Exploratory analysis: Loading and exploration of dataset

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.csv')
```

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

Out[3]:

	Age	Worklass	Fnlwgt	Education	Education-num	Marital-Status	Occupation	Relationship	Race	Sex	Capital-gain	Capital-loss	Hours-per-week	Native-country	Class-label
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50K

Q1 answer for head(2), head(10), tail(2)

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

Out[4]:

	Age	Worklass	Fnlwgt	Education	Education-num	Marital-Status	Occupation	Relationship	Race	Sex	Capital-gain	Capital-loss	Hours-per-week	Native-country	Class-label
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K

The code written above has displayed the first two rows of the dataframe

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

Out[5]:

	Age	Worklass	Fnlwgt	Education	Education-num	Marital-Status	Occupation	Relationship	Race	Sex	Capital-gain	Capital-loss	Hours-per-week	Native-country	Class-label
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50K
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Female	0	0	40	United-States	<=50K
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Female	0	0	16	Jamaica	<=50K
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	45	United-States	>50K
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Female	14084	0	50	United-States	>50K
9	42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	5178	0	40	United-States	>50K

The code that has been typed above shows the frist 10 rows of the dataframe

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

Out[6]:

	Age	Worklass	Fnlwgt	Education	Education-num	Marital-Status	Occupation	Relationship	Race	Sex	Capital-gain	Capital-loss	Hours-per-week	Native-country	Clas lab
32559	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	White	Male	0	0	20	United-States	<=5(
32560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Female	15024	0	40	United-States	>5(

The code typed above shows a display of the last two rows of the whole dataframe

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

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

Generation of my unique dataset

In [8]:
```
data = data.sample(n=30000, random_state = 71 )
```

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	38.542267	1.899139e+05	10.082167	1061.891633	87.759367	40.416200
std	13.643028	1.056092e+05	2.570545	7224.973199	404.341587	12.348471
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.179040e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.786055e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.375492e+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     9716
10    6717
13    4924
14    1582
11    1269
7     1076
12     984
6      856
4      585
15     539
5      474
8      388
16     380
3      305
2      157
1       48
Name: Education-num, dtype: int64
```

