1. Write a Pandas program to select the specified columns and rows from a given data frame. Go to the editorSample Python dictionary data and list labels:

```
In [1]:
import pandas as pd
import numpy as np
exam_data  = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
        'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
        'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df = pd.DataFrame(exam_data , index=labels)
print("Select specific columns and rows:")
print(df.iloc[[1, 3, 5, 6], [0, 1]])
```

```
Select specific columns and rows:
      name  score
b     Dima    9.0
d    James    NaN
f  Michael   20.0
g  Matthew   14.5
```

2. Use Crime dataset from LMS

I) find the aggregations like all moments of business decisions for all columns,value counts.

II) do the plottings like plottings like histogram, boxplot, scatterplot, barplot, piechart,dot chart.

```
In [3]:
import numpy as np
import pandas as pd
crime_data = pd.read_csv("C:\\Users\\DELL\\Downloads\\crime_data.csv")
```

```
In [4]: crime_data.Murder.count()
```

Out[4]: 50

```
In [5]: crime_data.Assault.sum()
```

Out[5]: 8538

```
In [6]: crime_data.UrbanPop.min()
```

Out[6]: 32

```
In [7]: crime_data.UrbanPop.max()
```

Out[7]: 91

```
In [8]: crime_data.Rape.mean()
```

Out[8]: 21.231999999999992

```
In [9]: crime_data.Rape.median()
```

Out[9]: 20.1

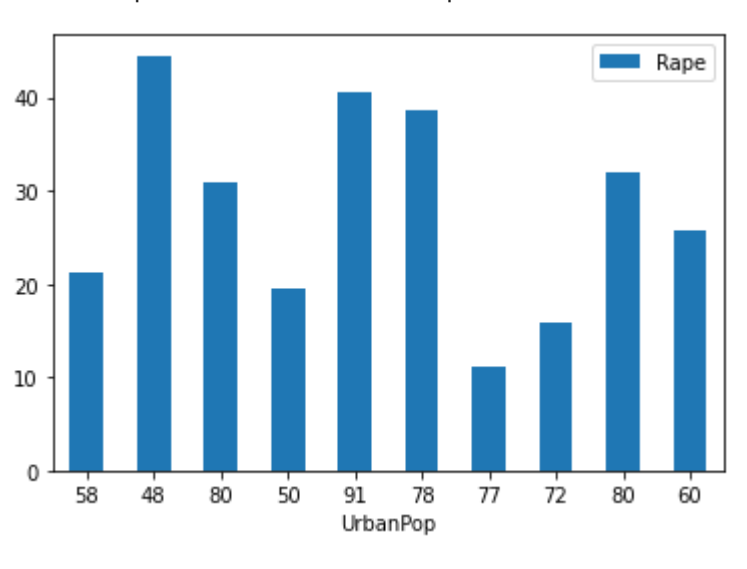
```
In [10]:
crime_data_10 = crime_data.head(10)
crime_data_10
```

Out[10]:

	Unnamed: 0	Murder	Assault	UrbanPop	Rape
0	Alabama	13.2	236	58	21.2
1	Alaska	10.0	263	48	44.5
2	Arizona	8.1	294	80	31.0
3	Arkansas	8.8	190	50	19.5
4	California	9.0	276	91	40.6
5	Colorado	7.9	204	78	38.7
6	Connecticut	3.3	110	77	11.1
7	Delaware	5.9	238	72	15.8
8	Florida	15.4	335	80	31.9
9	Georgia	17.4	211	60	25.8

