In [1]:

```
import pandas as pd,numpy as np,os
```

In [2]:

```
os.chdir('D:\machine learning\Raw data')
```

In [4]:

```
os.listdir()
```

Out[4]:

```
['a SQL',
 'Advertising.csv',
 'automobile_data.sas7bdat',
 'Automobile_data.xlsx',
 'Automobile_data2.csv',
 'Automobile_data2.xlsx',
 'bank_data.sas7bdat',
 'bigmart_data.csv',
 'Book1.xlsx',
 'carsnew2.xlsx',
 'casnew.csv',
 'ccapp_data.sas7bdat',
 'churn.csv',
 'churn.xlsx',
 'churn2.csv',
 'churn_data.pickle',
 'churn_data.xlsx',
 'chur_12.xlsx',
 'cleaned data',
 'concrete_data.csv',
 'Covid_data.xlsx',
 'creditCardFraudEDA-checkpoint.ipynb',
 'CREDIT_DISCOVERY_FOR_DS.csv',
 'data.csv',
 'data.sav',
 'dubai_refreshments_final.sas7bdat',
 'Ecommerce_data_p1vl9.xlsx',
 'employees.csv',
 'employee_detail.sas7bdat',
 'german.data.txt',
 'german_credit_data.csv',
 'Gold.xlsx',
 'House Price.csv',
 'House_Price_Scoring.csv',
 'insurance_claims.csv',
 'KMeans.sav',
 'loan_data.sas7bdat',
 'machine learning',
 'MANJU.csv',
 'marks',
 'merging',
 'nortel.csv',
 'Order01.csv',
 'Orders.csv',
 'payroll.sas7bdat',
 'payroll2.csv',
 'Problem Statement.docx',
 'state gdp',
 'test.csv',
 'Titanic_data.csv',
 'train.csv',
 'user devise',
 'user_usage.xlsx']
```

In [9]:

```
payroll=pd.read_sas('payroll.sas7bdat')
```

In [10]:

```
payroll.head()
```

Out[10]:

	VAR1	Employee_ID	Employee_Gender	Salary	Birth_Date	Employee_Hire_Date	Depender
0	0.0	120101.0	b'M'	163040.0	1978-08-18	2005-07-01	
1	1.0	120102.0	b'M'	108255.0	1971-08-11	1991-06-01	
2	2.0	120103.0	b'M'	87975.0	1951-01-22	1976-01-01	
3	3.0	120104.0	b'F'	46230.0	1956-05-11	1983-01-01	
4	4.0	120105.0	b'F'	27110.0	1976-12-21	2001-05-01	

In [11]:

```
payroll=pd.read_sas('payroll.sas7bdat',encoding='latin-1') # encoding use to remove the noi
```

In [12]:

```
payroll
```

Out[12]:

	VAR1	Employee_ID	Employee_Gender	Salary	Birth_Date	Employee_Hire_Date	Dep
0	0.0	120101.0	M	163040.0	1978-08-18	2005-07-01	
1	1.0	120102.0	M	108255.0	1971-08-11	1991-06-01	
2	2.0	120103.0	M	87975.0	1951-01-22	1976-01-01	
3	3.0	120104.0	F	46230.0	1956-05-11	1983-01-01	
4	4.0	120105.0	F	27110.0	1976-12-21	2001-05-01	
...	...	...	...	...	...	...	
419	419.0	121144.0	F	83505.0	1966-06-28	1993-11-01	
420	420.0	121145.0	M	84260.0	1951-11-22	1978-04-01	
421	421.0	121146.0	F	29320.0	1988-12-09	2008-04-01	
422	422.0	121147.0	F	29145.0	1971-05-28	1989-09-01	
423	423.0	121148.0	M	52930.0	1971-01-01	2000-01-01	

424 rows × 9 columns

In [15]:

```
payroll2=payroll.drop(['VAR1','avg_sal_per_head','test'],axis=1)
payroll2
```

Out[15]:

	Employee_ID	Employee_Gender	Salary	Birth_Date	Employee_Hire_Date	Dependents
0	120101.0	M	163040.0	1978-08-18	2005-07-01	0.0
1	120102.0	M	108255.0	1971-08-11	1991-06-01	2.0
2	120103.0	M	87975.0	1951-01-22	1976-01-01	1.0
3	120104.0	F	46230.0	1956-05-11	1983-01-01	1.0
4	120105.0	F	27110.0	1976-12-21	2001-05-01	0.0
...	...	...	...	...	...	...
419	121144.0	F	83505.0	1966-06-28	1993-11-01	3.0
420	121145.0	M	84260.0	1951-11-22	1978-04-01	2.0
421	121146.0	F	29320.0	1988-12-09	2008-04-01	1.0
422	121147.0	F	29145.0	1971-05-28	1989-09-01	2.0
423	121148.0	M	52930.0	1971-01-01	2000-01-01	1.0

