DATA ANALYSIS WITH PANDAS

This data was extracted from the Census Bureau Database found at http://www.census.gov/ftp/pub/DES/www/welcome.htn

Donor: Ronny Kohavi and Barry Becker.

Extraction was done by Barry Becker from the 1994 Census database.

Exploratory analysis: Loading and exploring the dataset

In [1]:
```
#Importing necessary libaries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
data = pd.read_csv(r'C:\Users\LenovoX260\Desktop\Data Minning and Informatics Assignme
```

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

Out[3]:

	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
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female

data.head() - This code displays the fist 5 rows of the dataset from 0-4

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

Out[4]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	ca
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	

Observation

data.head(2) - This shows the first 2 rows of the dataset from 0-1

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

Out[5]:

	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
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Female
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Female
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Female
9	42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male

Observation

data.head(10) - This shows the first 10 rows of the data set from 0-9

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	M
32560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Fem

Observation

data.tail(2) - This shows the last 2 rows of the data set

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

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

The above code shows the number of rows and columns in the dataset. In this data there are 32561 rows and 15 columns

Generating your unique dataset for this task

In this section we will be generating a unique dataset by replacing the last two digits in random state with 17

In [8]:
```
data = data.sample(n=30000, random_state = 17)
```

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

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

Running the code data.shape. There are now 3000 rows and 15 columns in the dataset

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	38.571033	1.896964e+05	10.084200	1076.818167	87.850200	40.471367
std	13.645176	1.055088e+05	2.572586	7412.566535	404.629371	12.388020
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.177670e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.783410e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.372968e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.484705e+06	16.000000	99999.000000	4356.000000	99.000000

data.describe() - This code shows the descriptive statistics of all numerical attributes in the dataset

In [11]:

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

Out[11]:

```
9     9676
10    6706
13    4948
14    1589
11    1269
7     1069
12     986
6      852
4      602
15     532
5      477
8      410
16     382
3      303
2      153
1       46
Name: education-num, dtype: int64
```

The above code shows the occurence of each values in the education_num attribute column sorted from the most to the least frequent in the dataset

In [12]:

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

Out[12]:

```
 HS-grad         9676
 Some-college    6706
 Bachelors       4948
 Masters         1589
 Assoc-voc       1269
 11th            1069
 Assoc-acdm       986
 10th             852
 7th-8th          602
 Prof-school      532
 9th              477
 12th             410
 Doctorate        382
 5th-6th          303
 1st-4th          153
 Preschool         46
Name: education, dtype: int64
```

The above code shows the occurence of each values in the education attribute column sorted from the most to the least frequent in the dataset

In [13]:

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

data=data.drop code is used to drop a column in the dataset. This was used to drop the 'fnlwgt' column which will result to the number of columns reducing from 15 columns from 14 columns

In [14]:

```
data.shape
```

Out[14]:

```
(30000, 14)
```

We run the data.shape code to confirm that the 'fnlwgt' column has been dropped off the dataset

In [15]:

```
data.describe(include='all')
```

Out[15]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race
count	30000.000000	30000	30000	30000.000000	30000	30000	30000	30000
unique	NaN	9	16	NaN	7	15	6	5
top	NaN	Private	HS-grad	NaN	Married-civ-spouse	Prof-specialty	Husband	White
freq	NaN	20941	9676	NaN	13785	3812	12144	25659
mean	38.571033	NaN	NaN	10.084200	NaN	NaN	NaN	NaN
std	13.645176	NaN	NaN	2.572586	NaN	NaN	NaN	NaN
min	17.000000	NaN	NaN	1.000000	NaN	NaN	NaN	NaN
25%	28.000000	NaN	NaN	9.000000	NaN	NaN	NaN	NaN
50%	37.000000	NaN	NaN	10.000000	NaN	NaN	NaN	NaN
75%	48.000000	NaN	NaN	12.000000	NaN	NaN	NaN	NaN
max	90.000000	NaN	NaN	16.000000	NaN	NaN	NaN	NaN

The data.describe(include='all') code is used to show the descriptive statistics of all numerical and categorical data in the dataset.

In [16]:

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

Out[16]:

```
HS-grad         9676
Some-college    6706
Bachelors       4948
Masters         1589
Assoc-voc       1269
11th            1069
Assoc-acdm       986
10th             852
7th-8th          602
Prof-school      532
9th              477
12th             410
Doctorate        382
5th-6th          303
1st-4th          153
Preschool         46
Name: education, dtype: int64
```

The above code shows the occurence of each values in the education attribute column sorted from the most to the least frequent in the dataset

In [17]:

```
data['education'].nunique()
```

Out[17]:

```
16
```

This code returns the count of unique values in the education column. There are 16 unique values in the education column.

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

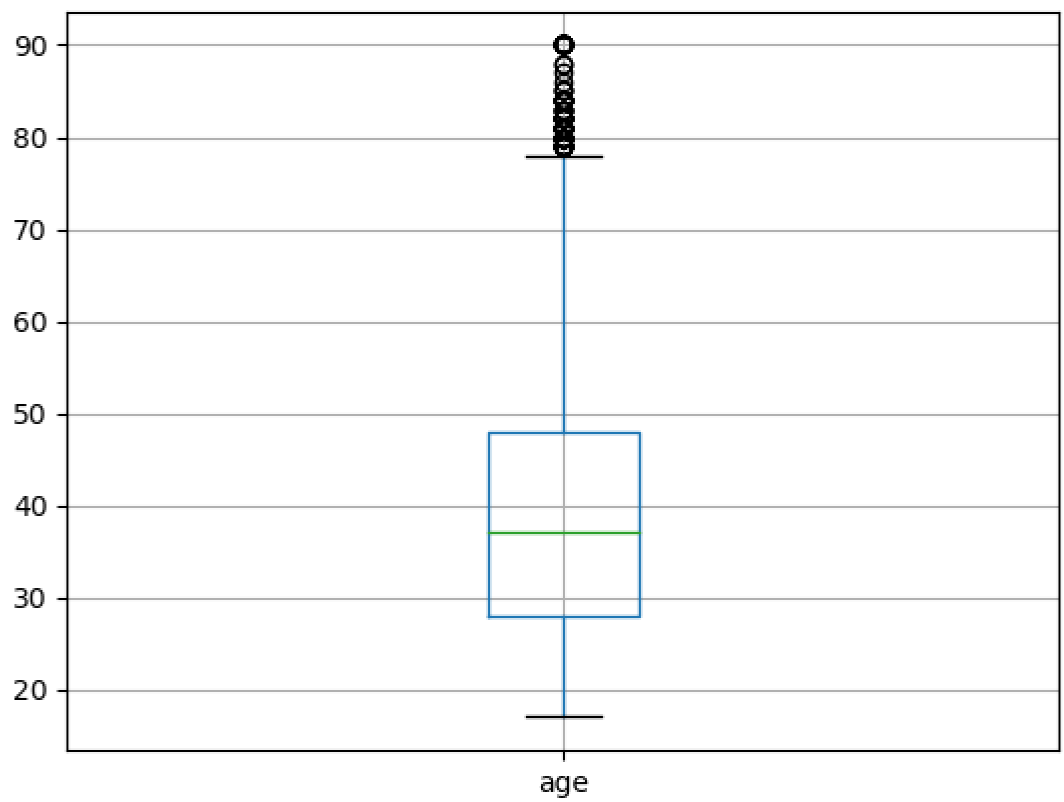
Out[18]:
```
31    828
34    815
33    813
23    812
36    810
     ...
83      5
85      3
88      2
87      1
86      1
Name: age, Length: 73, dtype: int64
```