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

Out[12]:
```
 HS-grad         9716
 Some-college    6717
 Bachelors       4924
 Masters         1582
 Assoc-voc       1269
 11th            1076
 Assoc-acdm       984
 10th             856
 7th-8th          585
 Prof-school      539
 9th              474
 12th             388
 Doctorate        380
 5th-6th          305
 1st-4th          157
 Preschool         48
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	Worklass	Education	Education-num	Marital-Status	Occupation	Relationship	Race	Sex	Capital-gain	Capital-loss	Hours
count	30000.000000	30000	30000	30000.000000	30000	30000	30000	30000	30000	30000.000000	30000.000000	30000.00
unique	NaN	9	16	NaN	7	15	6	5	2	NaN	NaN	
top	NaN	Private	HS-grad	NaN	Married-civ-spouse	Prof-specialty	Husband	White	Male	NaN	NaN	
freq	NaN	20890	9716	NaN	13791	3823	12160	25634	20088	NaN	NaN	
mean	38.542267	NaN	NaN	10.082167	NaN	NaN	NaN	NaN	NaN	1061.891633	87.759367	40.41
std	13.643028	NaN	NaN	2.570545	NaN	NaN	NaN	NaN	NaN	7224.973199	404.341587	12.34
min	17.000000	NaN	NaN	1.000000	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000	1.00
25%	28.000000	NaN	NaN	9.000000	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000	40.00
50%	37.000000	NaN	NaN	10.000000	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000	40.00
75%	48.000000	NaN	NaN	12.000000	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000	45.00
max	90.000000	NaN	NaN	16.000000	NaN	NaN	NaN	NaN	NaN	99999.000000	4356.000000	99.00

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

Out[16]:
```
 HS-grad         9716
 Some-college    6717
 Bachelors       4924
 Masters         1582
 Assoc-voc       1269
 11th            1076
 Assoc-acdm       984
 10th             856
 7th-8th          585
 Prof-school      539
 9th              474
 12th             388
 Doctorate        380
 5th-6th          305
 1st-4th          157
 Preschool         48
Name: Education, dtype: int64
```

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

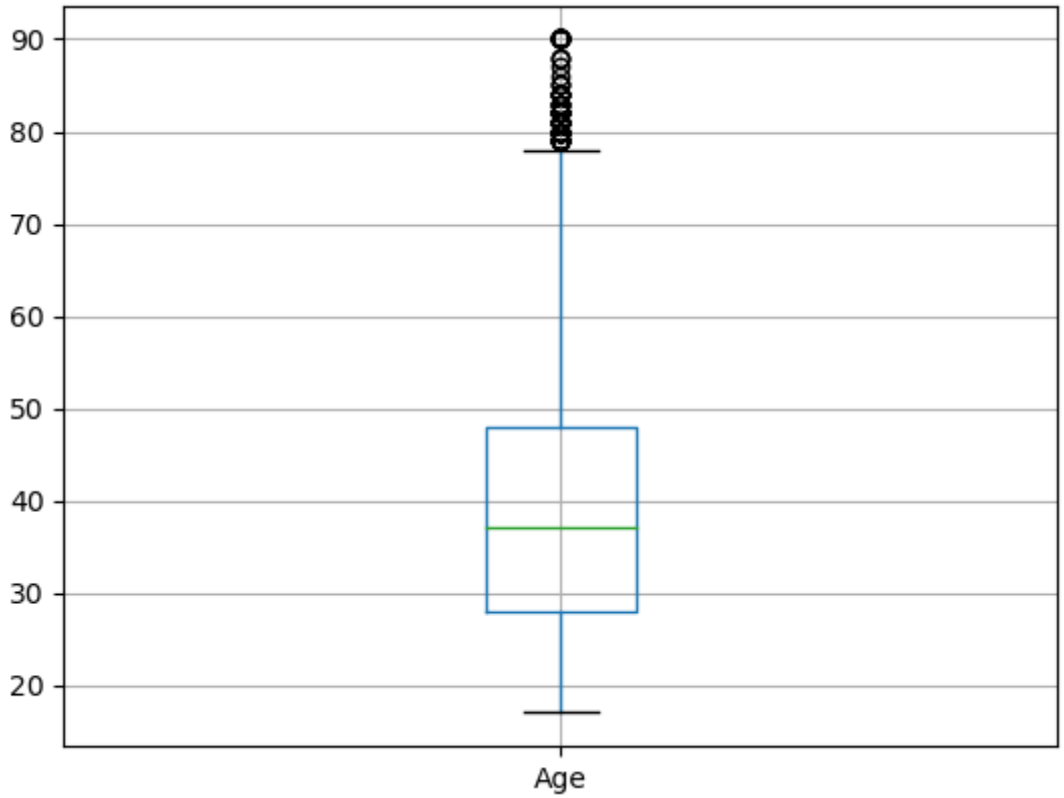
Out[17]:
```
16
```

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

Out[18]:
```
36    826
33    820
34    820
31    816
23    813
     ...
83      6
88      3
85      3
86      1
87      1
Name: Age, Length: 73, dtype: int64
```

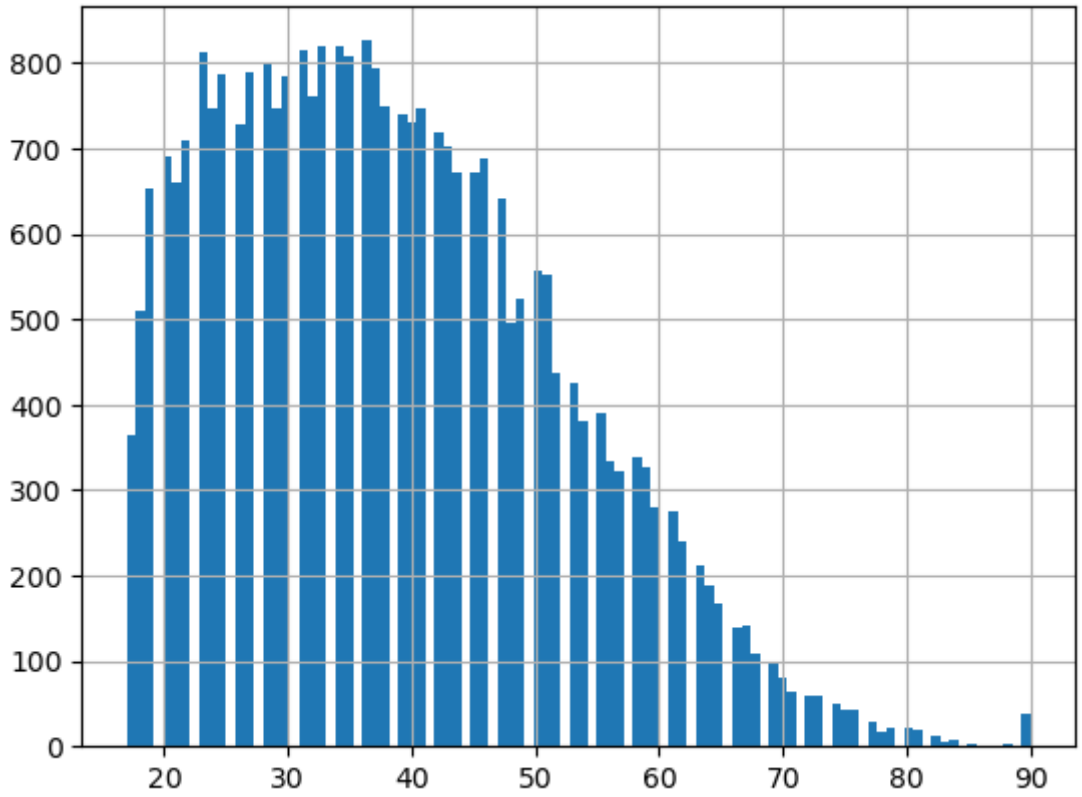
In [19]:
```
data.boxplot(column='Age')
```

Out[19]: <AxesSubplot:>



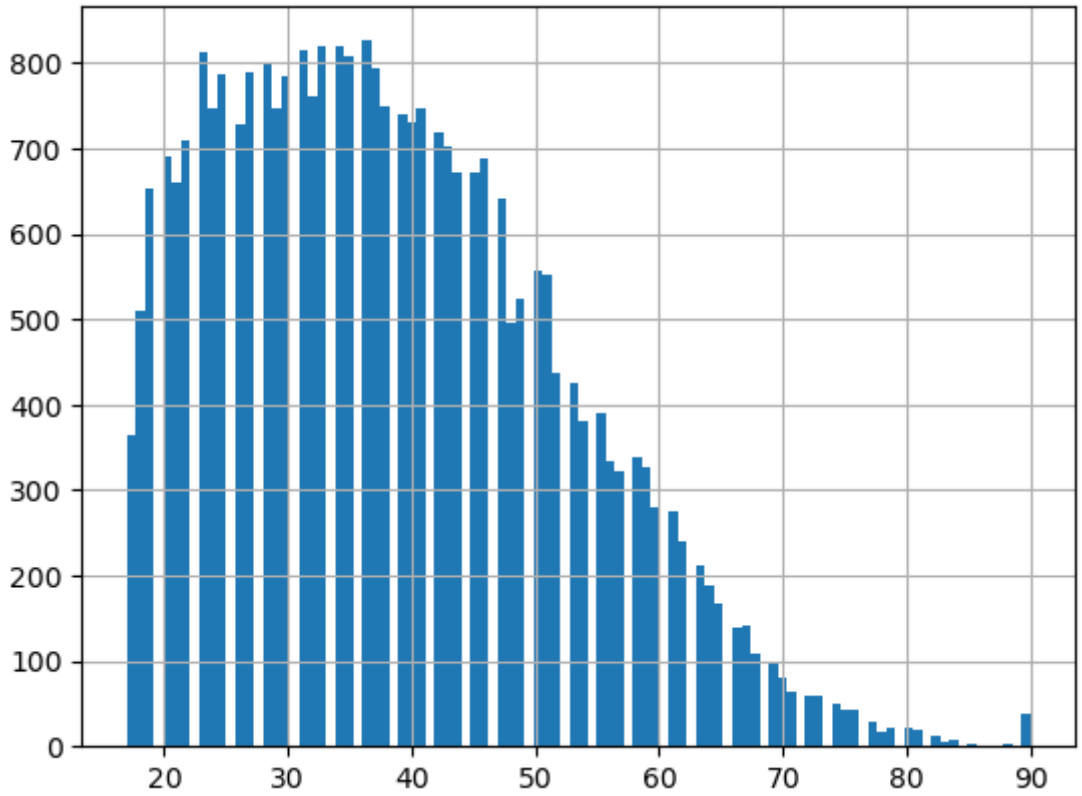
In [20]:
```
data['Age'].hist(bins=100)
```

Out[20]: <AxesSubplot:>



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

Out[21]: <AxesSubplot:>



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

Out[22]:
```
Male      20088
Female     9912
Name: Sex, dtype: int64
```

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

Out[23]:
```
