In [1]:
```
import pandas as pd
```

In [2]:
```
import numpy as np
```

In [3]:
```
import matplotlib.pyplot as plt
```

In [4]:
```
%matplotlib inline
```

In [5]:
```
data = pd.read_csv('adult.csv')
```

In [6]:
```
data.head()
```

Out[6]:

	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

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

In [7]:
```
data.head(2)
```

Out[7]:

	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 [8]:
```
data.head(10)
```

Out[8]:

	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

In [9]:
```
data.tail(2)
```

Out[9]:

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

Using the .head and .tail dataframe functions from the pandas package allows us to display our data in a tabular format; Passing a parameter value of 2 to the head function will display the first two rows in the dataset, likewise passing a parameter value of 10 will display the first 10 rows. The same applies for the tail function, except that the tail function will display the last rows; for example passsing a parameter value of 2 will display the last two rows of the dataframe

In [10]:
```
data.shape
```

Out[10]: (32561, 15)

In [11]:
```
data = data.sample(n=30000, random_state = 450)
```

In [12]:
```
data.shape
```

Out[12]: (30000, 15)

In [13]:
```
data.describe()
```

Out[13]:

	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	38.596800	1.894571e+05	10.082467	1063.867800	87.217367	40.423300
std	13.635848	1.050737e+05	2.570905	7299.627718	402.785077	12.351478
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.177500e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.779985e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.368742e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.455435e+06	16.000000	99999.000000	4356.000000	99.000000

In [14]:
```
data['education-num'].value_counts()
```

Out[14]:
```
9     9686
10    6723
13    4951
14    1583
11    1261
7     1093
12     982
6      843
4      596
15     528
5      470
8      398
16     380
3      301
2      155
1       50
Name: education-num, dtype: int64
```

In [15]:
```
data['education'].value_counts()
```

Out[15]:
```
HS-grad         9686
Some-college    6723
Bachelors       4951
Masters         1583
Assoc-voc       1261
11th            1093
Assoc-acdm       982
10th             843
7th-8th          596
Prof-school      528
9th              470
12th             398
Doctorate        380
5th-6th          301
1st-4th          155
Preschool         50
Name: education, dtype: int64
```

In [16]:
```
data.describe()
```

Out[16]:

	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	38.596800	1.894571e+05	10.082467	1063.867800	87.217367	40.423300
std	13.635848	1.050737e+05	2.570905	7299.627718	402.785077	12.351478
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.177500e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.779985e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.368742e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.455435e+06	16.000000	99999.000000	4356.000000	99.000000

In [17]:
```
data = data.drop(['fnlwgt'], axis=1)
```

In [18]:
```
data.describe(include='all')
```

Out[18]:

	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	20921	9686	NaN	13820	3815	12174	25619
mean	38.596800	NaN	NaN	10.082467	NaN	NaN	NaN	NaN
std	13.635848	NaN	NaN	2.570905	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

In [19]:
```
data['education'].value_counts()
```

Out[19]:
```
HS-grad         9686
Some-college    6723
Bachelors       4951
Masters         1583
Assoc-voc       1261
11th            1093
Assoc-acdm       982
10th             843
7th-8th          596
Prof-school      528
9th              470
12th             398
Doctorate        380
5th-6th          301
1st-4th          155
Preschool         50
Name: education, dtype: int64
```

