in each column which in this case is 30,000

- For age:
  - The average value of the age attribute is 38
  - The age ranges from 90 to 17
  - 25% of the age data lies below age 28. However, the third quartile (75%) lies below age 48.
  - The data is more dispersed after the third quartile as the difference between the age value between the first 25% and the minimum age value is much lesser than the difference between the last 25% of the age value and maximum age value.
- For capital-gain:
  - The average value returns 1063.00 while median is zero indicating that it is highly right skewed
  - The values of capital-gain are higly concentrated on one partiular value which happens to be after the 3rd quartile as values from min till third quartile is zero and max get its own value and resulted in the large standard deviation
- The average number of hours worked per week is 40.46 hours with a standard deviation of 12.38 hours. The minimum number of hours worked is 1 hour and the maximum number of hours worked is 99 hours.
- The majority of the individuals are from the United States (native country 38) and the class label indicates whether their income is above or below 50,000 dollars per year. The class label is a binary variable with 0 representing income below 50,000 dollars and 1 representing income above 50,000 dollars.

Q7. What are the different data types (or attribut types) in data mining? Explain with the help of the examples from Adult dataset. HINT: Don't get confused with data types in Python or Pandas.

In data mining, there are typically two main data types or attribute types: nominal and numeric.

Nominal attributes are categorical and don't have an order or meaningful magnitude, e.g. gender, occupation, race, etc. In the Adult dataset, the "workclass", "education", "marital-status", "occupation", "relationship", "race", "sex", and "native-country" attributes are nominal.

Numeric attributes are numerical and have magnitude, e.g. age, capital-gain, capital-loss, etc. In the Adult dataset, the "age", "fnlwgt", "educational-num", "capital-gain", "capital-loss", and "hours-per-week" attributes are numeric.

Ordinal Attributes is used to represent data that has a natural ordering or sequence. e.g Preschool, 1st-4th, 5th-6th, 7th-8th, 9th, 10th, 11th, 12th, HS-grad, Some-college, Assoc-voc, Assoc-acdm, Bachelors, Masters, and Prof-school. These values have a natural ordering, with higher values indicating higher levels of education.

Ratio: Ratio data is used to represent data where the ratio between any two values is meaningful. Examples of ratio data in the Adult dataset include the age attribute, capital-gain, capital-loss, and hours-per-week. These attributes have values that have a meaningful ratio, for example, an individual who is 40 years old is twice as old as an individual who is 20 years old. Similarly, an individual who works 40 hours per week is twice as much as an individual who works 20 hours per week.

Q8. Highest migrants belongs to which country?

In [60]:
```
data['native-country'].value_counts()
```

Out[60]:
```
38    26898
25      594
0       536
29      176
11      126
2       111
32      106
8       100
18       91
5        89
9        86
34       77
22       72
3        70
21       65
6        64
39       62
13       59
23       53
30       52
4        50
35       44
14       41
19       37
31       33
26       30
28       29
10       28
7        27
12       27
20       23
37       18
24       17
16       17
1        16
36       15
40       14
15       13
27       13
17       12
33        9
Name: native-country, dtype: int64
```

The country with the highest migrants is Mexico which corresponds to the code '25' with a count of 594. United States has an higher counts but is not the answer as the dataset is from the United States.

Q9. Which occupation represents more males than females?

In [61]:
```
data['occupation'].groupby([data['sex']]).value_counts()
```

Out[61]:
```
sex  occupation
0    1             2350
     8             1643
     10            1385
     12            1167
     4             1075
     0              769
     7              499
     13             314
     3              211
     6              154
     9              130
     14              82
     11              73
     5               57
1    3             3591
     4             2684
     10            2418
     12            2213
     14            1396
     8             1384
     7             1328
     1             1139
     6             1086
     0              931
     5              851
     13             537
     11             516
     2                9
     9                8
Name: occupation, dtype: int64
```

There are multiple occupations with more male than females, the output[61] shows this with an encoded list. However, I ran the code before encoding in input[48], the occupations with more males than females includes Craft-repair, Handlers-cleaners, Prof-specialty among others.

Q10. What is the difference between data.head() and data.tail()?

data.head() displays the first n rows of a dataset whereas data.tail() displays the last n rows of the dataset. Where the default value of n is 5 if unspecified.

References

Dua, D., & Graff, C. (2017). UCI Machine Learning Repository. University of California, Irvine, School of Information. http://archive.ics.uci.edu/ml