Data Science Course Project

In [1]:

```
#import libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import plotly.express as px
import matplotlib.pyplot as plt
```

In [2]:

```
#step 1 Reading dataset using Pandas Library
df=pd.read_csv("wage.csv")
df
```

Out[2]:

	married	hourly_wage	years_in_education	years_in_employment	num_dependents	gender	race
0	1.0	3.24	12.0	2.0	3.0	female	white
1	0.0	3.00	11.0	0.0	2.0	male	white
2	1.0	6.00	8.0	28.0	0.0	male	white
3	1.0	5.30	12.0	2.0	1.0	male	white
4	1.0	8.75	16.0	8.0	0.0	male	white
...	...	...	...	...	...	...	...
520	1.0	15.00	16.0	2.0	2.0	female	white
521	0.0	2.27	10.0	0.0	3.0	female	white
522	1.0	4.67	15.0	18.0	3.0	male	white
523	1.0	11.56	16.0	1.0	0.0	male	white
524	0.0	NaN	14.0	4.0	2.0	female	nonwhite

525 rows × 7 columns

In [3]:

```
#step 2 Reading describtive statistics like min,max,std,mean and quartile(25%,50%,75%)
df.describe()
```

Out[3]:

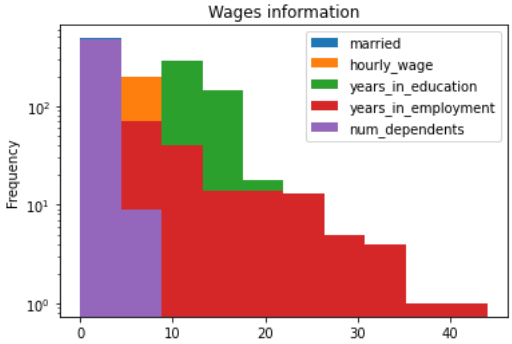
	married	hourly_wage	years_in_education	years_in_employment	num_dependents
count	522.000000	517.000000	522.000000	519.000000	520.000000
mean	0.609195	5.917737	12.557471	5.152216	1.044231
std	0.488399	3.699058	2.757219	7.257133	1.258484
min	0.000000	0.530000	0.000000	0.000000	0.000000
25%	0.000000	3.350000	12.000000	0.000000	0.000000
50%	1.000000	4.670000	12.000000	2.000000	1.000000
75%	1.000000	6.880000	14.000000	7.000000	2.000000
max	1.000000	24.980000	18.000000	44.000000	6.000000

In [24]:

```
#Step 3 visualization dataset variables using Histogram graph
df.plot(kind="hist", title='Wages information', logy=True)
```

Out[24]:

```
<AxesSubplot:title={'center':'Wages information'}, ylabel='Frequency'>
```



In [4]:

```
#step 4 checking missing values using isnull() method and handling missing values using dropna() method to cleaning data.
df.isnull().sum()
```

Out[4]:

```
married                 3
hourly_wage             8
years_in_education      3
years_in_employment     6
num_dependents          5
gender                  4
race                   10
dtype: int64
```

In [5]:

```
df.dropna(inplace=True, axis="rows")
```

In [6]:

```
df.isnull().sum()
```

Out[6]:

```
married                 0
hourly_wage             0
years_in_education      0
years_in_employment     0
num_dependents          0
gender                  0
race                    0
dtype: int64
```

In [9]:

```
#Step 5 identify how many values are unique each column or variable
df.nunique()
```

Out[9]:

```
married                   2
hourly_wage             227
years_in_education       18
years_in_employment      34
num_dependents            7
gender                    2
race                      2
dtype: int64
```

In [10]:

```
#Step 6 Retrieve the first subset of dataset and descriptive statistics based on first subset
sub1=df[(df['gender']=="male") & (df['years_in_education'] > 12)]
sub1
```

Out[10]:

	married	hourly_wage	years_in_education	years_in_employment	num_dependents	gender	race
4	1.0	8.75	16.0	8.0	0.0	male	white
5	0.0	11.25	18.0	7.0	0.0	male	white
8	1.0	18.18	17.0	21.0	0.0	male	white
14	1.0	17.33	16.0	0.0	1.0	male	white
24	1.0	9.56	16.0	3.0	1.0	male	nonwhite
...	...	...	...	...	...	...	...
510	1.0	4.38	13.0	0.0	1.0	male	nonwhite
516	1.0	9.33	14.0	11.0	3.0	male	white
518	1.0	4.75	13.0	1.0	0.0	male	white
522	1.0	4.67	15.0	18.0	3.0	male	white
523	1.0	11.56	16.0	1.0	0.0	male	white

119 rows × 7 columns

In [11]:

```
sub1.describe()
```

Out[11]:

	married	hourly_wage	years_in_education	years_in_employment	num_dependents
count	119.000000	119.000000	119.000000	119.000000	119.000000
mean	0.705882	8.314202	15.344538	6.117647	0.773109
std	0.457572	4.329249	1.492563	8.314961	1.036920
min	0.000000	2.920000	13.000000	0.000000	0.000000
25%	0.000000	5.000000	14.000000	0.000000	0.000000
50%	1.000000	6.880000	16.000000	3.000000	0.000000
75%	1.000000	10.665000	16.000000	7.000000	1.000000
max	1.000000	22.860000	18.000000	39.000000	4.000000

In [12]:

```
# Retrieve the second subset of dataset and descriptive statistics based on second subset
sub2=df[(df['race']=="white") & (df['years_in_education'] >= 8)]
sub2
```

Out[12]:

	married	hourly_wage	years_in_education	years_in_employment	num_dependents	gender	race
0	1.0	3.24	12.0	2.0	3.0	female	white
1	0.0	3.00	11.0	0.0	2.0	male	white
2	1.0	6.00	8.0	28.0	0.0	male	white
3	1.0	5.30	12.0	2.0	1.0	male	white
4	1.0	8.75	16.0	8.0	0.0	male	white
...	...	...	...	...	...	...	...
519	0.0	5.65	12.0	0.0	0.0	male	white
520	1.0	15.00	16.0	2.0	2.0	female	white
521	0.0	2.27	10.0	0.0	3.0	female	white
522	1.0	4.67	15.0	18.0	3.0	male	white
523	1.0	11.56	16.0	1.0	0.0	male	white

424 rows × 7 columns

In [13]:

```
sub2.describe()
```

Out[13]:

	married	hourly_wage	years_in_education	years_in_employment	num_dependents
count	424.000000	424.000000	424.000000	424.000000	424.000000
mean	0.620283	6.010708	12.915094	4.981132	0.948113
std	0.485890	3.728808	2.328233	7.262366	1.139095
min	0.000000	0.530000	8.000000	0.000000	0.000000
25%	0.000000	3.400000	12.000000	0.000000	0.000000
50%	1.000000	4.790000	12.000000	2.000000	0.500000
75%	1.000000	7.170000	14.000000	6.000000	2.000000
max	1.000000	22.860000	18.000000	44.000000	5.000000

In [68]:

```
#step 7 do not have categorical dataset
```

In [20]:

```
#step 8 create linear regression model to training and testing dataset
#to predictions the future and the resualt decision making
x= np.array(df.drop(["married","hourly_wage","gender","race",], axis=1))
y=np.array(df["hourly_wage"])
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size=0.4)
```

In [21]:

```
model = LinearRegression()
model.fit(xtrain,ytrain)
print(model.score(xtest,ytest))
```

```
0.3188975789641907
```

In [22]:

```
future = np.array([[12,4,1]])
print(model.predict(future))
```

```
[5.38769803]
```

In [ ]:

```

```