In [20]:
```
data['education'].nunique()
```

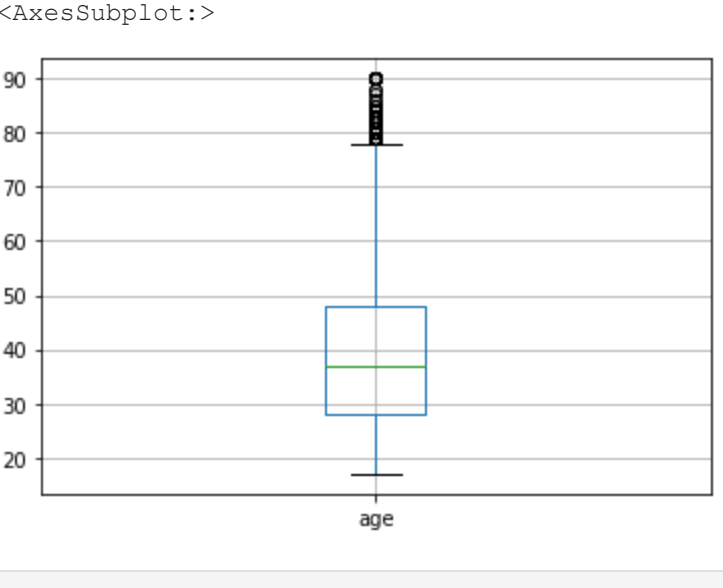
Out[20]: 16

In [21]:
```
data['age'].value_counts()
```

Out[21]:
```
31    833
34    830
36    826
35    812
33    807
     ...
83      6
85      3
88      2
86      1
87      1
Name: age, Length: 73, dtype: int64
```

