Exercise - Getting and Knowing your Data-Occupation Dataset

This time we are going to pull data directly from the internet.

Step 1. Import the necessary libraries

In [62]:

```
import numpy as np
import pandas as pd
```

Step 2. Import the dataset from this address.

In [63]:

```
data=pd.read_csv("https://raw.githubusercontent.com/justmarkham/DAT8/master/
data
```

Out[63]:

	user_id|age|gender|occupation|zip_code
0	1|24|M|technician|85711
1	2|53|F|other|94043
2	3|23|M|writer|32067
3	4|24|M|technician|43537
4	5|33|F|other|15213
...	...
938	939|26|F|student|33319
939	940|32|M|administrator|02215
940	941|20|M|student|97229
941	942|48|F|librarian|78209
942	943|22|M|student|77841

943 rows × 1 columns

In [64]:

```
data=pd.read_csv("https://raw.githubusercontent.com/justmarkham/DAT8/master/
data
```

Loading [MathJax]/extensions/Safe.js

Out[64]:

	user_id	age	gender	occupation	zip_code
0	1	24	M	technician	85711
1	2	53	F	other	94043
2	3	23	M	writer	32067
3	4	24	M	technician	43537
4	5	33	F	other	15213
...	...	...	...	...	...
938	939	26	F	student	33319
939	940	32	M	administrator	02215
940	941	20	M	student	97229
941	942	48	F	librarian	78209
942	943	22	M	student	77841

943 rows × 5 columns

Step 3. Assign it to a variable called users and use the 'user_id' as index

In [65]:

```
data=data.set_index("user_id")
data
```

Out[65]:

	age	gender	occupation	zip_code
user_id				
1	24	M	technician	85711
2	53	F	other	94043
3	23	M	writer	32067
4	24	M	technician	43537
5	33	F	other	15213
...	...	...	...	...
939	26	F	student	33319
940	32	M	administrator	02215
941	20	M	student	97229
942	48	F	librarian	78209
943	22	M	student	77841

943 rows × 4 columns

Loading [MathJax]/extensions/Safe.js

Step 4. See the first 25 entries

In [66]:

```
data.head(25)
```

Out[66]:

	age	gender	occupation	zip_code
user_id				
1	24	M	technician	85711
2	53	F	other	94043
3	23	M	writer	32067
4	24	M	technician	43537
5	33	F	other	15213
6	42	M	executive	98101
7	57	M	administrator	91344
8	36	M	administrator	05201
9	29	M	student	01002
10	53	M	lawyer	90703
11	39	F	other	30329
12	28	F	other	06405
13	47	M	educator	29206
14	45	M	scientist	55106
15	49	F	educator	97301
16	21	M	entertainment	10309
17	30	M	programmer	06355
18	35	F	other	37212
19	40	M	librarian	02138
20	42	F	homemaker	95660
21	26	M	writer	30068
22	25	M	writer	40206
23	30	F	artist	48197
24	21	F	artist	94533
25	39	M	engineer	55107

Step 5. See the last 10 entries

In [67]:

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

Out[67]:

user_id	age	gender	occupation	zip_code
934	61	M	engineer	22902
935	42	M	doctor	66221
936	24	M	other	32789
937	48	M	educator	98072
938	38	F	technician	55038
939	26	F	student	33319
940	32	M	administrator	02215
941	20	M	student	97229
942	48	F	librarian	78209
943	22	M	student	77841

Step 6. What is the number of observations in the dataset?

```
In [68]: data.shape
```

```
Out[68]: (943, 4)
```

Step 7. What is the number of columns in the dataset?

```
In [69]: len("columns")
```

```
Out[69]: 7
```

Step 8. Print the name of all the columns.

```
In [70]: c=data.columns
         c
```

```
Out[70]: Index(['age', 'gender', 'occupation', 'zip_code'], dtype='object')
```

Step 9. How is the dataset indexed?

```
In [71]: data.index
```

```
Out[71]: Int64Index([  1,   2,   3,   4,   5,   6,   7,   8,   9,  10,
                     ...
                     934, 935, 936, 937, 938, 939, 940, 941, 942, 943],
                    dtype='int64', name='user_id', length=943)
```