value_counts isn't an idle way to generate the occurence of each values in the age attribute column. To get a better view of age we will make use of a boxplot graph in visualizing age because age is a continuous value and is too lengthly to be analyzed using value_counts function

Lets visualize age using boxplot instead;

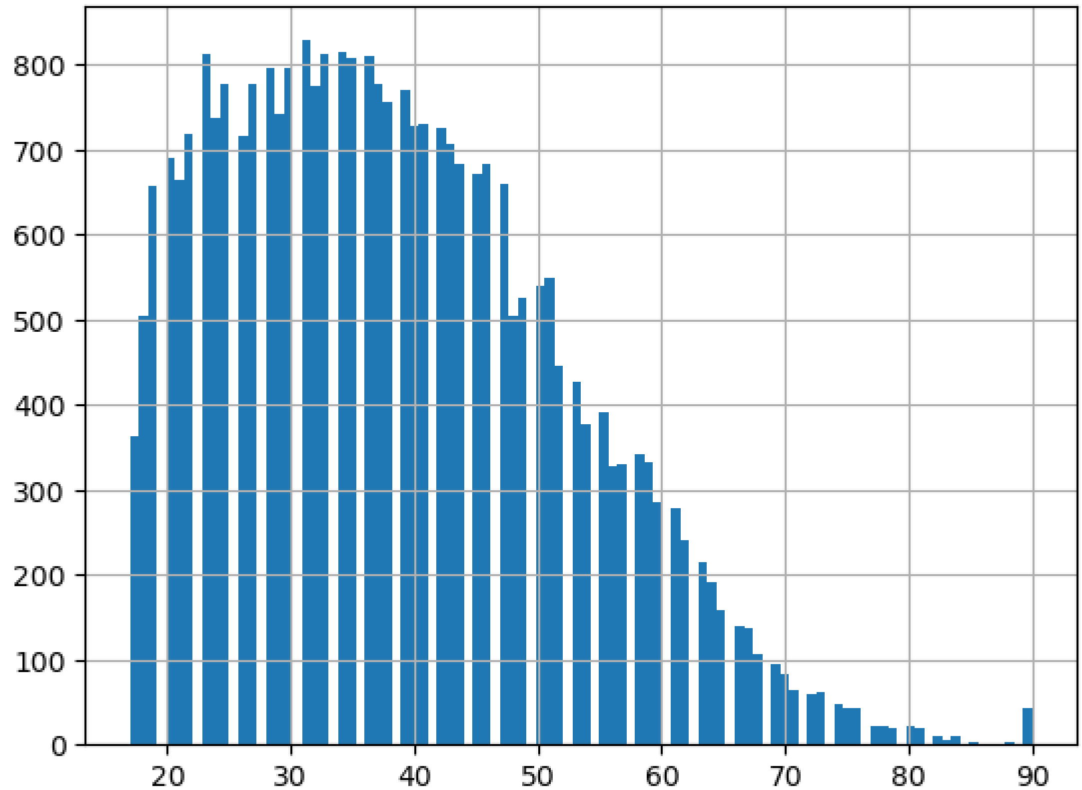
In [19]:
```python
data.boxplot(column='age')
```

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



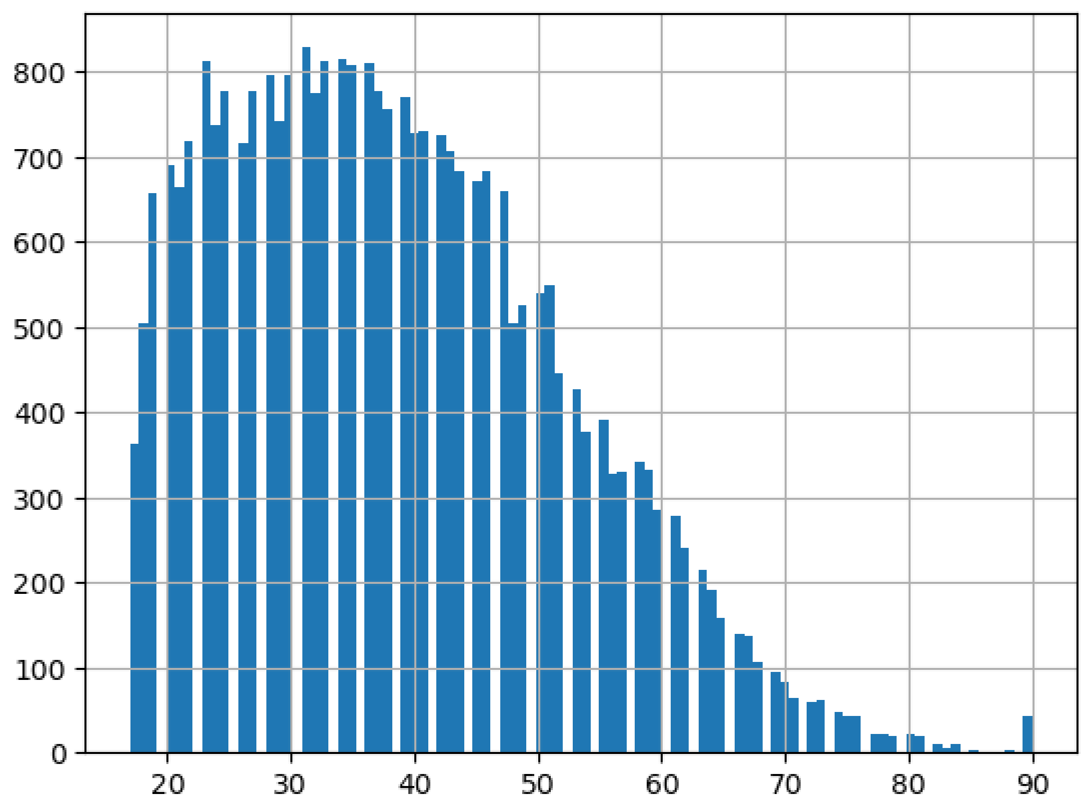
In [20]:
```python
data['age'].hist(bins=100)
```