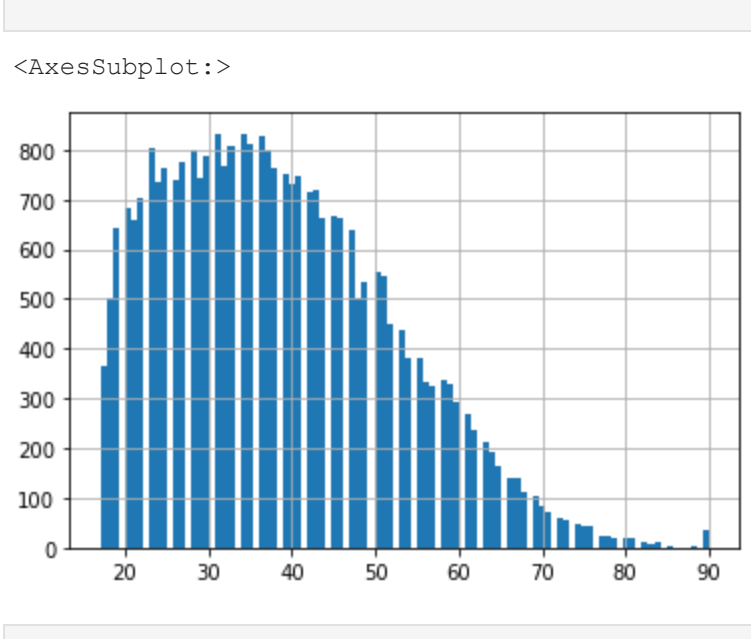
In [22]:
```
data.boxplot(['age'])
```

Out[22]: <AxesSubplot:>



In [23]:
```
data['age'].hist(bins=100)
```

Out[23]: <AxesSubplot:>



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

Out[24]:
```
Male      20066
Female     9934
Name: sex, dtype: int64
```

In [29]:
```
data.columns
```

Out[29]:
```
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 [30]:
```
data['workclass'].value_counts()
```

Out[30]:
```
Private             20921
Self-emp-not-inc     2334
Local-gov            1942
?                    1689
State-gov            1188
Self-emp-inc         1023
Federal-gov           885
Without-pay            12
Never-worked            6
Name: workclass, dtype: int64
```

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

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

Out[31]:
```
Male      20066
Female     9934
Name: sex, dtype: int64
```

There are 20066 males in the dataset. There are 9934 females in the dataset.

What is the average age of each gender in the given population?

In [32]:
```
data['age'].groupby([data['sex']]).mean()
```

Out[32]:
```
sex
Female    36.890175
Male      39.441692
Name: age, dtype: float64
```

What is the average age of males and females across different education categories?

In [33]:
```
data['age'].groupby([data['sex'],data['education']]).mean()
```

Out[33]:
```
sex     education
Female  10th            35.229927
        11th            30.439206
        12th            29.637037
        1st-4th         49.116279
        5th-6th         44.090909
        7th-8th         50.021277
        9th             42.330769
        Assoc-acdm      36.403694
        Assoc-voc       38.229399
        Bachelors       35.602402
        Doctorate       45.086420
        HS-grad         38.694144
        Masters         42.952772
        Preschool       40.933333