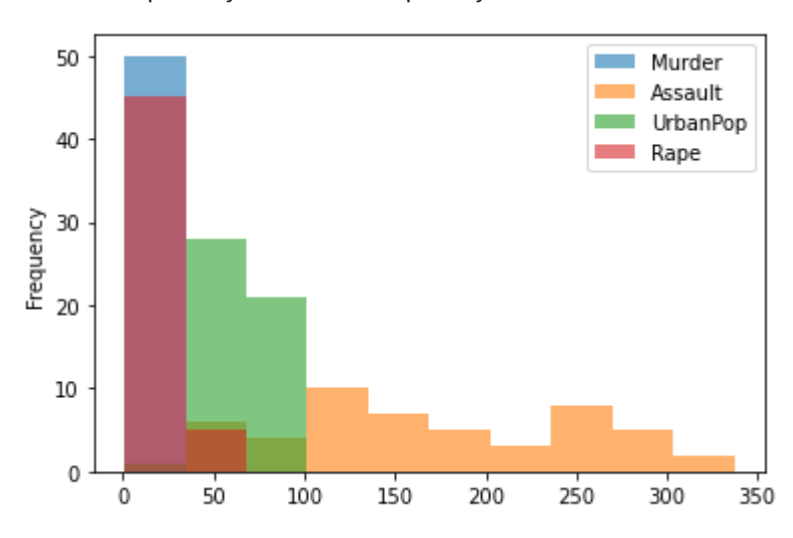
```
In [11]:
#barplot
x = crime_data_10.plot.bar(x='UrbanPop', y='Rape', rot=0)
x
```

Out[11]: <AxesSubplot:xlabel='UrbanPop'>

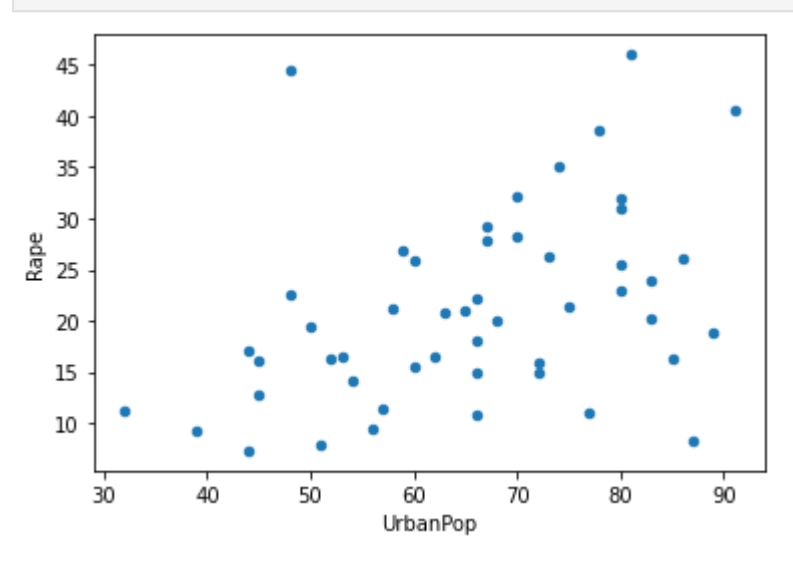



```
In [12]:
#histogram
crime_data.plot.hist(alpha=0.6)
```

Out[12]: <AxesSubplot:ylabel='Frequency'>

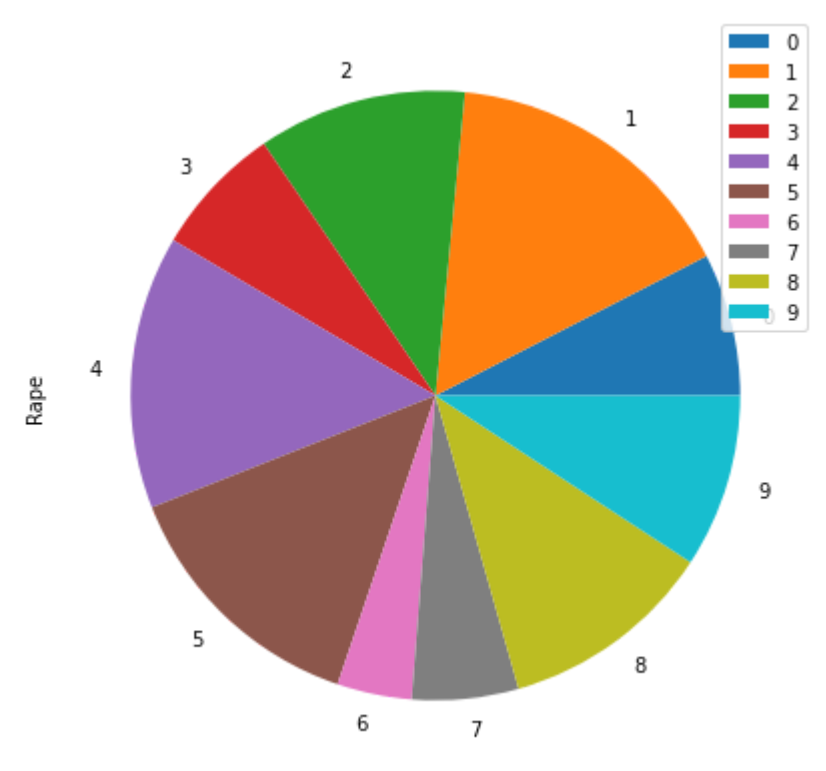



```
In [13]:
#scatterplot
crime_data.plot.scatter(x='UrbanPop', y='Rape');
```

