In [1]:
```
import os, pandas as pd, numpy as np
```

In [2]:
```
os.chdir('E:\\Data Science Course\\raw_data')
```

In [3]:
```
payroll=pd.read_csv('payroll.csv')
```

In [9]:
```
payroll.head()
```

Out[9]:

	Sl_no	Employee_ID	Employee_Gender	Salary	Birth_Date	Employee_Hire_Date	Dependents
0	0	120101	M	163040	8/18/1978	7/1/2005	0
1	1	120102	M	108255	8/11/1971	6/1/1991	2
2	2	120103	M	87975	1/22/1951	1/1/1976	1
3	3	120104	F	46230	5/11/1956	1/1/1983	1
4	4	120105	F	27110	12/21/1976	5/1/2001	0

In [8]:
```
payroll.rename(columns={'Unnamed: 0':'Sl_no'},inplace=True)
```

In [ ]:
```
payroll
```

In [5]:
```
payroll.describe()
```

Out[5]:

	Unnamed: 0	Employee_ID	Salary	Dependents
count	424.00000	424.000000	424.000000	424.000000
mean	211.50000	120701.172170	38041.509434	1.125000
std	122.54251	364.581266	31741.136023	1.146868
min	0.00000	120101.000000	22710.000000	0.000000
25%	105.75000	120266.750000	26742.500000	0.000000
50%	211.50000	120761.500000	28685.000000	1.000000
75%	317.25000	121042.250000	36386.250000	2.000000
max	423.00000	121148.000000	433800.000000	3.000000

In [11]:
```
payroll2=pd.read_sas('Payroll_1.sas7bdat',encoding='latin-1')
```

In [14]:
```
payroll2.head()
```

Out[14]:

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

In [16]:
```
payroll3=payroll2.drop(['VAR1','avg_sal_per_head','test'],axis=1)
```

In [17]:
```
payroll3.head()
```

Out[17]:

	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 [18]:
```
payroll3.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 [19]:
```
payroll3.describe()
```

Out[19]:

	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 [21]:
```
payroll3[['Salary','Dependents']].describe()
```

Out[21]:

	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 [22]:
```
payroll4=payroll3[payroll3['Salary']<100000]
```

In [23]:
```
payroll4
```

Out[23]:

	Employee_ID	Employee_Gender	Salary	Birth_Date	Employee_Hire_Date	Dependents
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
5	120106.0	M	26960.0	1946-12-23	1976-01-01	2.0
6	120107.0	F	30475.0	1951-01-21	1976-02-01	2.0
...	...	...	...	...	...	...
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

415 rows × 6 columns

In [24]:
```
payroll4['Salary'].std()/payroll4['Salary'].mean()*100
```

Out[24]: 36.673517934278856

In [26]:
```
len(payroll3[payroll3['Salary']>100000])
```

Out[26]: 9

In [27]:
```
payroll4.shape
```

Out[27]: (415, 6)

In [28]:
```
payroll3[payroll3['Employee_Gender']=='M']['Salary'].std()
```

Out[28]: 39550.523226604106

In [29]:
```
payroll3[payroll3['Employee_Gender']=='F']['Salary'].std()
```

Out[29]: 17944.52894525894

In [30]:
```
payroll3[payroll3['Employee_Gender']=='M']['Salary'].std()/payroll3[payroll3['Emp
```

Out[30]: 98.75276115716557

In [31]:
```
payroll3[payroll3['Employee_Gender']=='F']['Salary'].std()/payroll3[payroll3['Emp
```

Out[31]: 50.41828890140921

In [32]:
```
payroll3.describe(include='all')
```

```
<ipython-input-32-7dca57fcb73b>:1: FutureWarning: Treating datetime data as cat
egorical rather than numeric in `.describe` is deprecated and will be removed i
n a future version of pandas. Specify `datetime_is_numeric=True` to silence thi
s warning and adopt the future behavior now.
  payroll3.describe(include='all')
<ipython-input-32-7dca57fcb73b>:1: FutureWarning: Treating datetime data as cat
egorical rather than numeric in `.describe` is deprecated and will be removed i
n a future version of pandas. Specify `datetime_is_numeric=True` to silence thi
s warning and adopt the future behavior now.
  payroll3.describe(include='all')
```

Out[32]:

	Employee_ID	Employee_Gender	Salary	Birth_Date	Employee_Hire_Date	Depende
count	424.000000	424	424.000000	424	424	424.000
unique	NaN	2	NaN	405	200	
top	NaN	M	NaN	1946-12-23 00:00:00	1976-01-01 00:00:00	
freq	NaN	233	NaN	2	52	
first	NaN	NaN	NaN	1946-01-03 00:00:00	1976-01-01 00:00:00	
last	NaN	NaN	NaN	1990-12-27 00:00:00	2009-01-01 00:00:00	
mean	120701.172170	NaN	38041.509434	NaN	NaN	1.125
std	364.581266	NaN	31741.136023	NaN	NaN	1.146
min	120101.000000	NaN	22710.000000	NaN	NaN	0.000
25%	120266.750000	NaN	26742.500000	NaN	NaN	0.000
50%	120761.500000	NaN	28685.000000	NaN	NaN	1.000
75%	121042.250000	NaN	36386.250000	NaN	NaN	2.000
max	121148.000000	NaN	433800.000000	NaN	NaN	3.000

In [33]:
```
payroll3['Salary'].describe(percentiles=[.1,.2,.3,.4,.6,.7,.8,.9])
```

Out[33]:
```
count       424.000000
mean      38041.509434
std       31741.136023
min       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 [34]:
```
np.percentile(payroll3['Salary'],[25,35,45,65])
```