        Prof-school     40.209877
        Some-college    33.924172
Male    10th            38.543058
        11th            33.905797
        12th            32.562738
        1st-4th         44.928571
        5th-6th         42.683036
        7th-8th         48.197802
        9th             40.282353
        Assoc-acdm      37.935323
        Assoc-voc       39.110837
        Bachelors       40.344148
        Doctorate       48.170569
        HS-grad         39.095355
        Masters         44.447080
        Preschool       43.228571
        Prof-school     45.523490
        Some-college    36.995879
Name: age, dtype: float64
```

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

In [34]:
```
data['capital-gain'].groupby([data['sex'], data['occupation']]).mean()
```

Out[34]:
```
sex     occupation
Female  ?                     302.150454
        Adm-clerical          532.062794
        Craft-repair          819.614634
        Exec-managerial       924.759482
        Farming-fishing       766.050847
        Handlers-cleaners     119.298701
        Machine-op-inspct     187.298387
        Other-service         165.345531
        Priv-house-serv       116.676471
        Prof-specialty       1165.553546
        Protective-serv      1704.402778
        Sales                 263.819121
        Tech-support          658.317901
        Transport-moving      455.517647
Male    ?                     726.081169
        Adm-clerical          488.594356
        Armed-Forces            0.000000
        Craft-repair          660.604004
        Exec-managerial      2724.392684
        Farming-fishing       519.576832
        Handlers-cleaners     270.381638
        Machine-op-inspct     403.953066
        Other-service         247.755523
        Priv-house-serv        84.857143
        Prof-specialty       3433.873818
        Protective-serv       547.522727
