Basic Analysis using numpy and pandas

Salesworkload dataset

To import library

In [1]:

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

To import dataset

In [2]:

```
d=pd.read_csv(r"C:\Users\user\Downloads\6_Salesworkload1.csv")
d
```

Out[2]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	Hour
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	
...	...	...	...	...	...	...	...	...	
7653	06.2017	9.0	Sweden	29650.0	Gothenburg	12.0	Checkout	6322.323	
7654	06.2017	9.0	Sweden	29650.0	Gothenburg	16.0	Customer Services	4270.479	
7655	06.2017	9.0	Sweden	29650.0	Gothenburg	11.0	Delivery	0	
7656	06.2017	9.0	Sweden	29650.0	Gothenburg	17.0	others	2224.929	
7657	06.2017	9.0	Sweden	29650.0	Gothenburg	18.0	all	39652.2	

7658 rows × 14 columns

To get top 10 record

In [3]:

```
d.head(10)
```

Out[3]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	0.0
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	0.0
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	0.0
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	0.0
5	10.2016	1.0	United Kingdom	88253.0	London (I)	6.0	Meat	8270.316	0.0
6	10.2016	1.0	United Kingdom	88253.0	London (I)	13.0	Food	16468.251	0.0
7	10.2016	1.0	United Kingdom	88253.0	London (I)	7.0	Clothing	4698.471	0.0
8	10.2016	1.0	United Kingdom	88253.0	London (I)	8.0	Household	1183.272	0.0
9	10.2016	1.0	United Kingdom	88253.0	London (I)	9.0	Hardware	2029.815	0.0

To get last 10

In [4]:

```
d.tail(10)
```

Out[4]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	Hours
7648	06.2017	9.0	Sweden	29650.0	Gothenburg	7.0	Clothing	3587.58	
7649	06.2017	9.0	Sweden	29650.0	Gothenburg	8.0	Household	1312.299	
7650	06.2017	9.0	Sweden	29650.0	Gothenburg	9.0	Hardware	1598.676	
7651	06.2017	9.0	Sweden	29650.0	Gothenburg	14.0	Non Food	6498.555	
7652	06.2017	9.0	Sweden	29650.0	Gothenburg	15.0	Admin	3433.377	
7653	06.2017	9.0	Sweden	29650.0	Gothenburg	12.0	Checkout	6322.323	
7654	06.2017	9.0	Sweden	29650.0	Gothenburg	16.0	Customer Services	4270.479	
7655	06.2017	9.0	Sweden	29650.0	Gothenburg	11.0	Delivery	0	
7656	06.2017	9.0	Sweden	29650.0	Gothenburg	17.0	others	2224.929	
7657	06.2017	9.0	Sweden	29650.0	Gothenburg	18.0	all	39652.2	

To describe statistics Analysis

In [5]:

```
d.describe()
```

Out[5]:

	Time index	StoreID	Dept_ID	HoursLease	Sales units	Turnover	Cu
count	7650.000000	7650.000000	7650.000000	7650.000000	7.650000e+03	7.650000e+03	
mean	5.000000	61995.220000	9.470588	22.036078	1.076471e+06	3.721393e+06	
std	2.582158	29924.581631	5.337429	133.299513	1.728113e+06	6.003380e+06	
min	1.000000	12227.000000	1.000000	0.000000	0.000000e+00	0.000000e+00	
25%	3.000000	29650.000000	5.000000	0.000000	5.457125e+04	2.726798e+05	
50%	5.000000	75400.500000	9.000000	0.000000	2.932300e+05	9.319575e+05	
75%	7.000000	87703.000000	14.000000	0.000000	9.175075e+05	3.264432e+06	
max	9.000000	98422.000000	18.000000	3984.000000	1.124296e+07	4.271739e+07	

To get rows and columns

In [6]:

```
np.shape(d)
```

Out[6]:

```
(7658, 14)
```

To get number of elements

In [7]:

```
np.size(d)
```

Out[7]:

```
107212
```

To get the missing value

In [8]:

```
d.isna()
```

Out[8]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	Su
0	False	False	False	False	False	False	False	False	False	F
1	False	False	False	False	False	False	False	False	False	F
2	False	False	False	False	False	False	False	False	False	F
3	False	False	False	False	False	False	False	False	False	F
4	False	False	False	False	False	False	False	False	False	F
...	...	...	...	...	...	...	...	...	...	
7653	False	False	False	False	False	False	False	False	False	F
7654	False	False	False	False	False	False	False	False	False	F
7655	False	False	False	False	False	False	False	False	False	F
7656	False	False	False	False	False	False	False	False	False	F
7657	False	False	False	False	False	False	False	False	False	F

7658 rows × 14 columns

To drop the missing elements

In [9]:

```
d.dropna(axis=1,how='any')
```

Out[9]:

	MonthYear
0	10.2016
1	10.2016
2	10.2016
3	10.2016
4	10.2016
...	...
7653	06.2017
7654	06.2017
7655	06.2017
7656	06.2017
7657	06.2017

7658 rows × 1 columns

In [10]:

```
d["Country"]
```

Out[10]:

```
0       United Kingdom
1       United Kingdom
2       United Kingdom
3       United Kingdom
4       United Kingdom
             ...
7653           Sweden
7654           Sweden
7655           Sweden
7656           Sweden
7657           Sweden
Name: Country, Length: 7658, dtype: object
```

In [11]:

```
data=d[['Time index','Dept_ID']]
data
```

Out[11]:

	Time index	Dept_ID
0	1.0	1.0
1	1.0	2.0
2	1.0	3.0
3	1.0	4.0
4	1.0	5.0
...	...	...
7653	9.0	12.0
7654	9.0	16.0
7655	9.0	11.0
7656	9.0	17.0
7657	9.0	18.0

7658 rows × 2 columns

In [12]:

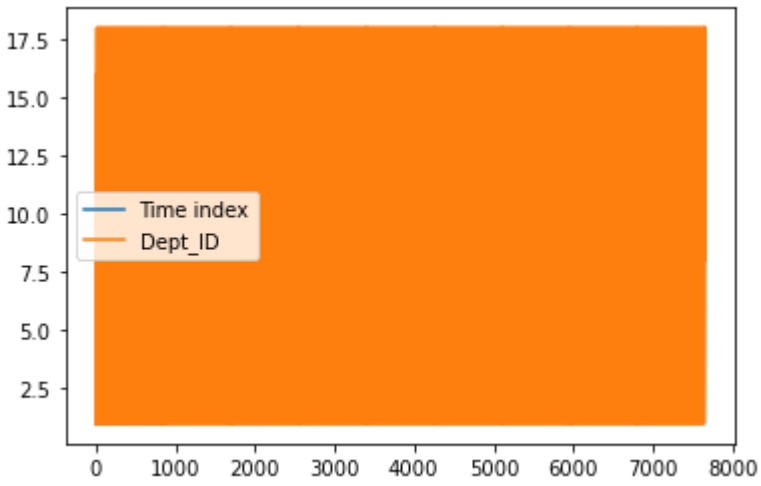
```
import matplotlib.pyplot as pp
```

In [13]:

```
data.plot.line()
```

Out[13]:

```
<AxesSubplot:>
```

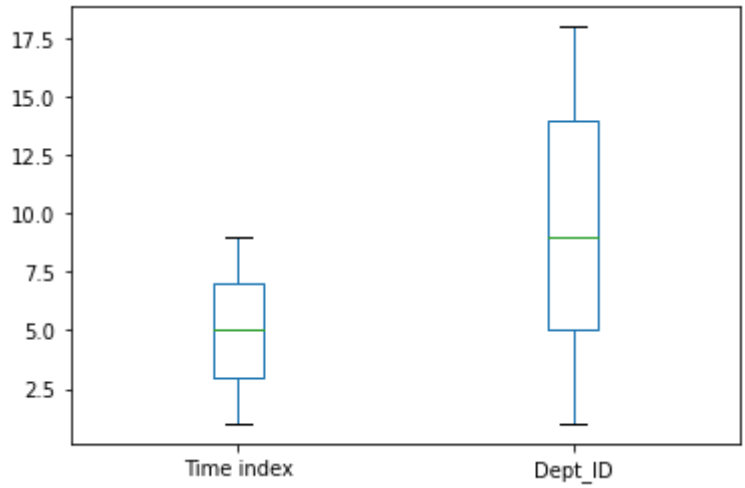


In [14]:

```
data.plot.box()
```

Out[14]:

```
<AxesSubplot:>
```

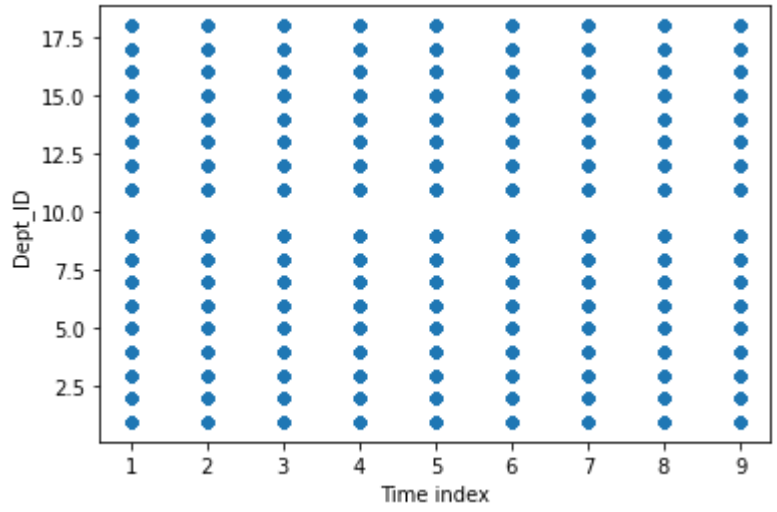


In [15]:

```
data.plot.scatter(x="Time index",y="Dept_ID")
```

Out[15]:

```
<AxesSubplot:xlabel='Time index', ylabel='Dept_ID'>
```

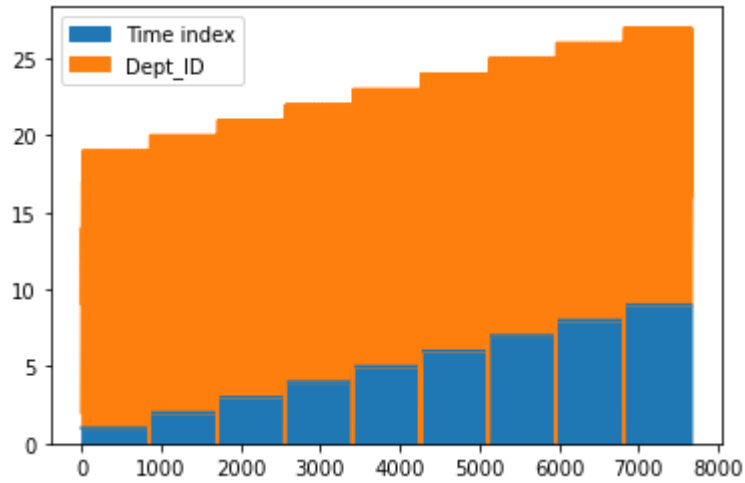


In [16]:

```
data.plot.area()
```

Out[16]:

```
<AxesSubplot:>
```

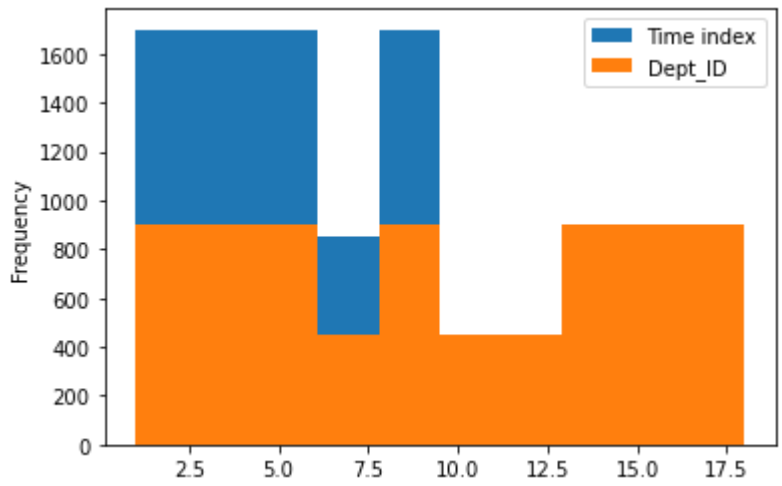


In [17]:

```
data.plot.hist()
```

Out[17]:

```
<AxesSubplot:ylabel='Frequency'>
```



In [19]:

```
d.plot.pie(y="Time index")
```

Out[19]:

```
<AxesSubplot:ylabel='Time index'>