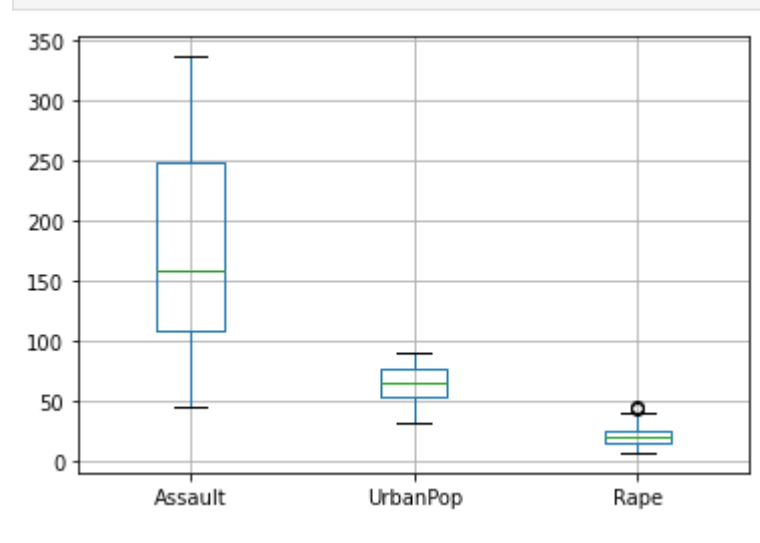



```
In [14]:
#piechart
crime_data_10.plot.pie(y='Rape', figsize=(7, 7))
```

Out[14]: <AxesSubplot:ylabel='Rape'>




```
In [15]:
#boxplot
boxplot = crime_data.boxplot(column=['Assault', 'UrbanPop', 'Rape'])
```




```
In [ ]:
#dotplots
import plotly.express as px
fig = px.scatter(crime_data, x="UrbanPop", y="Rape",
                 title="UrbanPop vs Rape",
                 labels={"UrbanPop":"URBANPOP DETAILS"}
                )
fig.show()
```

3. use mtcars dataset from LMS

A) delete/ drop rows-10 to 15 of all columns

B)drop the VOL column

C)write the forloop to get value_counts of all cloumns

```
In [17]:
mtcars = pd.read_csv("C:\\Users\\DELL\\Downloads\\mtcars.csv")
mtcars.head()
```

Out[17]:

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

```
In [18]:
#A:Ans
mtcars = mtcars.drop(mtcars.index[[10,11,12,13,14,15]])
mtcars.head()
```

Out[18]:

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

```
In [20]:
# B:Ans
mtcars = mtcars.drop(columns=mtcars.columns[3])
mtcars.head()
```