Out[34]:
```
array([26742.5 , 27250.5 , 28150.75, 31859.75])
```

In [36]:
```
payroll3.groupby('Employee_Gender')['Salary'].mean()
```

Out[36]:
```
Employee_Gender
F    35591.308901
M    40050.042918
Name: Salary, dtype: float64
```

In [49]:
```
pay_gen_stat=payroll3.groupby('Employee_Gender')['Salary'].describe()
```

In [50]:
```
pay_gen_stat
```

Out[50]:

Employee_Gender	count	mean	std	min	25%	50%	75%	max
F	191.0	35591.308901	17944.528945	24015.0	26835.0	28800.0	36400.0	207885.0
M	233.0	40050.042918	39550.523227	22710.0	26625.0	28615.0	36370.0	433800.0

In [51]:
```
type(pay_gen_stat)
```

Out[51]:
```
pandas.core.frame.DataFrame
```

In [52]:
```
pd.DataFrame(pay_gen_stat)
```

Out[52]:

Employee_Gender	count	mean	std	min	25%	50%	75%	max
F	191.0	35591.308901	17944.528945	24015.0	26835.0	28800.0	36400.0	207885.0
M	233.0	40050.042918	39550.523227	22710.0	26625.0	28615.0	36370.0	433800.0

In [53]:
```
type(pay_gen_stat)
```

Out[53]:
```
pandas.core.frame.DataFrame
```

In [54]:
```
pay_gen_stat['CV']=pay_gen_stat['std']/pay_gen_stat['mean']*100
```

In [55]:
```
pay_gen_stat
```

Out[55]:

Employee_Gender	count	mean	std	min	25%	50%	75%	max
F	191.0	35591.308901	17944.528945	24015.0	26835.0	28800.0	36400.0	207885.0
M	233.0	40050.042918	39550.523227	22710.0	26625.0	28615.0	36370.0	433800.0

In [57]:
```
pay_dep_stat=payroll3.groupby('Dependents')['Salary'].describe()
```

In [58]:
```
pay_dep_stat['CV']=pay_dep_stat['std']/pay_dep_stat['mean']*100
```

In [59]:
```
pay_dep_stat
```

Out[59]:

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]:
```
pay_dep_stat.reset_index(inplace=True)
```

In [61]:
```
pay_dep_stat
```

Out[61]:

	Dependents	count	mean	std	min	25%	50%	75%	max
0	0.0	179.0	36145.418994	19170.916247	24015.0	26892.50	29625.0	40057.50	194885.0
1	1.0	89.0	40861.516854	49493.791578	24390.0	26605.00	28585.0	34850.00	433800.0
2	2.0	80.0	40669.187500	37569.606677	24100.0	26911.25	28592.5	36327.50	268455.0
3	3.0	76.0	36438.947368	20519.835904	22710.0	26595.00	28335.0	36966.25	161290.0

In [62]:
```
pay_dep_stat.set_index('count')
```

Out[62]:

count	Dependents	mean	std	min	25%	50%	75%	max	
179.0	0.0	36145.418994	19170.916247	24015.0	26892.50	29625.0	40057.50	194885.0	5
89.0	1.0	40861.516854	49493.791578	24390.0	26605.00	28585.0	34850.00	433800.0	1
80.0	2.0	40669.187500	37569.606677	24100.0	26911.25	28592.5	36327.50	268455.0	9
76.0	3.0	36438.947368	20519.835904	22710.0	26595.00	28335.0	36966.25	161290.0	5

In [63]:
```
pay_dep_stat.reset_index(inplace=True)
```

In [64]:
```
pay_dep_stat
```

Out[64]:

	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	19
1	1	1.0	89.0	40861.516854	49493.791578	24390.0	26605.00	28585.0	34850.00	43
2	2	2.0	80.0	40669.187500	37569.606677	24100.0	26911.25	28592.5	36327.50	26
3	3	3.0	76.0	36438.947368	20519.835904	22710.0	26595.00	28335.0	36966.25	16

In [ ]:
```

```