        Sales                1941.171455
        Tech-support          723.411765
        Transport-moving      524.976139
Name: capital-gain, dtype: float64
```

Q4. What is the average capital-gain by males and females across different marital status?

In [35]:
```
data['capital-gain'].groupby([data['sex'], data['marital-status']]).mean()
```

Out[35]:
```
sex     marital-status
Female  Divorced                  445.054516
        Married-AF-spouse         189.500000
        Married-civ-spouse       1421.084912
        Married-spouse-absent     381.242268
        Never-married             337.220856
        Separated                 359.801695
        Widowed                   458.801325
Male    Divorced                 1169.319056
        Married-AF-spouse         810.888889
        Married-civ-spouse       1772.102124
        Married-spouse-absent    1007.090909
        Never-married             404.258475
        Separated                 900.275204
        Widowed                   838.347561
Name: capital-gain, dtype: float64
```

In [74]:
```
data['race'].value_counts()
```

Out[74]:
```
4    25619
2     2892
1      957
0      286
3      246
Name: race, dtype: int64
```

In [77]:
```
'''
    Q5. Minimum and Maximum age by sex are the same?
'''
data['age'].groupby([data['sex']]).min()
```

Out[77]:
```
sex
0    17
1    17
Name: age, dtype: int64
```

In [78]:
```
data['age'].groupby([data['sex']]).max()
```

Out[78]:
```
sex
0    90
1    90
Name: age, dtype: int64
```

In [40]:
```
import matplotlib.pyplot as plt
```

In [41]:
```
%matplotlib inline
```

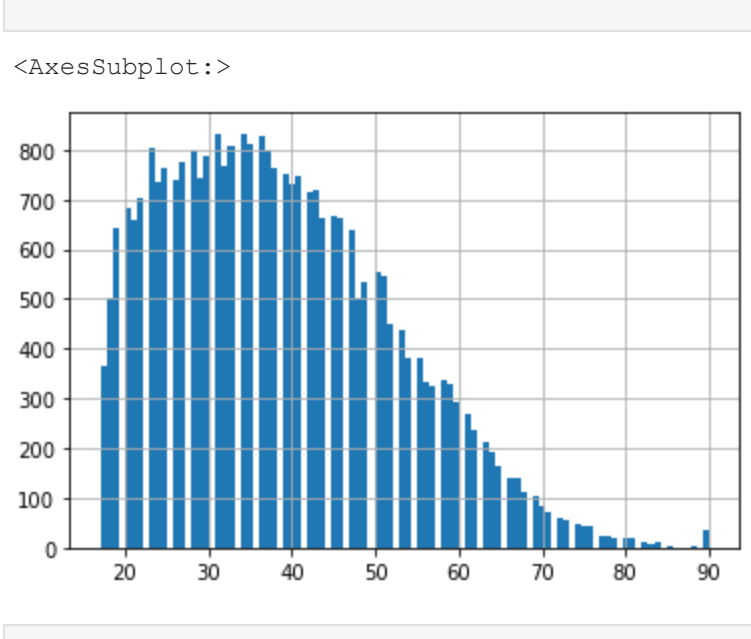
In [42]:
```
data.describe()
```

Out[42]:

	age	education-num	capital-gain	capital-loss	hours-per-week
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000
mean	38.596800	10.082467	1063.867800	87.217367	40.423300
std	13.635848	2.570905	7299.627718	402.785077	12.351478