Index(['Age', 'Worklass', '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['Worklass'].value_counts()
```

Out[24]:
```
 Private             20890
 Self-emp-not-inc     2351
 Local-gov            1916
 ?                    1699
 State-gov            1209
 Self-emp-inc         1034
 Federal-gov           885
 Without-pay            10
 Never-worked            6
Name: Worklass, dtype: int64
```

Q2 Answer

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

Out[25]:
```
 Male      20088
 Female     9912
Name: Sex, dtype: int64
```

The number of males in the dataset is equal to 20088 where as the number of females in the dataset is 9912

Application of groupby functions in order to summarise the data

In [26]:
```
data['Age'].groupby([data['Sex']]).mean()
```

Out[26]:
```
Sex
 Female    36.801150
 Male      39.401384
Name: Age, dtype: float64
```

In [27]:
```
data['Age'].groupby([data['Sex'],data['Education']]).mean()
```

Out[27]:
```
Sex      Education
 Female   10th            35.485075
          11th            29.967254
          12th            29.423077
          1st-4th         48.159091
          5th-6th         44.584416
          7th-8th         49.324503
          9th             42.029851
          Assoc-acdm      36.444730
          Assoc-voc       38.086207
          Bachelors       35.632502
          Doctorate       45.637500
          HS-grad         38.679475
          Masters         42.975309
          Preschool       38.571429
          Prof-school     39.788235
          Some-college    33.614998
 Male     10th            38.285714
          11th            33.427099
          12th            32.674419
          1st-4th         45.601770
          5th-6th         42.631579
          7th-8th         48.142857
          9th             40.891176
          Assoc-acdm      38.077311
          Assoc-voc       38.967702
          Bachelors       40.254287
          Doctorate       48.466667
          HS-grad         39.097224
          Masters         44.330292
          Preschool       42.588235
          Prof-school     45.678414
          Some-college    36.940920
Name: Age, dtype: float64
```

Q3 answer

In [28]:
```
data['Capital-gain'].groupby([data['Sex'],data['Occupation']]).mean()
```

Out[28]:
```
Sex      Occupation
 Female   ?                   303.177003
          Adm-clerical        484.564070
          Craft-repair        720.652381
          Exec-managerial    1000.279886
          Farming-fishing    1120.616667
          Handlers-cleaners   100.108108
          Machine-op-inspct   172.676587
          Other-service        94.999398
          Priv-house-serv     316.184615
          Prof-specialty     1352.516919
          Protective-serv     385.579710
          Sales               255.299739
          Tech-support        376.154545