Loading [MathJax]/extensions/Safe.js

Step 10. What is the data type of each column?

```
In [73]: data.dtype
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[73], line 1
----> 1 data.dtype

File ~\anaconda3\lib\site-packages\pandas\core\generic.py:5902, in NDFrame.__
getattr__(self, name)
   5895 if (
   5896     name not in self._internal_names_set
   5897     and name not in self._metadata
   5898     and name not in self._accessors
   5899     and self._info_axis._can_hold_identifiers_and_holds_name(name)
   5900 ):
   5901     return self[name]
-> 5902 return object.__getattribute__(self, name)

AttributeError: 'DataFrame' object has no attribute 'dtype'
```

Step 11. Print only the occupation column

```
In [27]: data["occupation"]
```

```
Out[27]: user_id
1         technician
2              other
3             writer
4         technician
5              other
             ...
939          student
940    administrator
941          student
942        librarian
943          student
Name: occupation, Length: 943, dtype: object
```

Step 12. How many different occupations are in this dataset?

```
In [28]: d=data["occupation"]
         d
```

Loading [MathJax]/extensions/Safe.js

Out[28]:

```
user_id
1         technician
2              other
3             writer
4         technician
5              other
           ...
939          student
940    administrator
941          student
942        librarian
943          student
Name: occupation, Length: 943, dtype: object
```

Step 13. What is the most frequent occupation?

In [29]:

```
d=data["occupation"].value_counts()
d
```

Out[29]:

```
student          196
other            105
educator          95
administrator     79
engineer          67
programmer        66
librarian         51
writer            45
executive         32
scientist         31
artist            28
technician        27
marketing         26
entertainment     18
healthcare        16
retired           14
lawyer            12
salesman          12
none               9
homemaker          7
doctor             7
Name: occupation, dtype: int64
```

Step 14. Summarize the DataFrame.

In [30]:

```
data.describe()
```

Out[30]:

	age
count	943.000000
mean	34.051962
std	12.192740
min	7.000000
25%	25.000000
50%	31.000000
75%	43.000000
max	73.000000

Step 15. Summarize all the columns

In [74]:
```
data.describe(include="all")
```

Out[74]:

	age	gender	occupation	zip_code
count	943.000000	943	943	943
unique	NaN	2	21	795
top	NaN	M	student	55414
freq	NaN	670	196	9
mean	34.051962	NaN	NaN	NaN
std	12.192740	NaN	NaN	NaN
min	7.000000	NaN	NaN	NaN
25%	25.000000	NaN	NaN	NaN
50%	31.000000	NaN	NaN	NaN
75%	43.000000	NaN	NaN	NaN
max	73.000000	NaN	NaN	NaN

Step 16. Summarize only the occupation column

In [88]:
```
data.describe(include="all")
```

Loading [MathJax]/extensions/Safe.js

Out[88]:

	age	gender	occupation	zip_code
count	943.000000	943	943	943
unique	NaN	2	21	795
top	NaN	M	student	55414
freq	NaN	670	196	9
mean	34.051962	NaN	NaN	NaN
std	12.192740	NaN	NaN	NaN
min	7.000000	NaN	NaN	NaN
25%	25.000000	NaN	NaN	NaN
50%	31.000000	NaN	NaN	NaN
75%	43.000000	NaN	NaN	NaN
max	73.000000	NaN	NaN	NaN

Step 17. What is the mean age of users?

In [33]:

```
import numpy as np
```

In [40]:

```
a=data["age"]
a
```

Out[40]:

```
user_id
1      24
2      53
3      23
4      24
5      33
       ..
939    26
940    32
941    20
942    48
943    22
Name: age, Length: 943, dtype: int64
```

In [41]:

```
np.mean(a)
```

Out[41]:

```
34.05196182396607
```

Step 18. What is the age with least occurrence?

In [91]:

```
ag=data["age"].value_counts()
ag
```

Loading [MathJax]/extensions/Safe.js

```
Out[91]: 30    39
         25    38
         22    37
         28    36
         27    35
               ..
         7      1
         66     1
         11     1
         10     1
         73     1
         Name: age, Length: 61, dtype: int64
```

Loading [MathJax]/extensions/Safe.js