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



In [21]:
```
data.age.hist(bins=100)
```

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



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

Out[22]:
```
 Male      20082
 Female     9918
Name: sex, dtype: int64
```

value_counts by 'sex' is used to show the number of males and females that exist in the dataset. Here there are 20082 males and 9918 females in the dataset

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             20941
 Self-emp-not-inc     2353
 Local-gov            1920
 ?                    1675
 State-gov            1178
 Self-emp-inc         1028
 Federal-gov           885
 Without-pay            14
 Never-worked            6
Name: workclass, dtype: int64
```

value_counts by 'workclass' is used to show the number of the occurence of each work sector workclasss attribute column sorted from the most to the least frequent in the dataset

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

Out[25]:
```
 Male      20082
 Female     9918
Name: sex, dtype: int64
```

The value_count by sex function above shows there are 20082 males and 9918 females in the dataset

Applying Groupby Functions in Order to Summarise the Data.

Average age of each gender in the given population

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

Out[26]:
```
sex
 Female    36.854104
 Male      39.418982
Name: age, dtype: float64
```

Using groupby aggregate function we can derive the average age of each gender in the dataset.

In the above we grouped by 'sex' and calculated the average of 'age'. Hence the average age for Female is 36.854104 and Male is 39.418982

Average age of male and female across different education categories

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

Out[27]:
```
sex      education
 Female   10th             35.381818
          11th             30.687179
          12th             30.305970
          1st-4th          49.214286
          5th-6th          44.986486
          7th-8th          49.891156
          9th              42.044118
          Assoc-acdm       36.418848
          Assoc-voc        37.805252
          Bachelors        35.671096
          Doctorate        45.012500
          HS-grad          38.633834
          Masters          43.030488
          Preschool        42.933333
          Prof-school      40.011765
          Some-college     33.687621
 Male     10th             38.150780
          11th             33.597938
          12th             33.246377
          1st-4th          45.090090
          5th-6th          41.851528
          7th-8th          48.292308
          9th              40.577713
          Assoc-acdm       37.865894
          Assoc-voc        38.948276
          Bachelors        40.284055
          Doctorate        48.311258
          HS-grad          39.101390
          Masters          44.515041
          Preschool        43.096774
          Prof-school      45.702461
          Some-college     36.995156
Name: age, dtype: float64
```

Using the groupby function we grouped by 'sex' and 'education' and computed the mean for 'age'

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   ?                     317.976864
          Adm-clerical          530.984985
          Craft-repair          815.635922
          Exec-managerial      1020.321731
          Farming-fishing       667.949153
          Handlers-cleaners     141.777070
          Machine-op-inspct     175.299413
          Other-service         165.004225
          Priv-house-serv       313.770992
          Prof-specialty       1212.759857
          Protective-serv      1525.614286
          Sales                 228.426230
          Tech-support          662.320872
          Transport-moving      513.825000
 Male     ?                     824.759690
          Adm-clerical          467.401918
          Armed-Forces            0.000000
          Craft-repair          627.810544
          Exec-managerial      2778.824294
          Farming-fishing       536.919240
          Handlers-cleaners     281.325088
          Machine-op-inspct     387.201800
          Other-service         232.912446
          Priv-house-serv        84.857143
          Prof-specialty       3548.676045
          Protective-serv       539.611006
          Sales                1911.073039
          Tech-support          691.472590
          Transport-moving      514.976412
Name: capital-gain, dtype: float64
```

Using the groupby function we grouped by 'sex' and 'occupation' and computed the mean for 'capital-gain'

Identifying the average capital-gain by males and females across different marital-status

In [29]:
```
data['capital-gain'].groupby([data['sex'],data['marital-status']]).mean()
```

Out[29]:
```
sex      marital-status
 Female   Divorced                  452.100000
          Married-AF-spouse         204.076923
          Married-civ-spouse       1562.743119
          Married-spouse-absent     227.136842
          Never-married             342.128311
          Separated                 194.290657
          Widowed                   476.324573
 Male     Divorced                 1093.146969
          Married-AF-spouse         912.250000
          Married-civ-spouse       1799.035076
          Married-spouse-absent     923.910448
          Never-married             422.352015
          Separated                 789.347107
          Widowed                  1005.943038
Name: capital-gain, dtype: float64
```

Maximum age across different races

In [30]:
```
data['race'].value_counts()
```
Out[30]:
```
 White                 25659
 Black                  2842
 Asian-Pac-Islander      966
 Amer-Indian-Eskimo      290
 Other                   243
Name: race, dtype: int64
```
In [31]:
```
data['age'].groupby([data['race']]).max()
```
Out[31]:
```
race
 Amer-Indian-Eskimo    80
 Asian-Pac-Islander    90
 Black                 90
 Other                 77
 White                 90
Name: age, dtype: int64
```
90 is the maximum age amongst the Asian-Pac-Islander, Black and White races, using the groupby function above.

Are minimum and maximum age by sex same?

Minimum Age by Sex

In [32]:
```
data['age'].groupby([data['sex']]).min()
```
Out[32]:
```
sex
 Female    17
 Male      17
Name: age, dtype: int64
```
Minimum Age by Sex

In [33]:
```
data['age'].groupby([data['sex']]).max()
```
Out[33]:
```
sex
 Female    90
 Male      90
Name: age, dtype: int64
```
Using the groupby function the minimum and maximum age by sex are the same

The minimum age for Male and Female is 17 while the maximum age for Male and Female is 90 as well.