          Transport-moving    440.976190
 Male     ?                   859.747583
          Adm-clerical        493.827586
          Armed-Forces          0.000000
          Craft-repair        653.294118
          Exec-managerial    2655.741264
          Farming-fishing     561.351852
          Handlers-cleaners   192.829710
          Machine-op-inspct   322.710112
          Other-service       257.845421
          Priv-house-serv      99.000000
          Prof-specialty     3523.470419
          Protective-serv     608.423372
          Sales              1940.457635
          Tech-support        711.582857
          Transport-moving    489.793696
Name: Capital-gain, dtype: float64
```

Q4 answer

In [29]:
```
data['Capital-gain'].groupby([data['Sex'],data['Marital-Status']]).mean()
```

Out[29]:
```
Sex      Marital-Status
 Female   Divorced                 414.178060
          Married-AF-spouse        204.076923
          Married-civ-spouse      1467.480896
          Married-spouse-absent    323.659459
          Never-married            320.442638
          Separated                367.610052
          Widowed                  499.541444
 Male     Divorced                1225.074436
          Married-AF-spouse        810.888889
          Married-civ-spouse      1763.432820
          Married-spouse-absent   1022.584615
          Never-married            412.297416
          Separated                910.195592
          Widowed                  740.394737
Name: Capital-gain, dtype: float64
```

In [30]:
```
data['Race'].value_counts()
```

Out[30]:
```
 White                 25634
 Black                  2885
 Asian-Pac-Islander      951
 Amer-Indian-Eskimo      284
 Other                   246
Name: Race, dtype: int64
```

In [31]:
```
data['Age'].groupby([data['Race']]).max()
```

Out[31]:
```
Race
 Amer-Indian-Eskimo    82
 Asian-Pac-Islander    90
 Black                 90
 Other                 77
 White                 90
Name: Age, dtype: int64
```

Q5 answer

In [32]:
```
data['Age'].groupby([data['Sex']]).min()
```

Out[32]:
```
Sex
 Female    17
 Male      17
Name: Age, dtype: int64
```

In [33]:
```
data['Age'].groupby([data['Sex']]).max()
```

Out[33]:
```
Sex
 Female    90
 Male      90
Name: Age, dtype: int64
```

From the data generated from the codes above it can be concluded that minimum and maximum age by sex is the same

Data visualisation

In [34]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

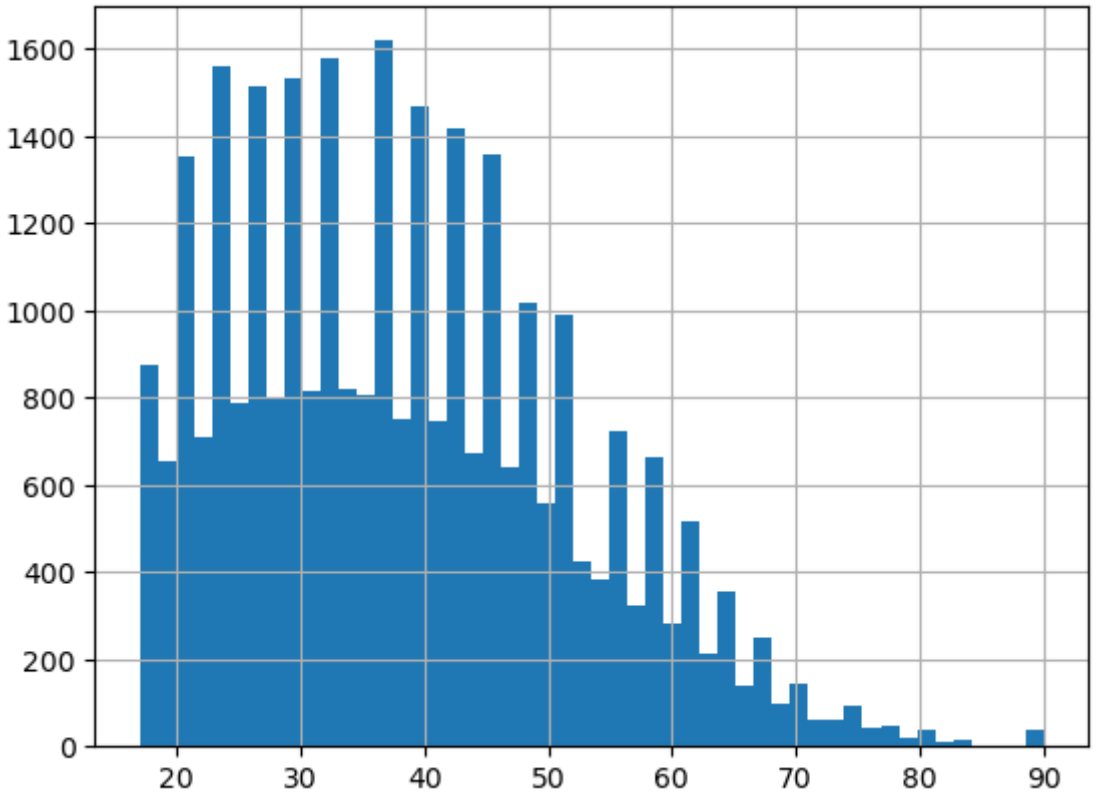
In [35]:
```
data.describe()
```

Out[35]:

	Age	Education-num	Capital-gain	Capital-loss	Hours-per-week
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000
mean	38.542267	10.082167	1061.891633	87.759367	40.416200
std	13.643028	2.570545	7224.973199	404.341587	12.348471
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)
```