Out[20]:

	mpg	cyl	disp	wt	qsec	vs	am	gear	carb
0	21.0	6	160.0	2.620	16.46	0	1	4	4
1	21.0	6	160.0	2.875	17.02	0	1	4	4
2	22.8	4	108.0	2.320	18.61	1	1	4	1
3	21.4	6	258.0	3.215	19.44	1	0	3	1
4	18.7	8	360.0	3.440	17.02	0	0	3	2

```
In [21]:
#C:Ans
for col in mtcars:
    print('-' * 40 + col + '-' * 40 , end=' - ')
    display(mtcars[col].value_counts().head(10))
```

```
----------------------------------------mpg---------------------------------------- -
30.4    2
21.4    2
19.2    2
22.8    2
21.0    2
15.8    1
21.5    1
15.5    1
26.0    1
15.2    1
Name: mpg, dtype: int64
----------------------------------------cyl---------------------------------------- -
4    11
8     9
6     6
Name: cyl, dtype: int64
----------------------------------------disp---------------------------------------- -
160.0    2
360.0    2
71.1     1
108.0    1
258.0    1
225.0    1
440.0    1
318.0    1
304.0    1
350.0    1
Name: disp, dtype: int64
----------------------------------------wt---------------------------------------- -
3.440    2
3.570    2
2.780    1
3.435    1
2.320    1
5.345    1
2.465    1
3.460    1
1.513    1
2.770    1
Name: wt, dtype: int64
----------------------------------------qsec---------------------------------------- -
17.02    2
18.90    1
18.30    1
20.00    1
14.50    1
19.47    1
15.84    1
17.30    1
20.01    1
15.41    1
Name: qsec, dtype: int64
----------------------------------------vs---------------------------------------- -
1    13
0    13
Name: vs, dtype: int64
----------------------------------------am---------------------------------------- -
1    13
0    13
Name: am, dtype: int64
----------------------------------------gear---------------------------------------- -
4    11
3    10
5     5
Name: gear, dtype: int64
----------------------------------------carb---------------------------------------- -
2    10
4     7
1     7
8     1
6     1
Name: carb, dtype: int64
```

4. Use Bank Dataset from LMS

A)change all the categorical columns into numerical by creating Dummies and using label encoder.

B) rename all the column names DF

C) Rename only one specific column in DF

```
In [22]:
from sklearn.preprocessing import LabelEncoder
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
```

```
In [23]:
df=pd.read_csv("C:\\Users\\DELL\\Downloads\\bank-full.csv")
df
```

Out[23]:

	age;"job";"marital";"education";"default";"balance";"housing";"loan";"contact";"day";"month";"duration";"campaign";"pdays";"previous";"poutcome";"y"
0	58;"management";"married";"tertiary";"no";2143...
1	44;"technician";"single";"secondary";"no";29;"...
2	33;"entrepreneur";"married";"secondary";"no";2...
3	47;"blue-collar";"married";"unknown";"no";1506...
4	33;"unknown";"single";"unknown";"no";1;"no";"n...
...	...
45206	51;"technician";"married";"tertiary";"no";825;...
45207	71;"retired";"divorced";"primary";"no";1729;"n...
45208	72;"retired";"married";"secondary";"no";5715;"...
45209	57;"blue-collar";"married";"secondary";"no";66...
45210	37;"entrepreneur";"married";"secondary";"no";2...

45211 rows × 1 columns

```
In [24]:
df=df.iloc[:,:].values
df
```

```
Out[24]:
array([['58;"management";"married";"tertiary";"no";2143;"yes";"no";"unknown";5;"may";261;1;-1;0;"unknown";"no"'],
       ['44;"technician";"single";"secondary";"no";29;"yes";"no";"unknown";5;"may";151;1;-1;0;"unknown";"no"'],
       ['33;"entrepreneur";"married";"secondary";"no";2;"yes";"yes";"unknown";5;"may";76;1;-1;0;"unknown";"no"'],
       ...,
       ['72;"retired";"married";"secondary";"no";5715;"no";"no";"cellular";17;"nov";1127;5;184;3;"success";"yes"'],
       ['57;"blue-collar";"married";"secondary";"no";668;"no";"no";"telephone";17;"nov";508;4;-1;0;"unknown";"no"'],
       ['37;"entrepreneur";"married";"secondary";"no";2971;"no";"no";"cellular";17;"nov";361;2;188;11;"other";"no"']],
      dtype=object)
```

```
In [ ]:
m_status=LabelEncoder()
df[:, ]=m_status.fit_transform(df[:, ])
df
```

```
In [25]:
df=pd.DataFrame(df)
df
```