Data Visualization

In [34]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```
In [35]:
```
data.describe()
```

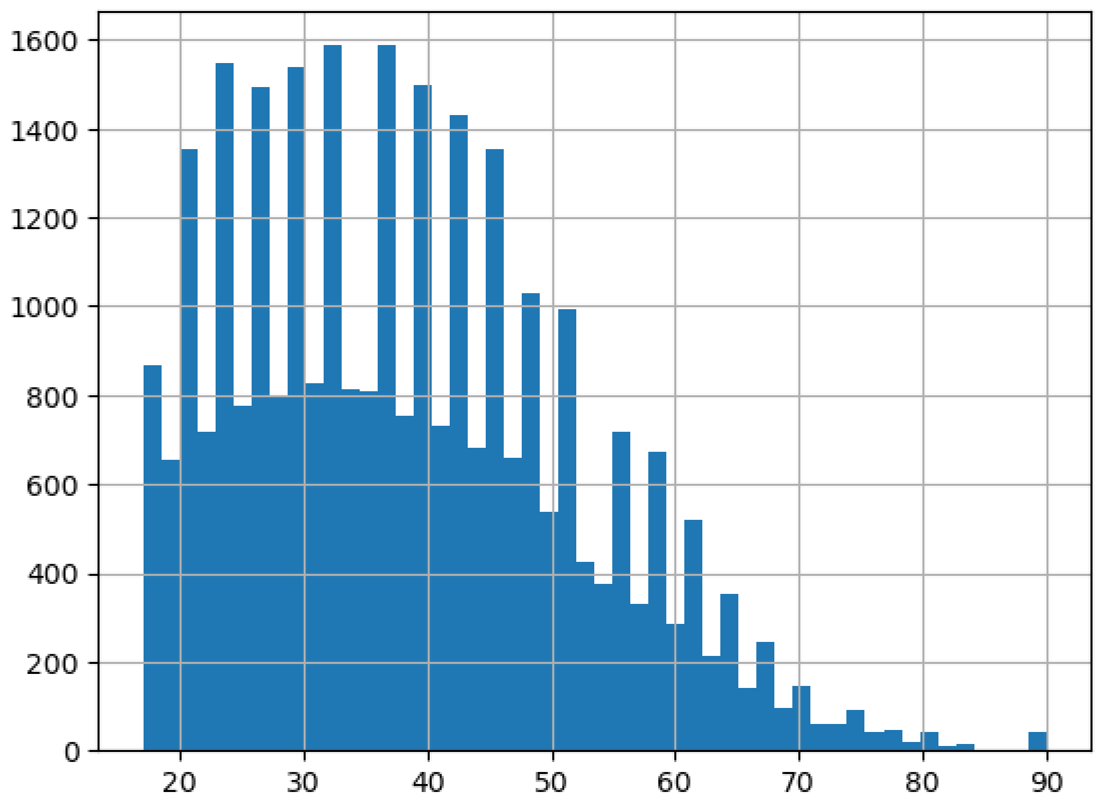
Out[35]:

	age	education-num	capital-gain	capital-loss	hours-per-week
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000
mean	38.571033	10.084200	1076.818167	87.850200	40.471367
std	13.645176	2.572586	7412.566535	404.629371	12.388020
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 [36]:

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

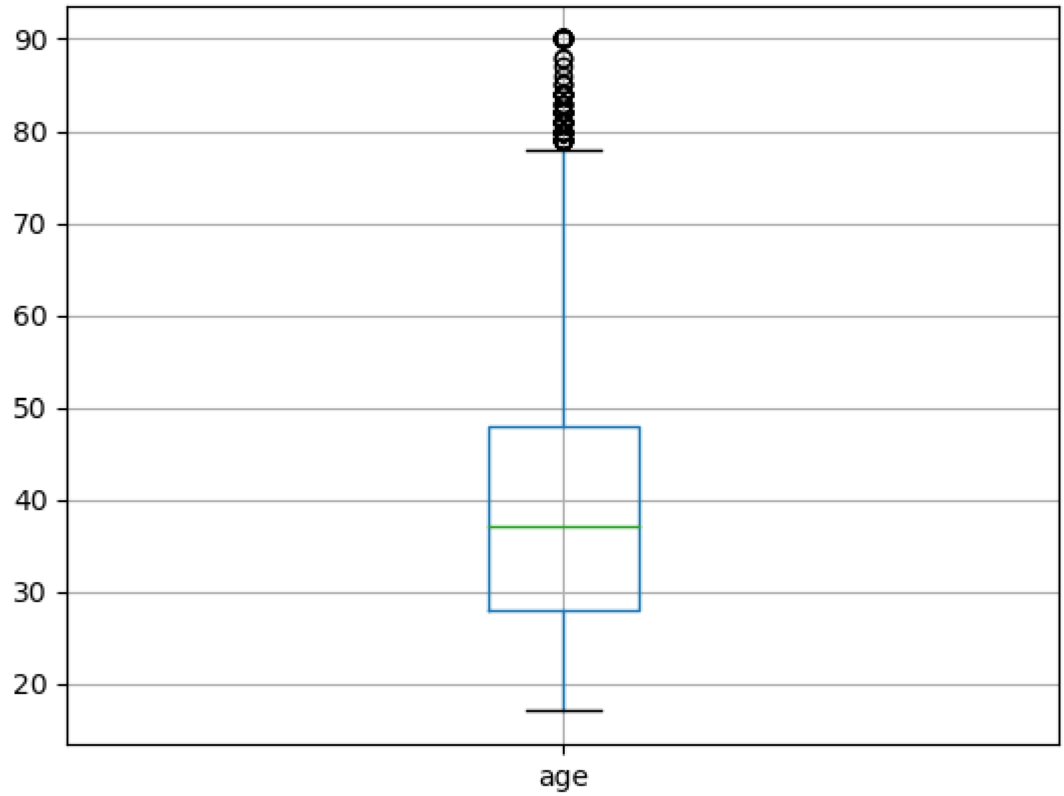
Out[36]: <AxesSubplot:>



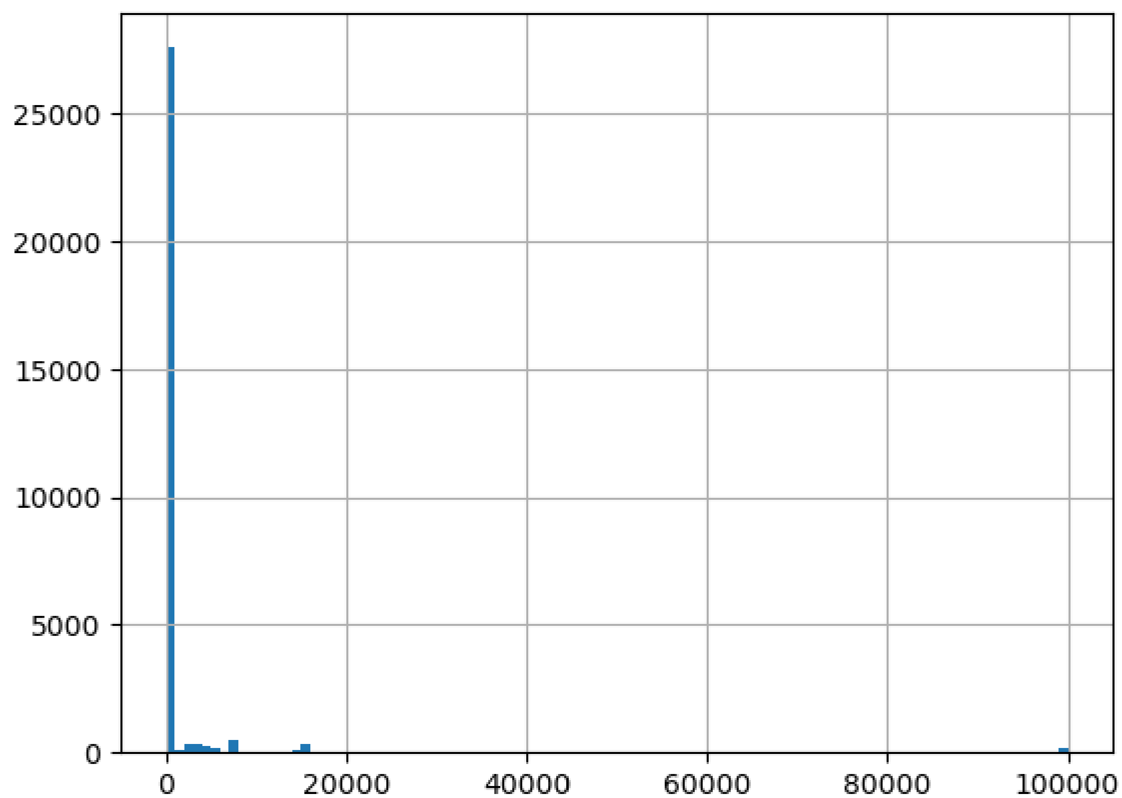
In [37]:

```
data.boxplot(column='age')
```