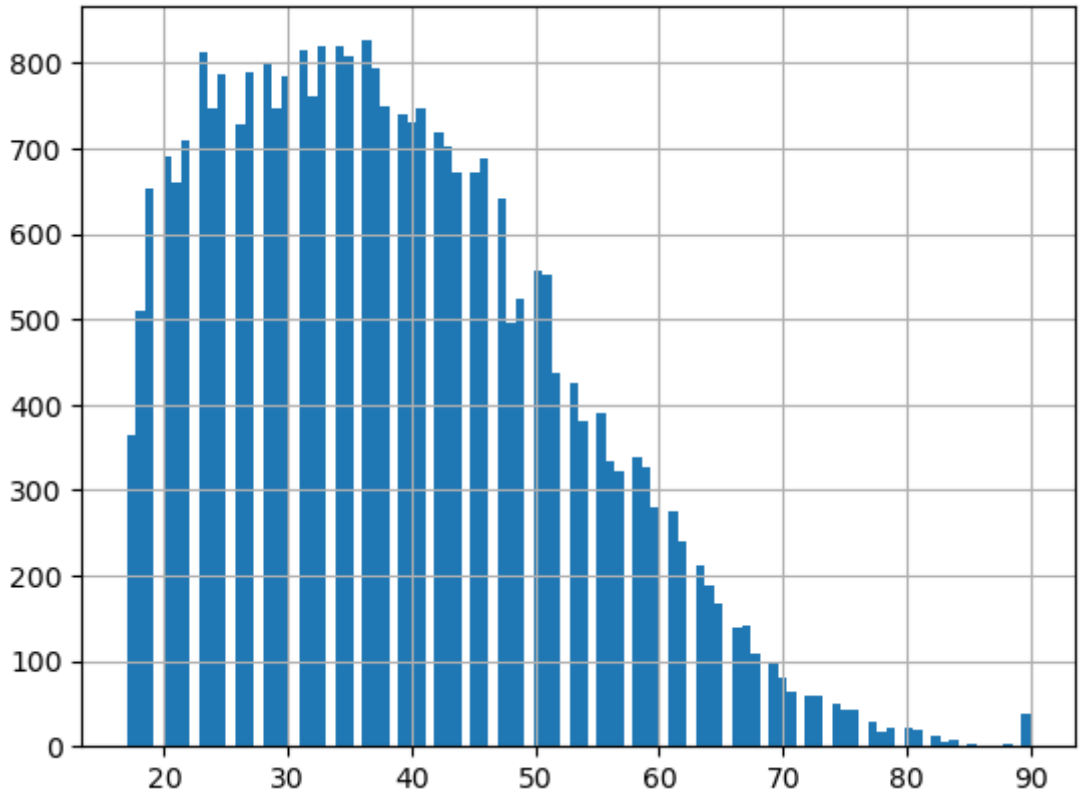
Out[36]: <AxesSubplot:>



The 'try it yourself' exercise answer

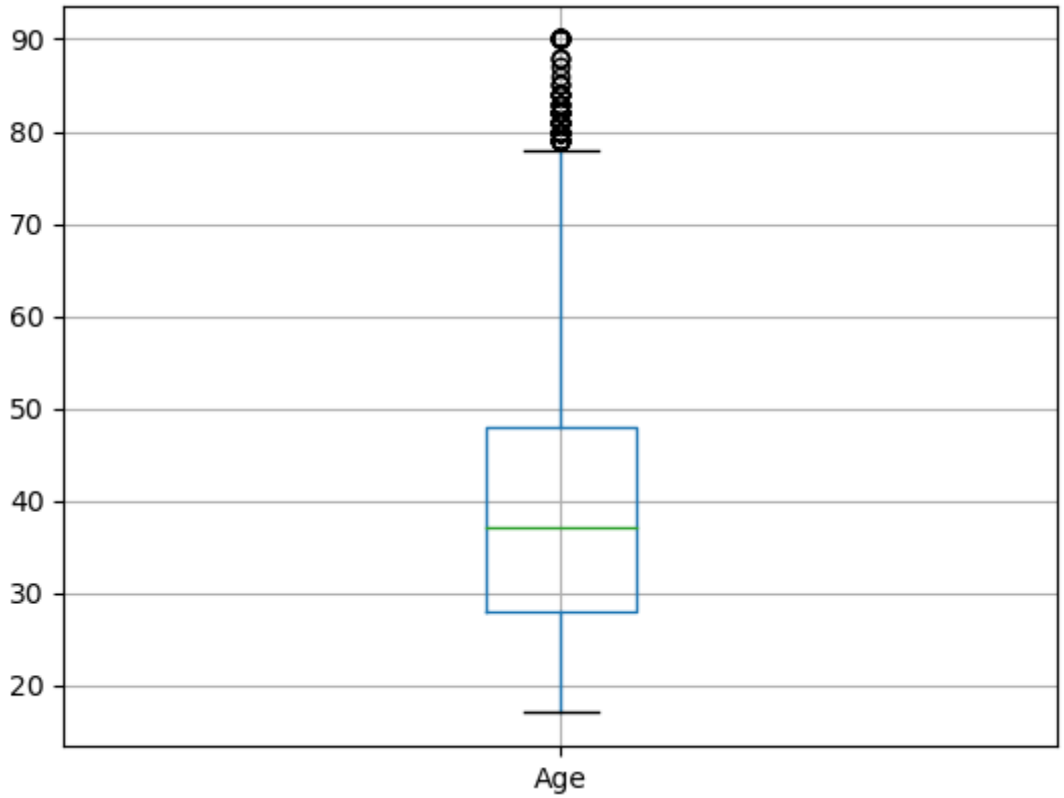
In [37]:
```
data['Age'].hist(bins=100)
```

Out[37]: <AxesSubplot:>



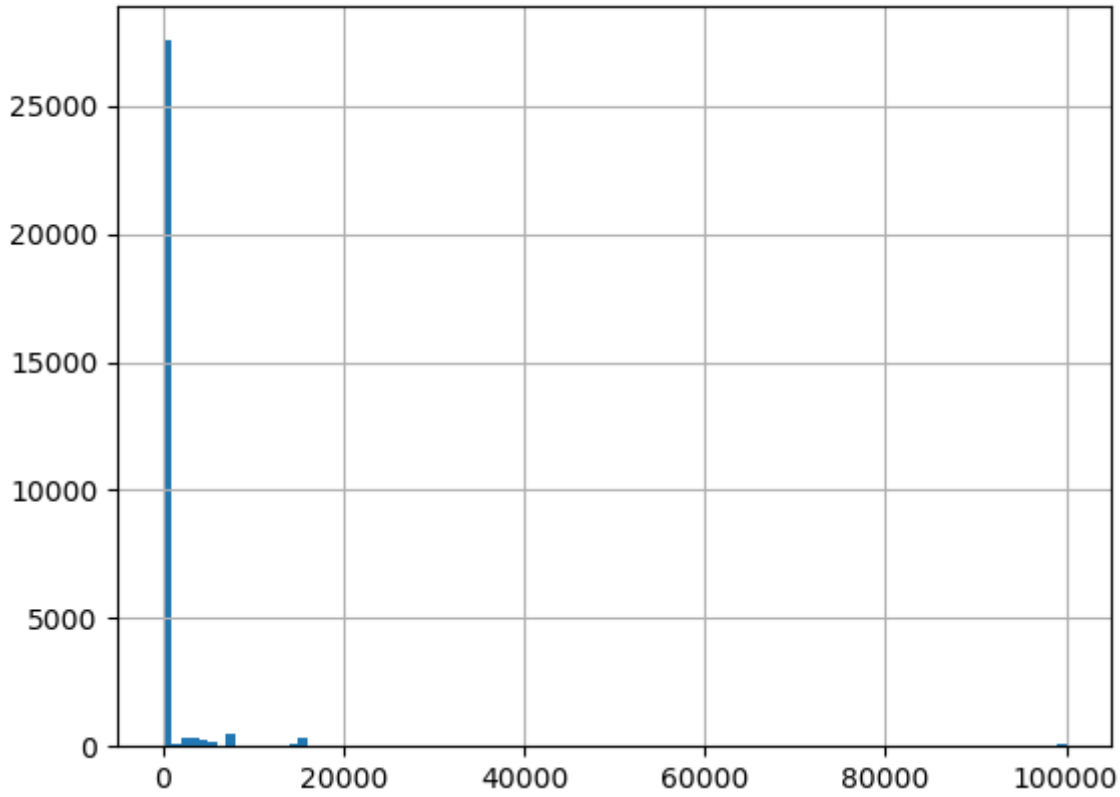
In [38]:
```
data.boxplot(column='Age')
```

Out[38]: <AxesSubplot:>



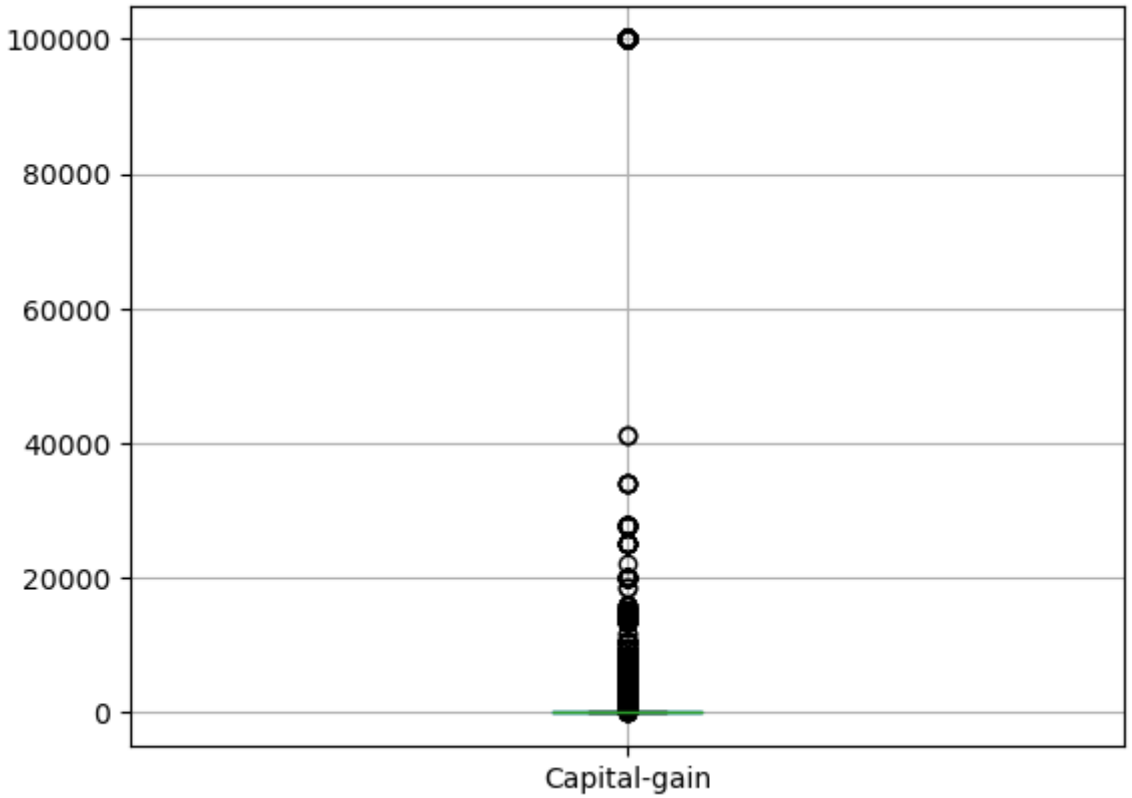
In [39]:
```
data['Capital-gain'].hist(bins=100)
```