424 rows × 6 columns

In [17]:

```
payroll2.shape
```

Out[17]:

```
(424, 6)
```

In [18]:

```
payroll2.head()
```

Out[18]:

	Employee_ID	Employee_Gender	Salary	Birth_Date	Employee_Hire_Date	Dependents
0	120101.0	M	163040.0	1978-08-18	2005-07-01	0.0
1	120102.0	M	108255.0	1971-08-11	1991-06-01	2.0
2	120103.0	M	87975.0	1951-01-22	1976-01-01	1.0
3	120104.0	F	46230.0	1956-05-11	1983-01-01	1.0
4	120105.0	F	27110.0	1976-12-21	2001-05-01	0.0

In [19]:

```
payroll2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 424 entries, 0 to 423
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype         
---  ------              --------------  -----         
 0   Employee_ID         424 non-null    float64       
 1   Employee_Gender     424 non-null    object        
 2   Salary              424 non-null    float64       
 3   Birth_Date          424 non-null    datetime64[ns]
 4   Employee_Hire_Date  424 non-null    datetime64[ns]
 5   Dependents          424 non-null    float64       
dtypes: datetime64[ns](2), float64(3), object(1)
memory usage: 20.0+ KB
```

In [20]:

```
payroll2.describe()
```

Out[20]:

	Employee_ID	Salary	Dependents
count	424.000000	424.000000	424.000000
mean	120701.172170	38041.509434	1.125000
std	364.581266	31741.136023	1.146868
min	120101.000000	22710.000000	0.000000
25%	120266.750000	26742.500000	0.000000
50%	120761.500000	28685.000000	1.000000
75%	121042.250000	36386.250000	2.000000
max	121148.000000	433800.000000	3.000000

In [23]:

```
payroll2[['Salary','Dependents']].describe()
```

Out[23]:

	Salary	Dependents
count	424.000000	424.000000
mean	38041.509434	1.125000
std	31741.136023	1.146868
min	22710.000000	0.000000
25%	26742.500000	0.000000
50%	28685.000000	1.000000
75%	36386.250000	2.000000
max	433800.000000	3.000000

In [24]:

```
payroll2[payroll2['Employee_Gender']=='M']['Salary'].std()
```

Out[24]:

```
39550.523226604106
```

In [27]:

```
payroll2[payroll2['Employee_Gender']=='F']['Salary'].std()
```

Out[27]:

```
17944.52894525894
```

In [28]:

```
payroll2[payroll2['Employee_Gender']=='M']['Salary'].mean()
```

Out[28]:

```
40050.042918454936
```

In [29]:

```
payroll2[payroll2['Employee_Gender']=='M']['Salary'].std()/payroll2[payroll2['Employee_Gend
```

Out[29]:

```
98.75276115716557
```

In [30]:

```
payroll2.describe(include='number')
```

Out[30]:

	Employee_ID	Salary	Dependents
count	424.000000	424.000000	424.000000
mean	120701.172170	38041.509434	1.125000
std	364.581266	31741.136023	1.146868
min	120101.000000	22710.000000	0.000000
25%	120266.750000	26742.500000	0.000000
50%	120761.500000	28685.000000	1.000000
75%	121042.250000	36386.250000	2.000000
max	121148.000000	433800.000000	3.000000

In [32]:

```
payroll2['Salary'].describe(percentiles=[.0,.1,.2,.3,.4,.5,.6,.7,.8,.9])
```

Out[32]:

```
count       424.000000
mean      38041.509434
std       31741.136023
min       22710.000000
0%        22710.000000
10%       25912.500000
20%       26548.000000
30%       26953.500000
40%       27481.000000
50%       28685.000000
60%       30781.000000
70%       33572.000000
80%       43600.000000
90%       54454.000000
max      433800.000000
Name: Salary, dtype: float64
```

In [33]:

```
np.percentile(payroll2['Salary'],[.0,.1,.2,.3,.4,.5,.6,.7,.8,.9,100])
```