Out[25]:

	0
0	58;"management";"married";"tertiary";"no";2143...
1	44;"technician";"single";"secondary";"no";29;"...
2	33;"entrepreneur";"married";"secondary";"no";2...
3	47;"blue-collar";"married";"unknown";"no";1506...
4	33;"unknown";"single";"unknown";"no";1;"no";"n...
...	...
45206	51;"technician";"married";"tertiary";"no";825;...
45207	71;"retired";"divorced";"primary";"no";1729;"n...
45208	72;"retired";"married";"secondary";"no";5715;"...
45209	57;"blue-collar";"married";"secondary";"no";66...
45210	37;"entrepreneur";"married";"secondary";"no";2...

45211 rows × 1 columns

```
In [26]: ct= ColumnTransformer(transformers=[('encode',OneHotEncoder(),[2])],remainder='passthrough')
```

```
In [ ]:
df=ct.fit_transform(df)
df
```

```
In [28]:
df=pd.DataFrame(df)
df
```

Out[28]:

	0
0	58;"management";"married";"tertiary";"no";2143...
1	44;"technician";"single";"secondary";"no";29;"...
2	33;"entrepreneur";"married";"secondary";"no";2...
3	47;"blue-collar";"married";"unknown";"no";1506...
4	33;"unknown";"single";"unknown";"no";1;"no";"n...
...	...
45206	51;"technician";"married";"tertiary";"no";825;...
45207	71;"retired";"divorced";"primary";"no";1729;"n...
45208	72;"retired";"married";"secondary";"no";5715;"...
45209	57;"blue-collar";"married";"secondary";"no";66...
45210	37;"entrepreneur";"married";"secondary";"no";2...

45211 rows × 1 columns

```
In [29]:
#B Rename all columns as df
df.rename(columns={0:'df1', 1:'df2', 2:'df3', 3:'df4',4:'df5', 5: 'df6',6:'df7',7:'df8',
                   8:'df9',9:'df10', 10:'df11',11:'df12',12:'df13',
                   13:'df14',14:'df15', 15:'df16',16:'df17',17:'df18',18:'df19'})
df
```

Out[29]:

	0
0	58;"management";"married";"tertiary";"no";2143...
1	44;"technician";"single";"secondary";"no";29;"...
2	33;"entrepreneur";"married";"secondary";"no";2...
3	47;"blue-collar";"married";"unknown";"no";1506...
4	33;"unknown";"single";"unknown";"no";1;"no";"n...
...	...
45206	51;"technician";"married";"tertiary";"no";825;...
45207	71;"retired";"divorced";"primary";"no";1729;"n...
45208	72;"retired";"married";"secondary";"no";5715;"...
45209	57;"blue-collar";"married";"secondary";"no";66...
45210	37;"entrepreneur";"married";"secondary";"no";2...

45211 rows × 1 columns

```
In [30]:
#C rename only one column in df
df.rename(columns={4:'age'})
df.head
```

```
Out[30]:
<bound method NDFrame.head of                                                   0
0      58;"management";"married";"tertiary";"no";2143...
1      44;"technician";"single";"secondary";"no";29;"...
2      33;"entrepreneur";"married";"secondary";"no";2...
3      47;"blue-collar";"married";"unknown";"no";1506...
4      33;"unknown";"single";"unknown";"no";1;"no";"n...
...                                                  ...
45206  51;"technician";"married";"tertiary";"no";825;...
45207  71;"retired";"divorced";"primary";"no";1729;"n...
45208  72;"retired";"married";"secondary";"no";5715;"...
45209  57;"blue-collar";"married";"secondary";"no";66...
45210  37;"entrepreneur";"married";"secondary";"no";2...

[45211 rows x 1 columns]>
```

```
In [ ]:
```