Out[39]: <AxesSubplot:>



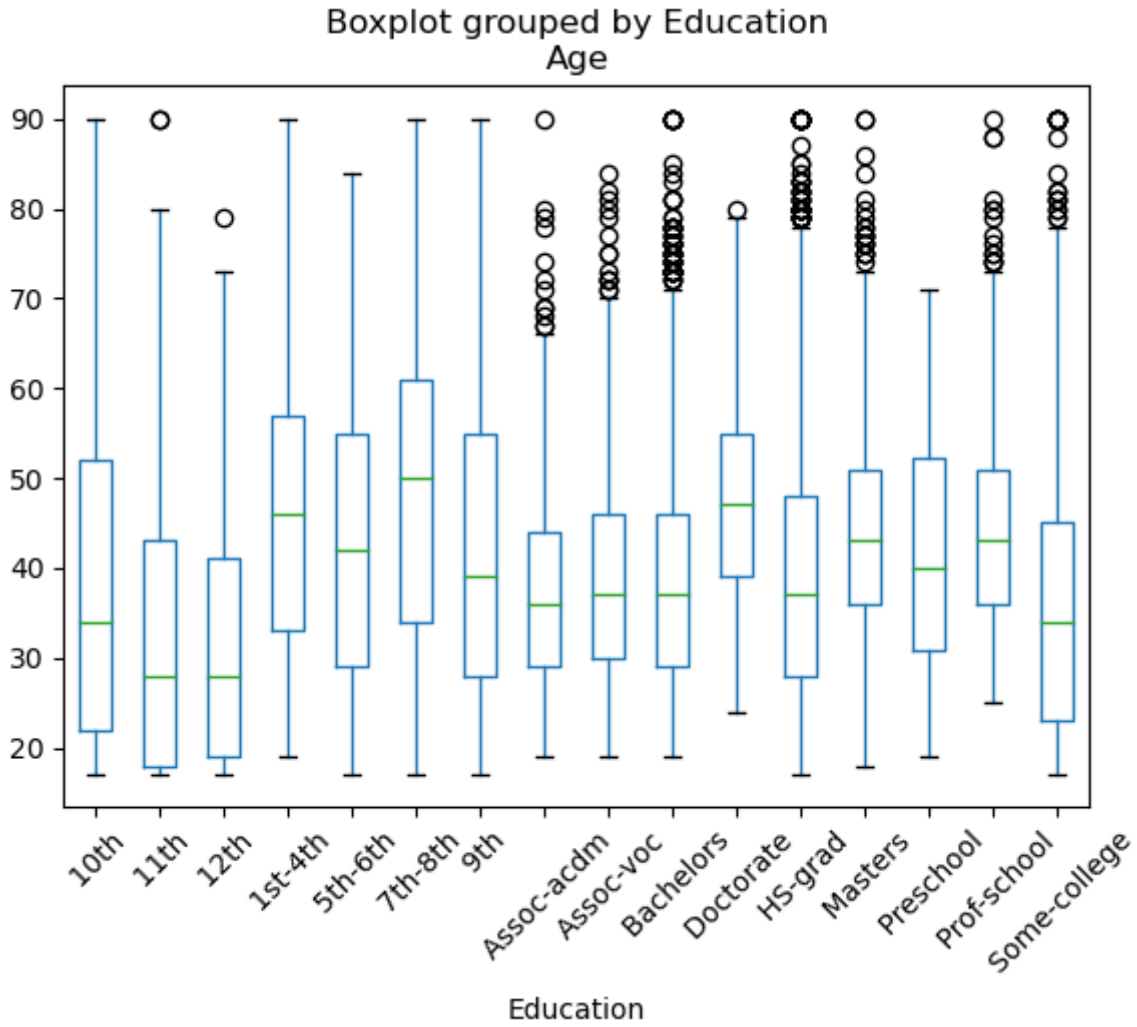
In [40]:
```
data.boxplot(column='Capital-gain')
```

Out[40]: <AxesSubplot:>



In [41]:
```
data.boxplot(column='Age', by = 'Education', grid=False, rot = 45, fontsize = 10)
```