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\formatters.py in _
_call__(self, obj)
    339                 pass
    340             else:
--> 341                 return printer(obj)
    342             # Finally look for special method names
    343             method = get_real_method(obj, self.print_method)

C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\pylabtools.py in <
lambda>(fig)
    246
    247     if 'png' in formats:
--> 248         png_formatter.for_type(Figure, lambda fig: print_figure(fi
g, 'png', **kwargs))
    249     if 'retina' in formats or 'png2x' in formats:
    250         png_formatter.for_type(Figure, lambda fig: retina_figure(f
ig, **kwargs))

C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\pylabtools.py in p
rint_figure(fig, fmt, bbox_inches, **kwargs)
    130         FigureCanvasBase(fig)
    131
--> 132     fig.canvas.print_figure(bytes_io, **kw)
    133     data = bytes_io.getvalue()
    134     if fmt == 'svg':

C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\backend_bases.py in
print_figure(self, filename, dpi, facecolor, edgecolor, orientation, forma
t, bbox_inches, pad_inches, bbox_extra_artists, backend, **kwargs)
   2208
In [9]:             try:
-> 2210                 result = print_method(
   2211                     filename,
   2212                     dpi=dpi,

C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\backend_bases.py in
wrapper(*args, **kwargs)
   1637             kwargs.pop(arg)
   1638
-> 1639         return func(*args, **kwargs)
   1640
```