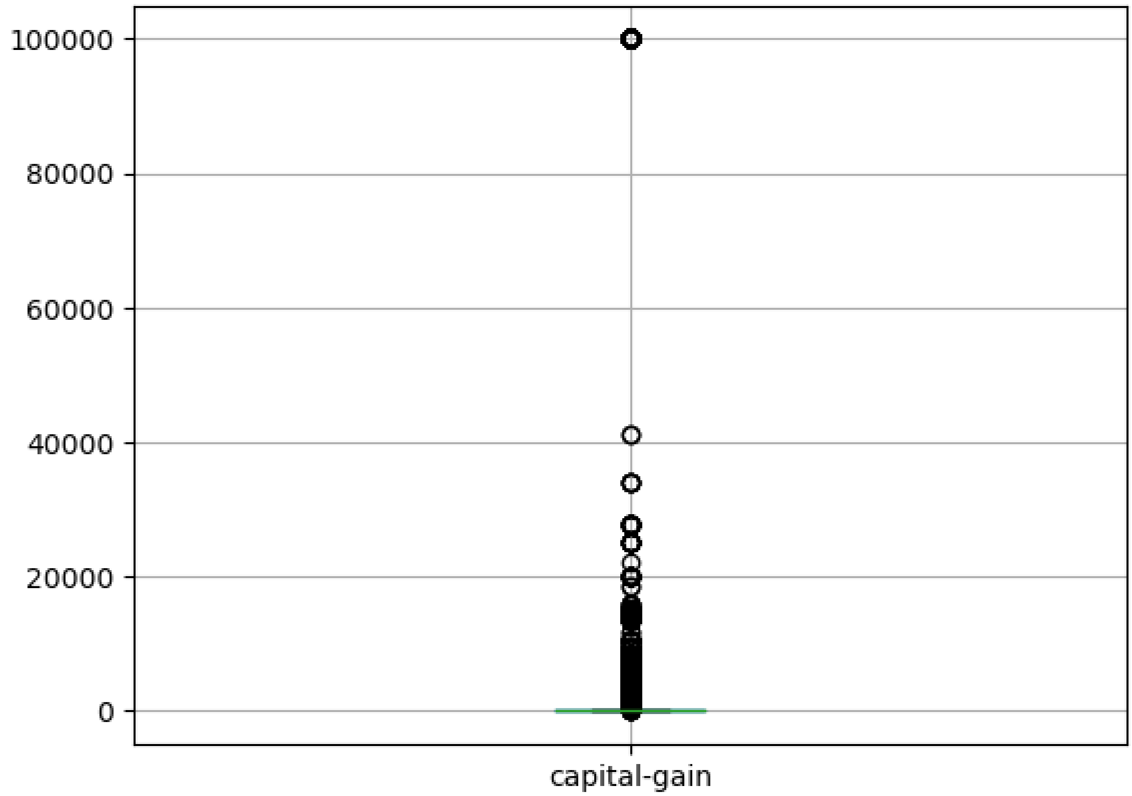
Out[37]: <AxesSubplot:>



In [38]:
```
data['capital-gain'].hist(bins=100)
```
Out[38]: <AxesSubplot:>

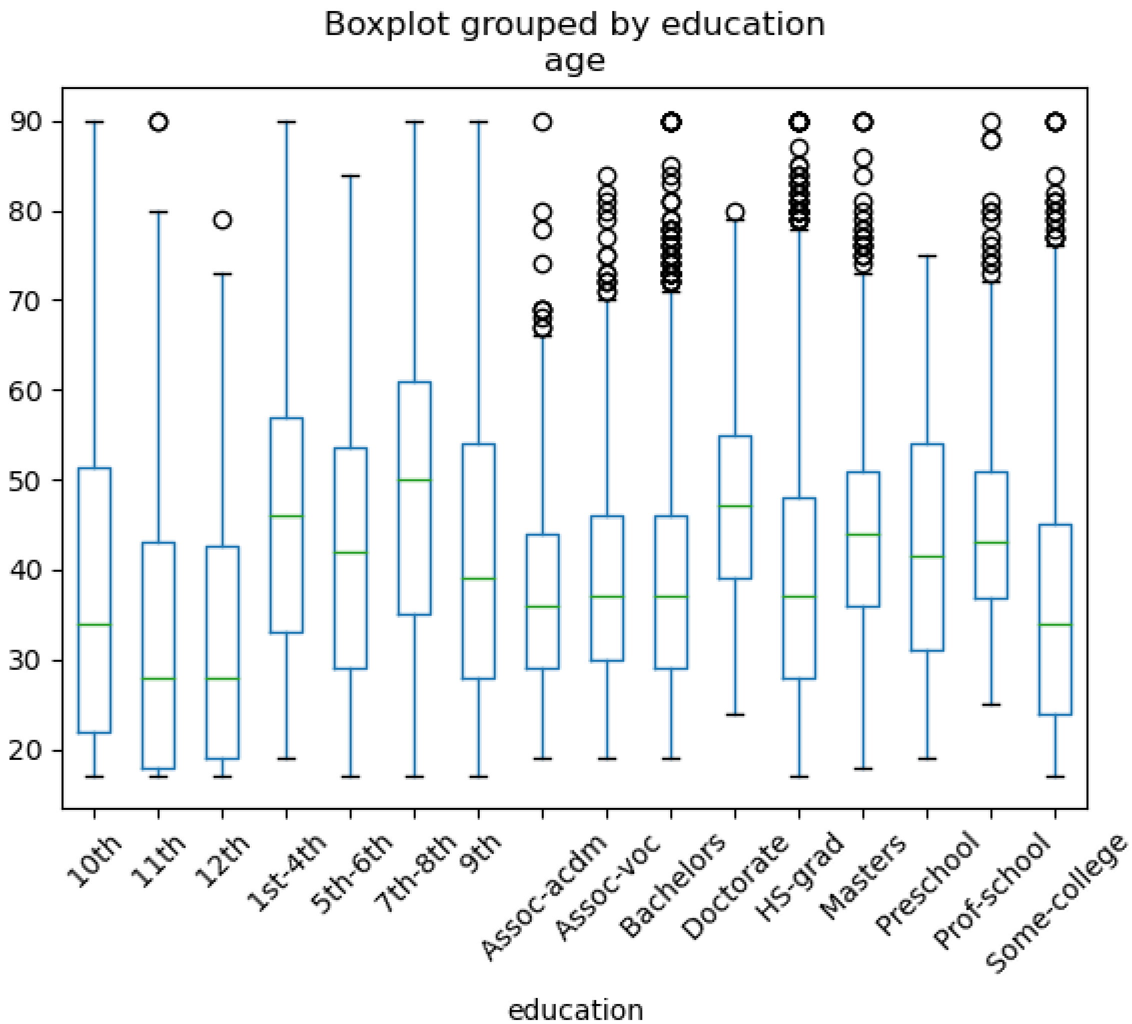


In [39]:
```
data.boxplot(column='capital-gain')
```
Out[39]: <AxesSubplot:>



In [40]:
```
data.boxplot(column='age', by = 'education', grid=False, rot = 45, fontsize = 10)
```