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

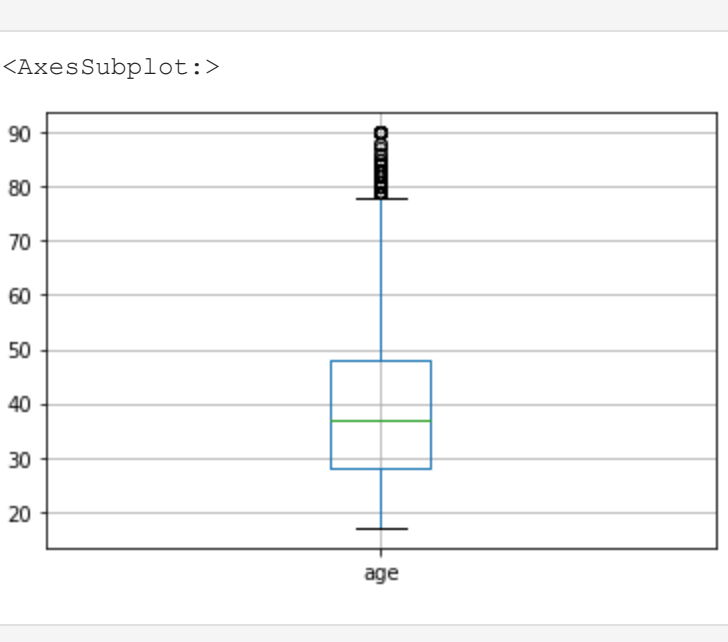
In [43]:
```
data['age'].hist(bins=100)
```

Out[43]: <AxesSubplot:>



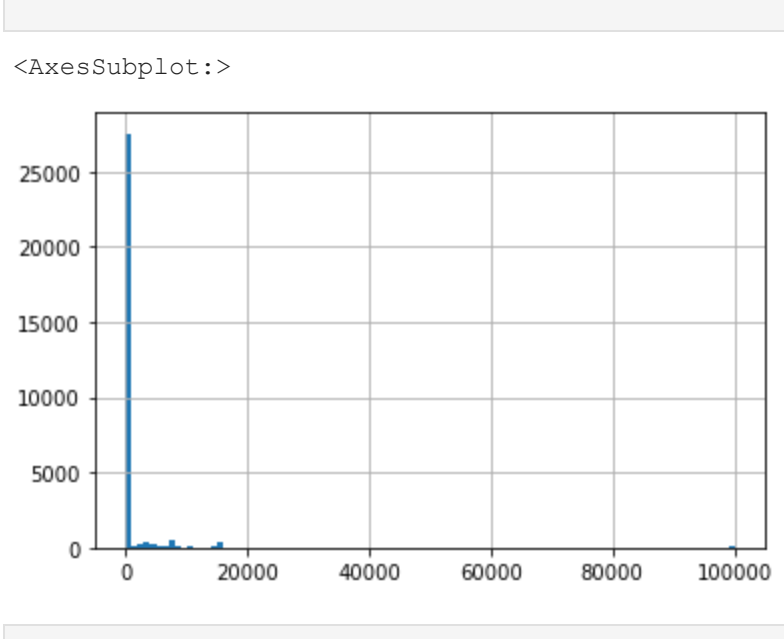
In [44]:
```
data.boxplot(column='age')
```

Out[44]: <AxesSubplot:>



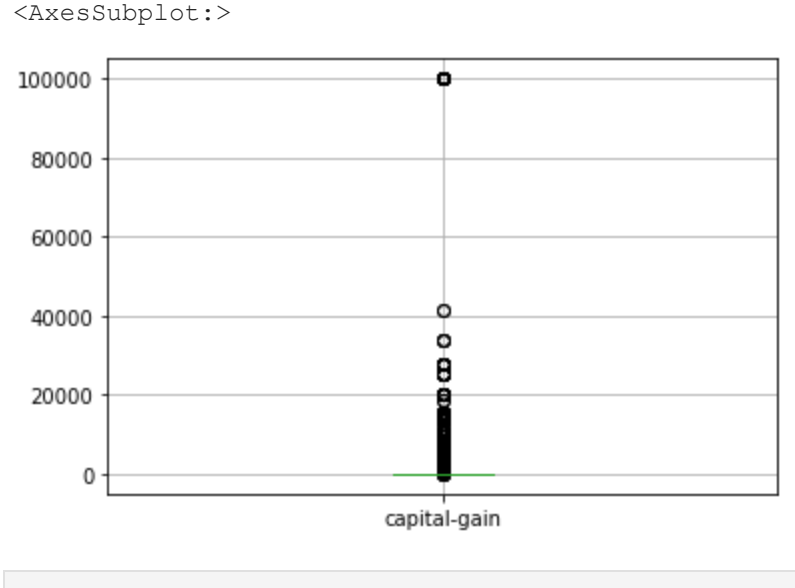
In [45]:
```
data['capital-gain'].hist(bins=100)
```

Out[45]: <AxesSubplot:>



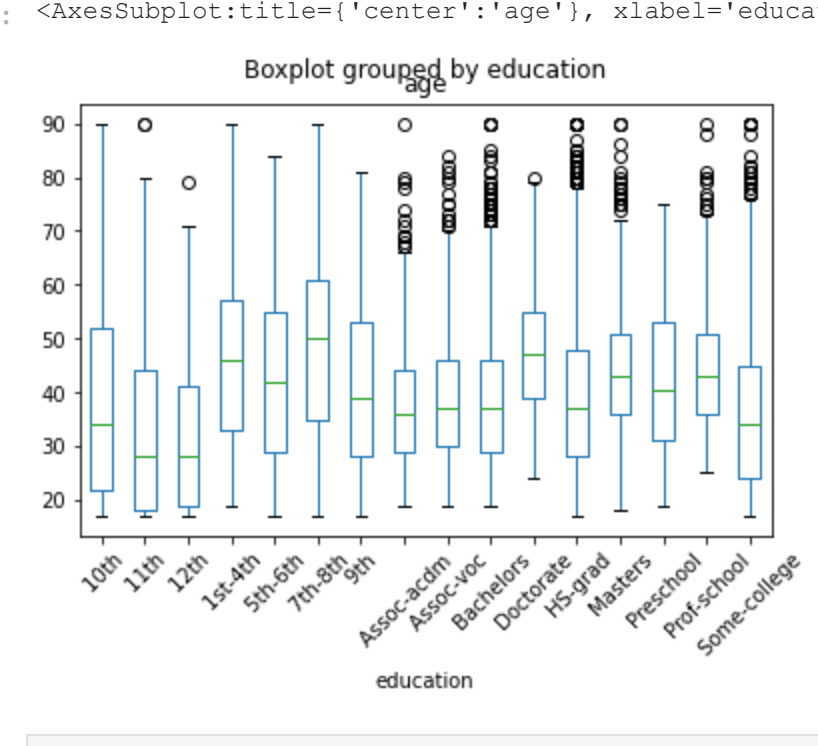
In [46]:
```
data.boxplot(column='capital-gain')
```

Out[46]: <AxesSubplot:>



In [47]:
```
data.boxplot(column='age', by = 'education', grid=False, rot=45, fontsize=10)
```

Out[47]: <AxesSubplot:title={'center':'age'}, xlabel='education'>

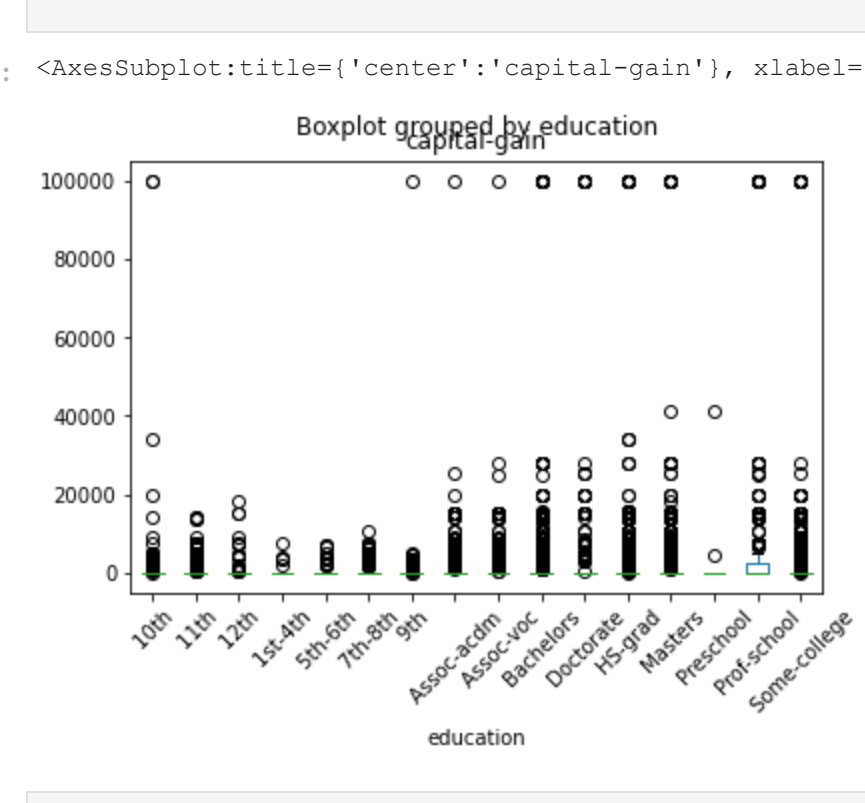


In [48]:
```
data['education'].value_counts()
```

Out[48]:
```
HS-grad         9686
Some-college    6723
Bachelors       4951
Masters         1583
Assoc-voc       1261
11th            1093
Assoc-acdm       982
10th             843
7th-8th          596
Prof-school      528
9th              470
12th             398
Doctorate        380
5th-6th          301
1st-4th          155
Preschool         50
Name: education, dtype: int64
```

In [50]:
```
data.boxplot(column='capital-gain', by = 'education', grid=False, rot = 45 , fontsize
```

Out[50]: <AxesSubplot:title={'center':'capital-gain'}, xlabel='education'>



In [51]:
```
data['marital-status'].value_counts()
```

Out[51]:
```
Married-civ-spouse       13820
Never-married             9820
Divorced                  4069
Separated                  957
Widowed                    919
Married-spouse-absent      392
Married-AF-spouse           23
Name: marital-status, dtype: int64
```

In [55]:
```
data.apply(lambda x:sum(x .isnull()), axis = 0)
```

Out[55]:
```
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 [56]:
```
from sklearn.preprocessing import LabelEncoder
```