Out[41]: <AxesSubplot:title={'center':'Age'}, xlabel='Education'>

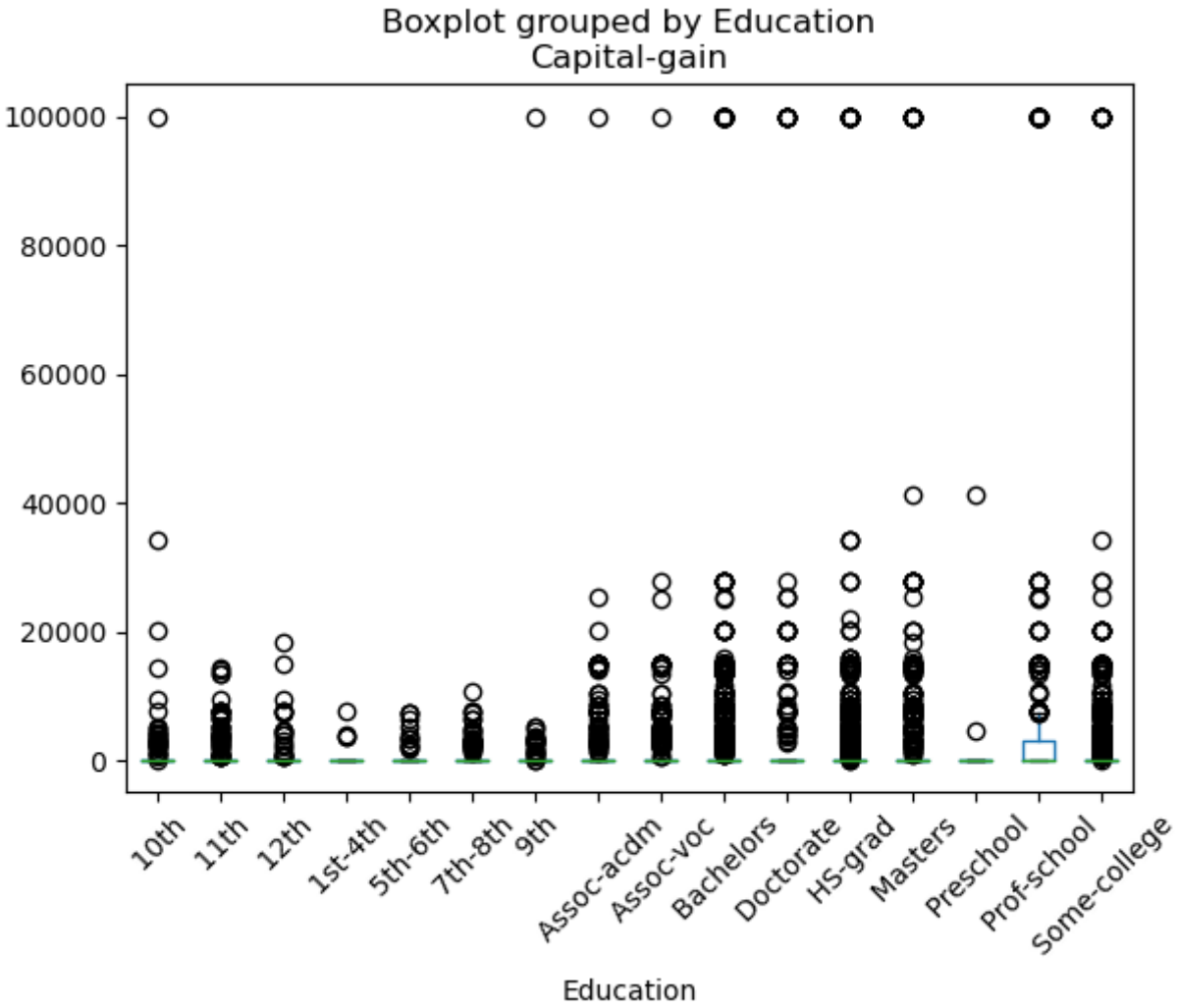


In [42]:
```
data['Education'].value_counts()
```

Out[42]:
```
 HS-grad         9716
 Some-college    6717
 Bachelors       4924
 Masters         1582
 Assoc-voc       1269
 11th            1076
 Assoc-acdm       984
 10th             856
 7th-8th          585
 Prof-school      539
 9th              474
 12th             388
 Doctorate        380
 5th-6th          305
 1st-4th          157
 Preschool         48
Name: Education, dtype: int64
```

In [43]:
```
data.boxplot(column='Capital-gain', by = 'Education', grid=False, rot = 45, fontsize = 10)
```

Out[43]:
```
<AxesSubplot:title={'center':'Capital-gain'}, xlabel='Education'>
```



In [44]:
```
data['Marital-Status'].value_counts()
```

Out[44]:
```
 Married-civ-spouse       13791
 Never-married             9885
 Divorced                  4082
 Separated                  940
 Widowed                    900
 Married-spouse-absent      380
 Married-AF-spouse           22
Name: Marital-Status, dtype: int64
```

Checking NULL values in the dataset

In [45]:
```
data.apply(lambda x: sum(x.isnull()), axis = 0)
```

Out[45]:
```
Age              0
Worklass         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 [46]:
```
from sklearn.preprocessing import LabelEncoder
```

In [47]:
```
data.head()
```

Out[47]:

	Age	Worklass	Education	Education-num	Marital-Status	Occupation	Relationship	Race	Sex	Capital-gain	Capital-loss	Hours-per-week	Native-country	Class-label
3242	29	Private	HS-grad	9	Never-married	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
3139	22	Private	Some-college	10	Never-married	Adm-clerical	Own-child	White	Male	0	0	30	United-States	<=50K
20837	58	Local-gov	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	40	United-States	<=50K
24510	50	Self-emp-not-inc	Prof-school	15	Married-civ-spouse	Exec-managerial	Husband	White	Male	99999	0	50	United-States	>50K
19135	19	Private	11th	7	Never-married	Farming-fishing	Own-child	White	Male	0	0	24	United-States	<=50K

In [48]:
```
data.dtypes
```

Out[48]:
```
Age                int64
Worklass          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 [49]:
```
columns = list(data.select_dtypes(exclude=['int64']))
```

In [50]:
```
columns
```

Out[50]:
```
['Worklass',
 'Education',
 'Marital-Status',
 'Occupation',
 'Relationship',
 'Race',
 'Sex',
 'Native-country',
 'Class-label']
```

In [51]:
```
data['Class-label'].value_counts()
```

Out[51]:
```
 <=50K    22747
 >50K      7253
Name: Class-label, dtype: int64
```

In [52]:
```
le = LabelEncoder()
for i in columns:
 #print(i)
 data[i] = le.fit_transform(data[i])
data.dtypes
```