Out[40]:
```
<AxesSubplot:title={'center':'age'}, xlabel='education'>
```

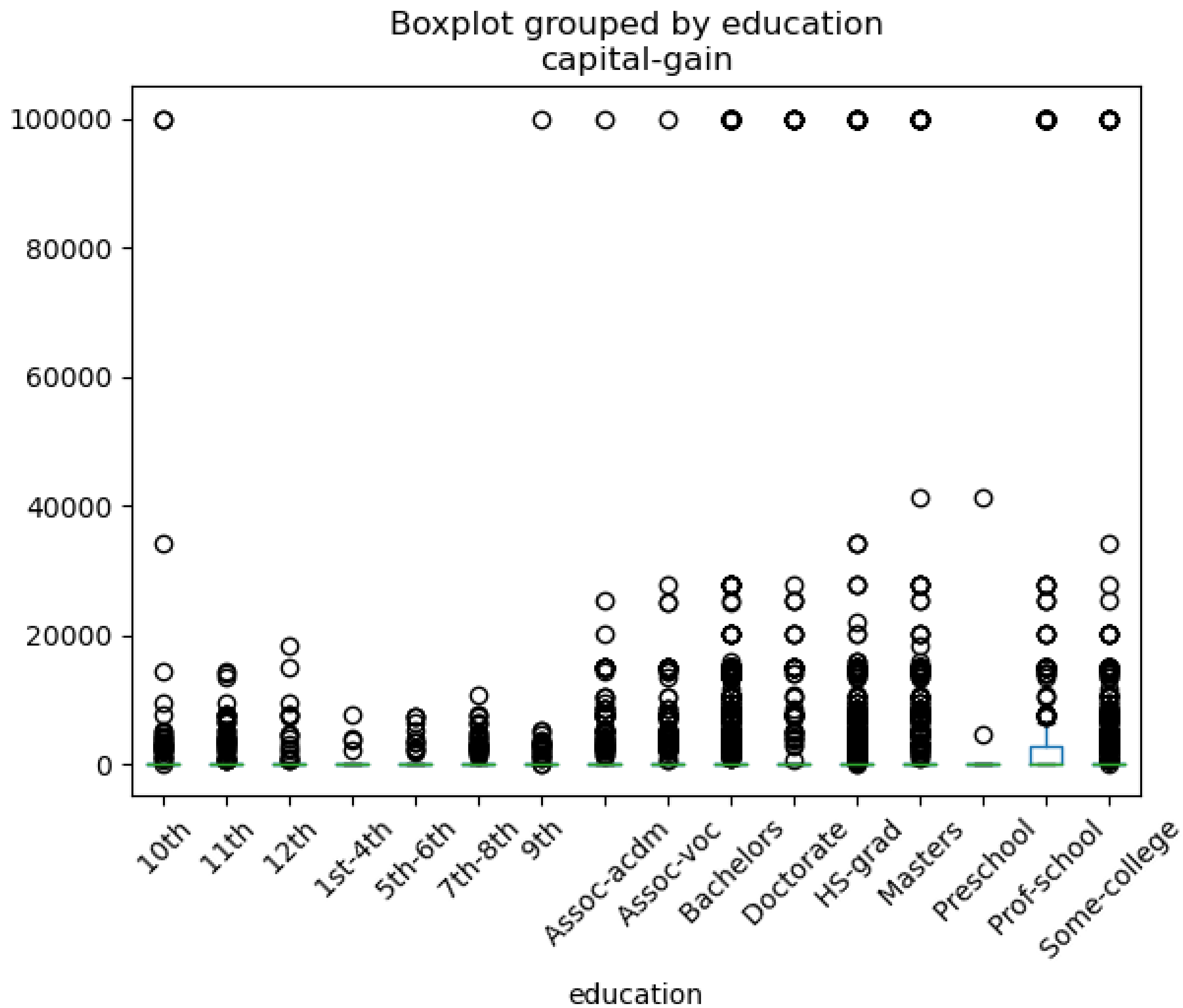


In [41]:
```
data['education'].value_counts()
```

Out[41]:
```
HS-grad         9676
Some-college    6706
Bachelors       4948
Masters         1589
Assoc-voc       1269
11th            1069
Assoc-acdm       986
10th             852
7th-8th          602
Prof-school      532
9th              477
12th             410
Doctorate        382
5th-6th          303
1st-4th          153
Preschool         46
Name: education, dtype: int64
```

In [42]:
```
data.boxplot(column='capital-gain', by = 'education', grid=False, rot = 45, fontsize =
```

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



In [43]:
```
data['marital-status'].value_counts()
```

Out[43]:
```
Married-civ-spouse       13785
Never-married             9840
Divorced                  4103
Separated                  941
Widowed                    919
Married-spouse-absent      391
Married-AF-spouse           21
Name: marital-status, dtype: int64
```

In [44]:
```
data.apply(lambda x: sum(x.isnull()), axis = 0)
```

Out[44]:
```
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
```

Data Transformation

In [45]:
```
from sklearn.preprocessing import LabelEncoder
```

In [46]:
```
data.head()
```

Out[46]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	ca
17121	54	Private	Some-college	10	Married-civ-spouse	Transport-moving	Husband	White	Male	
363	43	Private	Bachelors	13	Divorced	Exec-managerial	Not-in-family	White	Male	
17291	42	Self-emp-not-inc	HS-grad	9	Married-civ-spouse	Craft-repair	Husband	White	Male	
16063	65	?	HS-grad	9	Married-civ-spouse	?	Husband	White	Male	
25565	34	State-gov	Some-college	10	Married-spouse-absent	Adm-clerical	Unmarried	Asian-Pac-Islander	Female	

In [47]:
```
data.dtypes
```

Out[47]:
```
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 [48]:
```
columns = list(data.select_dtypes(exclude=['int64']))
```

In [49]:
```
columns
```

Out[49]:
```
['workclass',
 'education',
 'marital-status',
 'occupation',
 'relationship',
 'race',
 'sex',
 'native-country',
 'class-label']
```

In [50]:
```
data['class-label'].value_counts()
```

Out[50]:
```
 <=50K    22781
 >50K      7219
Name: class-label, dtype: int64
```

In [51]:
```
le = LabelEncoder()
for i in columns:
    #print(i)
    data[i] = le.fit_transform(data[i])
data.dtypes
```

Out[51]:
```
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 [52]:
```
data.head()
```

Out[52]:

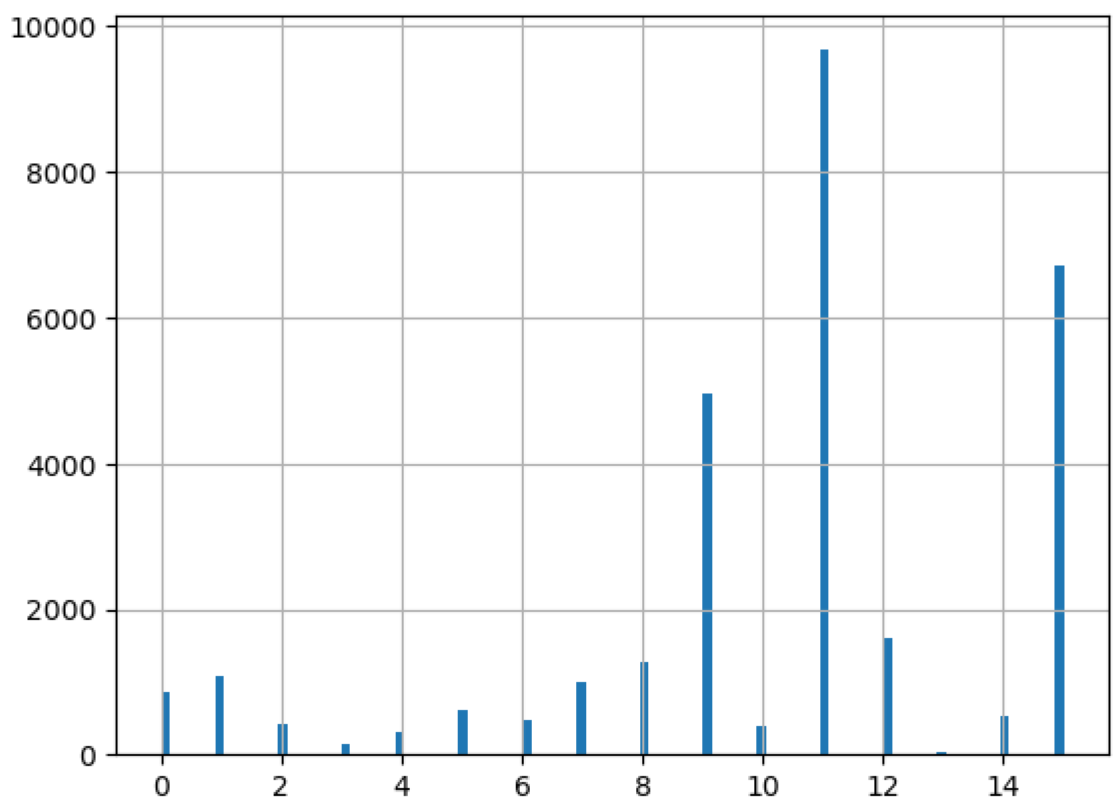
	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain
17121	54	4	15	10	2	14	0	4	1	0
363	43	4	9	13	0	4	1	4	1	0
17291	42	6	11	9	2	3	0	4	1	0
16063	65	0	11	9	2	0	0	4	1	0
25565	34	7	15	10	3	1	4	1	0	0