In [57]:
```
data.head()
```

Out[57]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capita ga
28236	18	Private	HS-grad	9	Never-married	Sales	Not-in-family	White	Female	
2639	28	Private	5th-6th	3	Never-married	Machine-op-inspct	Not-in-family	White	Female	
16205	36	Private	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	
2344	44	Private	Masters	14	Married-civ-spouse	Exec-managerial	Husband	White	Male	1502
11620	67	Self-emp-not-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	

In [58]:
```
data.dtypes
```

Out[58]:
```
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 [59]:
```
columns = list(data.select_dtypes(exclude=['int64']))
```

In [60]:
```
columns
```

Out[60]:
```
['workclass',
 'education',
 'marital-status',
 'occupation',
 'relationship',
 'race',
 'sex',
 'native-country',
 'class-label']
```

In [61]:
```
data['class-label'].value_counts()
```

Out[61]:
```
 <=50K    22756
 >50K      7244
Name: class-label, dtype: int64
```

In [67]:
```
le = LabelEncoder()
```

In [68]:
```
for i in columns:
    data[i] = le.fit_transform(data[i])
```

In [69]:
```
data.dtypes
```

Out[69]:
```
age               int64
workclass         int64
education         int64
education-num     int64
marital-status    int64
occupation        int64
relationship      int64
race              int64
sex               int64
capital-gain      int64
capital-loss      int64
hours-per-week    int64
native-country    int64
class-label       int64
dtype: object
```

In [70]:
```
data.head()
```

Out[70]:

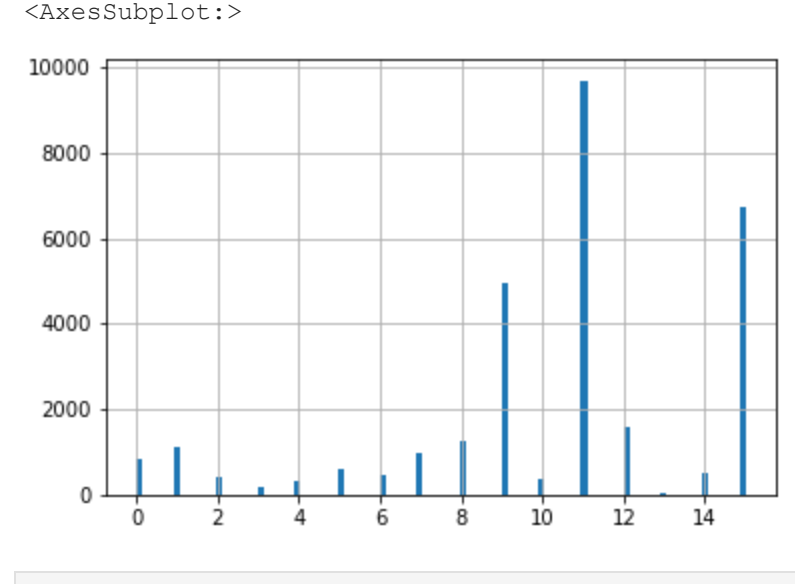
	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain
28236	18	4	11	9	4	12	1	4	0	0
2639	28	4	4	3	4	7	1	4	0	0
16205	36	4	15	10	2	7	0	4	1	0
2344	44	4	12	14	2	4	0	4	1	15024
11620	67	6	9	13	2	4	0	4	1	0

In [71]:
```
data['workclass'].value_counts()
```

Out[71]:
```
4    20921
6     2334
2     1942
0     1689
7     1188
5     1023
1      885
8       12
3        6
Name: workclass, dtype: int64
```

In [72]:
```
data['education'].hist(bins=100)
```

Out[72]: <AxesSubplot:>



In [73]:
```
data.describe(include='all')
```

Out[73]:

	age	workclass	education	education-num	marital-status	occupation	rela
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000
mean	38.596800	3.866733	10.303600	10.082467	2.613933	6.569700	1
std	13.635848	1.454594	3.865358	2.570905	1.505355	4.230627	'
min	17.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0
25%	28.000000	4.000000	9.000000	9.000000	2.000000	3.000000	0
50%	37.000000	4.000000	11.000000	10.000000	2.000000	7.000000	1
75%	48.000000	4.000000	12.000000	12.000000	4.000000	10.000000	3
max	90.000000	8.000000	15.000000	16.000000	6.000000	14.000000	5

Report

Q6. Write a summary of the outcome of data.describe()

In [79]:
```
data.describe()
```

Out[79]:

	age	workclass	education	education-num	marital-status	occupation	rela
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000
mean	38.596800	3.866733	10.303600	10.082467	2.613933	6.569700	1
std	13.635848	1.454594	3.865358	2.570905	1.505355	4.230627	'
min	17.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0

25%	28.000000	4.000000	9.000000	9.000000	2.000000	3.000000	0
50%	37.000000	4.000000	11.000000	10.000000	2.000000	7.000000	1
75%	48.000000	4.000000	12.000000	12.000000	4.000000	10.000000	3
max	90.000000	8.000000	15.000000	16.000000	6.000000	14.000000	5

data.describe() is a method provided by the pandas library which allows us to review some standard calculations performed on our data. For example, I can see that the mean (average) age in my specific data set is 38.5. I can see that the 25th percentile of workclass within my data has a value of 4, before the values were encoded/transformed, this was 'Private'. The outcome/output of data.describe() is a table with values populated by standard analytical calculations performed on the data in each column.

Q7. What are the different datatypes in data mining? Explain with the help of the examplesfrom Adult dataset.

In [80]:
```
data.dtypes
```

Out[80]:
```
age               int64
workclass         int64
education         int64
education-num     int64
marital-status    int64
occupation        int64