Out[52]:
```
Age              int64
Worklass         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 [53]:
```
data.head()
```

Out[53]:

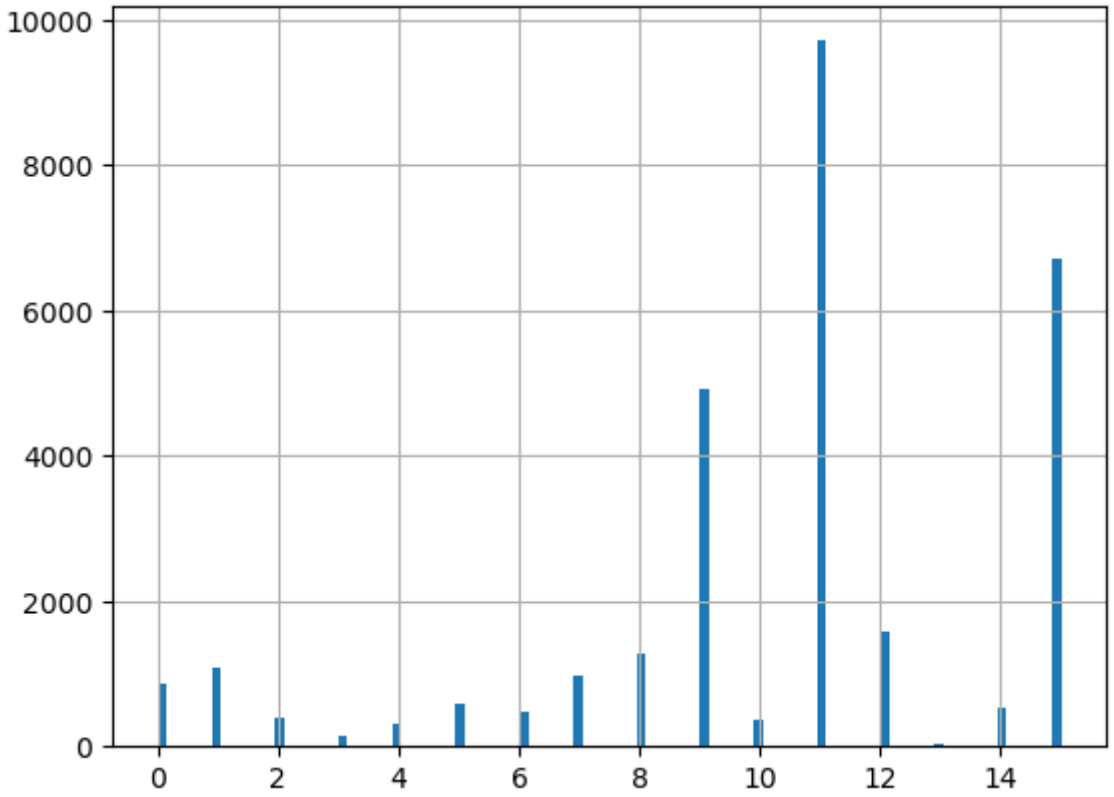
	Age	Worklass	Education	Education-num	Marital-Status	Occupation	Relationship	Race	Sex	Capital-gain	Capital-loss	Hours-per-week	Native-country	Class-label
3242	29	4	11	9	4	6	1	4	1	0	0	40	39	0
3139	22	4	15	10	4	1	3	4	1	0	0	30	39	0
20837	58	2	9	13	2	4	0	4	1	0	0	40	39	0
24510	50	6	14	15	2	4	0	4	1	99999	0	50	39	1
19135	19	4	1	7	4	5	3	4	1	0	0	24	39	0

In [54]:
```
data['Worklass'].value_counts()
```

Out[54]:
```
4    20890
6     2351
2     1916
0     1699
7     1209
5     1034
1      885
8       10
3        6
Name: Worklass, dtype: int64
```

In [55]:
```
data['Education'].hist(bins=100)
```

Out[55]: <AxesSubplot:>



In [56]:
```
data.describe(include='all')
```

Out[56]:

	Age	Worklass	Education	Education-num	Marital-Status	Occupation	Relationship	Race	Sex	Capital-g
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000
mean	38.542267	3.870467	10.308433	10.082167	2.612800	6.570100	1.442700	3.666500	0.669600	1061.891
std	13.643028	1.458272	3.862262	2.570545	1.504067	4.232038	1.604307	0.847605	0.470365	7224.973
min	17.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
25%	28.000000	4.000000	9.000000	9.000000	2.000000	3.000000	0.000000	4.000000	0.000000	0.000
50%	37.000000	4.000000	11.000000	10.000000	2.000000	7.000000	1.000000	4.000000	1.000000	0.000
75%	48.000000	4.000000	12.000000	12.000000	4.000000	10.000000	3.000000	4.000000	1.000000	0.000
max	90.000000	8.000000	15.000000	16.000000	6.000000	14.000000	5.000000	4.000000	1.000000	99999.000

REPORT

Q6 answer

The code data.describe() on the dataset with the use of Pandas is used to provide a statistical summary of the dataframe that has been created. There with the tools used in phyton programming stataistical results such as mean, minimum and maximum value, first quartile and third quartile, median and outliers are found out for each of the numerical variables or attributes such as age, workclass, occupation and other variables found on the first row of the dataset.

Q7 answer

The different data types/attributes in data mining are classification, clustering, association rule learning, regression, anomaly detection, sequentila pattern mining, artificial network classifier, outlier analysis, prediction, and genetic algorithm (Simplilearn.com, 2022)

Q8 answer

Type *Markdown* and LaTeX: α^2

Q9 answer

In [59]:
```
data['Sex'].groupby([data['Occupation']]).value_counts()
```

Out[59]:
```
Occupation  Sex
0           1       931
            0       774
1           0      2349
            1      1131
2           1         8
3           1      3570
            0       210
4           1      2690
            0      1054
5           1       864
            0        60
6           1      1104
            0       148
7           1      1335
            0       504
8           0      1660
            1      1365
9           0       130
            1         6
10          1      2434
            0      1389
11          1       522
            0        69
12          1      2207
            0      1151
13          1       525
            0       330
14          1      1396
            0        84
Name: Sex, dtype: int64
```

The occupation that reprsents more males than females is Handlers-Cleaners.

Q10 answer

The difference between data.head() and data.tail() is that the code data.head() is what is used to display the first 5 rows of the dataframe data where as the code data.tail() is used to display the last 5 rows or whatever last n rows of the dataframe data in order to perform proper analysis.

References

Simplilearn.com. (2022). Types of Data Mining Techniques | Simplilearn. [online] Available at: https://www.simplilearn.com/types-of-data-mining-techniques-article (https://www.simplilearn.com/types-of-data-mining-techniques-article).