In [53]:
```
data['workclass'].value_counts()
```

Out[53]:
```
4    20941
6     2353
2     1920
0     1675
7     1178
5     1028
1      885
8       14
3        6
Name: workclass, dtype: int64
```

In [54]:
```
data['education'].hist(bins=100)
```

Out[54]:
```
<AxesSubplot:>
```



In [55]:
```
data.describe(include='all')
```

Out[55]:

	age	workclass	education	education-num	marital-status	occupation	relations
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000
mean	38.571033	3.870767	10.299267	10.084200	2.611433	6.581300	1.448
std	13.645176	1.451644	3.864152	2.572586	1.507136	4.227661	1.607
min	17.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0.000
25%	28.000000	4.000000	9.000000	9.000000	2.000000	3.000000	0.000
50%	37.000000	4.000000	11.000000	10.000000	2.000000	7.000000	1.000
75%	48.000000	4.000000	12.000000	12.000000	4.000000	10.000000	3.000
max	90.000000	8.000000	15.000000	16.000000	6.000000	14.000000	5.000

Report

Summary of the outcome of data.describe()

In [56]:

```
data = pd.read_csv(r'C:\Users\LenovoX260\Desktop\Data Minning and Informatics Assignme
```

In [57]:

```
data.describe()
```

Out[57]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week
count	32561.000000	3.256100e+04	32561.000000	32561.000000	32561.000000	32561.000000
mean	38.581647	1.897784e+05	10.080679	1077.648844	87.303830	40.437456
std	13.640433	1.055500e+05	2.572720	7385.292085	402.960219	12.347429
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.178270e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.783560e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.370510e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.484705e+06	16.000000	99999.000000	4356.000000	99.000000

The outcome of data.describe() function used here shows the descriptive statistics of all numerical attributes in the dataset

It shows the count, mean, standard deviation, minimum, first quartile, second quartile, third quartile and maximum value of age, education-num, capital-gain and hours-per-week attributes in the dataset.

The different data types (or attribute types) in data mining.

From the Adult dataset we have two types of attributes

1. Categorical or Qualitative Attribue.
2. Numerical or Quantitative Attribute.

Categorical or Qualitative Attributes -

These attribute takes qualitative values with qualitative characteristics. They are classified into two types;

- Nominal Attributes - This type of attribite provides enough information to differentiate between one object from another. Examples are; customer ID, student ID, zip codes, employee ID, gender, sex etc.
- Ordinal Attribute: The ordinal attribute value provides sufficient information to order the objects such as rankings, grades, street numbers, height etc

From the Adult dataset attributes like workclass, education, marital-status,occupation, relationship,race, sex and native-country are all classfied as Categorical or Qualitative attributes.

Numerical or Quantitative Attributes -

This are data that contains whole numbers and decimals and have most properties of numbers. T Numerical attributes are further divided into;

- Binary Attribute: These are 0 and 1. Where 0 is the absence of any features and 1 is the inclusion of any characteristics. Numeric attribute:It is quantitative, such that quantity can be measured and represented in integer or real values ,are of two types
- Interval Scaled attribute: It is measured on a scale of equal size units,these attributes allow us to compare values such as calendar dates temperature in Celsius or Fahrenheit.
- Ratio Scaled attribute: For ratio both differences and ratios are significant. For example, age, counts, temperature in Kelvin, length, and Weight.

From the Adult dataset we can see some Numerical or Quantitative attributes like age, education number, capital gain, capital loss and hours per week.

Country with the Highest Migrants.

Using the code below we can see that United-States have the largest number of migrants but in our answer we can't say the United-States have the largest migrant because the data was derived in the United States and the citizens of the United-States were included in the data which we obviously can't ignore in our analysis cause they are part of the dataset. Hence, in this case we would pick the country with the second largest number, people who are actually migrants in the United-States. Therefore, Mexico is the country with the largest number with 643 migrants after the United States.

In [58]:
```
data = pd.read_csv(r'C:\Users\LenovoX260\Desktop\Data Minning and Informatics Assignme
```

In [61]:
```
a=data["native-country"].value_counts()
a.head(2)
```

Out[61]:
```
 United-States    29170
 Mexico             643
Name: native-country, dtype: int64
```

Occupation that represents more males than females

To get this we will apply the code below.

In [62]:
```
data['sex'].groupby([data['occupation']]).value_counts()
```

Out[62]:
```
occupation          sex
 ?                   Male      1002
                     Female     841
 Adm-clerical        Female    2537
                     Male      1233
 Armed-Forces        Male         9
 Craft-repair        Male      3877
                     Female     222
 Exec-managerial     Male      2907
                     Female    1159
 Farming-fishing     Male       929
                     Female      65
 Handlers-cleaners   Male      1206
                     Female     164
 Machine-op-inspct   Male      1452
                     Female     550
 Other-service       Female    1800
                     Male      1495
 Priv-house-serv     Female     141
                     Male         8
 Prof-specialty      Male      2625
                     Female    1515
 Protective-serv     Male       573
                     Female      76
 Sales               Male      2387
                     Female    1263
 Tech-support        Male       580
                     Female     348
 Transport-moving    Male      1507
                     Female      90
Name: sex, dtype: int64
```

From the code above, we can see there are 12 occupation that consist of more male than female. Occupation such as;

1. Craft repair - which has 3877 males than females.
2. Exec-managerial - consist of 2907 males and less females.
3. Farming-fishing - with 929 more males than females.
4. Handlers-cleaners - with 1206 males and females.
5. Machine-op-inspct - which has 1452 more males than females.
6. Prof-specialty - with 2625 more males than females.
7. Protective-serv - Has 573 more males than females.

8. Sales - Has 2387 more males than females.
9. Tech-support - Has 580 more males than females.
10. Transport-moving - Has more males of 1507 than females.

Finally, lets not forget the unidentified occupation and the armed-forces in the dataset contains more males and females. With the unidentified occupation having 1002 males and the armed forces having only just males with no females.

Difference between data.head() and data.tail()

In [63]:
```
data.head()
```

Out[63]:

	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
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female

data.head() - This function is used to display the first 5 rows in the data set from 0-4

In [64]:
```
data.tail()
```

Out[64]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	
32556	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	White	Fer
32557	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	M
32558	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarried	White	Fer
32559	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	White	M
32560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Fer

data.tail() - This code is used to display the last 5 rows in the data set.