Out[33]:

```
array([ 22710.   ,  23262.015,  23814.03 ,  24017.69 ,  24021.92 ,
        24033.625,  24065.35 ,  24097.075,  24211.36 ,  24334.03 ,
       433800.   ])
```

In [34]:

```
np.percentile(payroll2['Salary'],[25,50,75,100])
```

Out[34]:

```
array([ 26742.5 ,  28685.  ,  36386.25, 433800.  ])
```

In [35]:

```
payroll2.groupby('Employee_Gender') ['Salary'].mean()
```

Out[35]:

```
Employee_Gender
F    35591.308901
M    40050.042918
Name: Salary, dtype: float64
```

In [36]:

```
payroll2.groupby(['Employee_Gender','Dependents']) ['Salary'].mean()
```

Out[36]:

```
Employee_Gender  Dependents
F                0.0           34622.434211
                 1.0           35090.945946
                 2.0           38055.750000
                 3.0           35422.105263
M                0.0           37269.174757
                 1.0           44967.500000
                 2.0           43282.625000
                 3.0           37455.789474
Name: Salary, dtype: float64
```

In [45]:

```
payroll_stat=payroll2.groupby('Employee_Gender') ['Salary'].describe()
payroll_stat
```

Out[45]:

Employee_Gender	count	mean	std	min	25%	50%	75%	
F	191.0	35591.308901	17944.528945	24015.0	26835.0	28800.0	36400.0	2078
M	233.0	40050.042918	39550.523227	22710.0	26625.0	28615.0	36370.0	4338

In [44]:

```
type('payroll_stat')
```

Out[44]:

```
str
```

In [46]:

```
payroll_stat['cv']=payroll_stat['std']/payroll_stat['mean']*100
```

In [47]:

```
payroll_stat
```

Out[47]:

Employee_Gender	count	mean	std	min	25%	50%	75%	
F	191.0	35591.308901	17944.528945	24015.0	26835.0	28800.0	36400.0	2078
M	233.0	40050.042918	39550.523227	22710.0	26625.0	28615.0	36370.0	4338

In [54]:

```
pay3=payroll2.groupby(['Dependents']) ['Salary'].describe()
pay3
```

Out[54]:

Dependents	count	mean	std	min	25%	50%	75%	max
0.0	179.0	36145.418994	19170.916247	24015.0	26892.50	29625.0	40057.50	194885.0
1.0	89.0	40861.516854	49493.791578	24390.0	26605.00	28585.0	34850.00	433800.0
2.0	80.0	40669.187500	37569.606677	24100.0	26911.25	28592.5	36327.50	268455.0
3.0	76.0	36438.947368	20519.835904	22710.0	26595.00	28335.0	36966.25	161290.0

In [55]:

```
pay3['cv']=pay3['std']/pay3['mean']*100
```

In [56]:

```
pay3
```

Out[56]:

Dependents	count	mean	std	min	25%	50%	75%	max
0.0	179.0	36145.418994	19170.916247	24015.0	26892.50	29625.0	40057.50	194885.0
1.0	89.0	40861.516854	49493.791578	24390.0	26605.00	28585.0	34850.00	433800.0
2.0	80.0	40669.187500	37569.606677	24100.0	26911.25	28592.5	36327.50	268455.0
3.0	76.0	36438.947368	20519.835904	22710.0	26595.00	28335.0	36966.25	161290.0

In [60]:

```
pay3.reset_index(inplace=True)
pay3
```

Out[60]:

	index	Dependents	count	mean	std	min	25%	50%	75%
0	0	0.0	179.0	36145.418994	19170.916247	24015.0	26892.50	29625.0	40057.50
1	1	1.0	89.0	40861.516854	49493.791578	24390.0	26605.00	28585.0	34850.00
2	2	2.0	80.0	40669.187500	37569.606677	24100.0	26911.25	28592.5	36327.50
3	3	3.0	76.0	36438.947368	20519.835904	22710.0	26595.00	28335.0	36966.25

In [62]:

```
pay3.set_index('count').reset_index()
```

Out[62]:

	count	index	Dependents	mean	std	min	25%	50%	75%
0	179.0	0	0.0	36145.418994	19170.916247	24015.0	26892.50	29625.0	40057.50
1	89.0	1	1.0	40861.516854	49493.791578	24390.0	26605.00	28585.0	34850.00
2	80.0	2	2.0	40669.187500	37569.606677	24100.0	26911.25	28592.5	36327.50
3	76.0	3	3.0	36438.947368	20519.835904	22710.0	26595.00	28335.0	36966.25

In [ ]:

```

```