relationship      int64
race              int64
sex               int64
capital-gain      int64
capital-loss      int64
hours-per-week    int64
native-country    int64
class-label       int64
dtype: object
```

The four data types I've seen so far for data mining are int64 (a 64bit integer), float64 (a 64 bit floating point number), Object and datetime64[ns]. An Integer is a numeric value, for example in our Adult dataset, age is a number and so would naturally be type int64. Sex however is a string value, though when you run data.dtypes you would see Object. At present we do not have any float64 values in our dataframe, however it is important to note that they are a data type found in data mining. There are more according to the documentation for the pandas packages, which can be found here: https://pandas.pydata.org/pandas-docs/stable/user_guide/basics.html#dtypes

Q8. Highest migrants belongs to which country?

In [81]:
```
data['native-country'].value_counts()
```

Out[81]:
```
39    26884
26      588
0       544
30      188
11      126
2       108
33      101
8        97
19       91
9        85
5        84
23       74
35       71
22       68
6        66
3        64
40       61
13       58
31       56
24       54
4        54
36       50
14       43
20       43
32       34
12       27
10       26
27       26
29       26
7        26
21       24
17       19
38       17
25       17
1        17
37       17
41       16
18       13
16       13
28       12
34       11
15        1
Name: native-country, dtype: int64
```

Most migrants belong to country 26 (Mexico). This is because we are working with a US dataset and people from country 39 (United States) are not migrants. You could use the .value_counts function on the data['native-country'] column to summarize value counts for each country. You could also use a histogram chart to display this data.

Q9. Which occupation represents more males than females?

In [87]:
```
data['occupation'].groupby([data['sex']]).value_counts()
```

Out[87]:
```
sex  occupation
0    1             2341
     8             1667
     10            1382
     12            1161
     4             1081
     0              771
     7              496
     13             324
     3              205
     6              154
     9              136
     14              85
     11              72
     5               59
1    3             3596
     4             2679
     10            2433
     12            2193
     14            1383
     8             1358
     7             1321
     1             1134
     6             1111
     0              924
     5              846
     13             544
     11             528
     2                9
     9                7
Name: occupation, dtype: int64
```

Craft-repair, occuopation with a value of 3 has the highest representation of males with a value count of 3596. The female value count for the same occupation is 205. Therefore Craft-repair has the highest male representation. value_counts provides us with a frequency table, this allows us to quickly analyse summary data. groupby allows us to group our data by a selected category, in the example above we have chosen sex, this is ideal because the options available in this dataset are binary.

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

In [88]:
```
data.head()
```

Out[88]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	c
28236	18	4	11	9	4	12	1	4	0	0	
2639	28	4	4	3	4	7	1	4	0	0	
16205	36	4	15	10	2	7	0	4	1	0	
2344	44	4	12	14	2	4	0	4	1	15024	
11620	67	6	9	13	2	4	0	4	1	0	

In [89]:
```
data.tail()
```

Out[89]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	c
4826	20	4	15	10	4	1	4	1	0	0	
7751	69	6	6	5	2	1	0	4	1	0	
2254	63	5	11	9	2	4	0	4	1	0	
13679	34	4	15	10	2	3	0	4	1	0	
16057	32	4	15	10	4	6	3	4	1	0	

data.head and data.tail are similar functions in that they display data from the dataset in a tabular format. They are different however, in that they retrieve data from different ends of the dataset. data.head() retrieves the first five rows in the dataframe, whereas data.tail() retrieves the last five rows in the dataframe.