In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv(r"C:\Users\user\Downloads\6_Salesworkload1 (1).csv")
df
```

Out[2]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLea
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

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7658 entries, 0 to 7657
Data columns (total 14 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   MonthYear      7658 non-null   object 
 1   Time index     7650 non-null   float64
 2   Country        7650 non-null   object 
 3   StoreID        7650 non-null   float64
 4   City           7650 non-null   object 
 5   Dept_ID        7650 non-null   float64
 6   Dept. Name     7650 non-null   object 
 7   HoursOwn       7650 non-null   object 
 8   HoursLease     7650 non-null   float64
 9   Sales units    7650 non-null   float64
 10  Turnover       7650 non-null   float64
 11  Customer       0 non-null      float64
 12  Area (m2)      7650 non-null   object 
 13  Opening hours  7650 non-null   object 
dtypes: float64(7), object(7)
memory usage: 837.7+ KB
```

In [4]:
```
df1 = df[0:500]
df1
```

Out[4]:

	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
...	...	...	...	...	...	...	...	...	...
495	10.2016	1.0	Italy	64983.0	Milano	3.0	other	47.205	0.0
496	10.2016	1.0	Italy	64983.0	Milano	4.0	Fish	2451.513	0.0
497	10.2016	1.0	Italy	64983.0	Milano	5.0	Fruits & Vegetables	1944.846	0.0
498	10.2016	1.0	Italy	64983.0	Milano	6.0	Meat	11980.629	122.0
499	10.2016	1.0	Italy	64983.0	Milano	13.0	Food	23665.44	122.0

500 rows × 14 columns

In [5]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 14 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   MonthYear      500 non-null    object 
 1   Time index     500 non-null    float64
 2   Country        500 non-null    object 
 3   StoreID        500 non-null    float64
 4   City           500 non-null    object 
 5   Dept_ID        500 non-null    float64
 6   Dept. Name     500 non-null    object 
 7   HoursOwn       500 non-null    object 
 8   HoursLease     500 non-null    float64
 9   Sales units    500 non-null    float64
 10  Turnover       500 non-null    float64
 11  Customer       0 non-null      float64
 12  Area (m2)      500 non-null    object 
 13  Opening hours  500 non-null    object 
dtypes: float64(7), object(7)
memory usage: 54.8+ KB
```

In [6]:
```
df1.describe()
```

Out[6]:

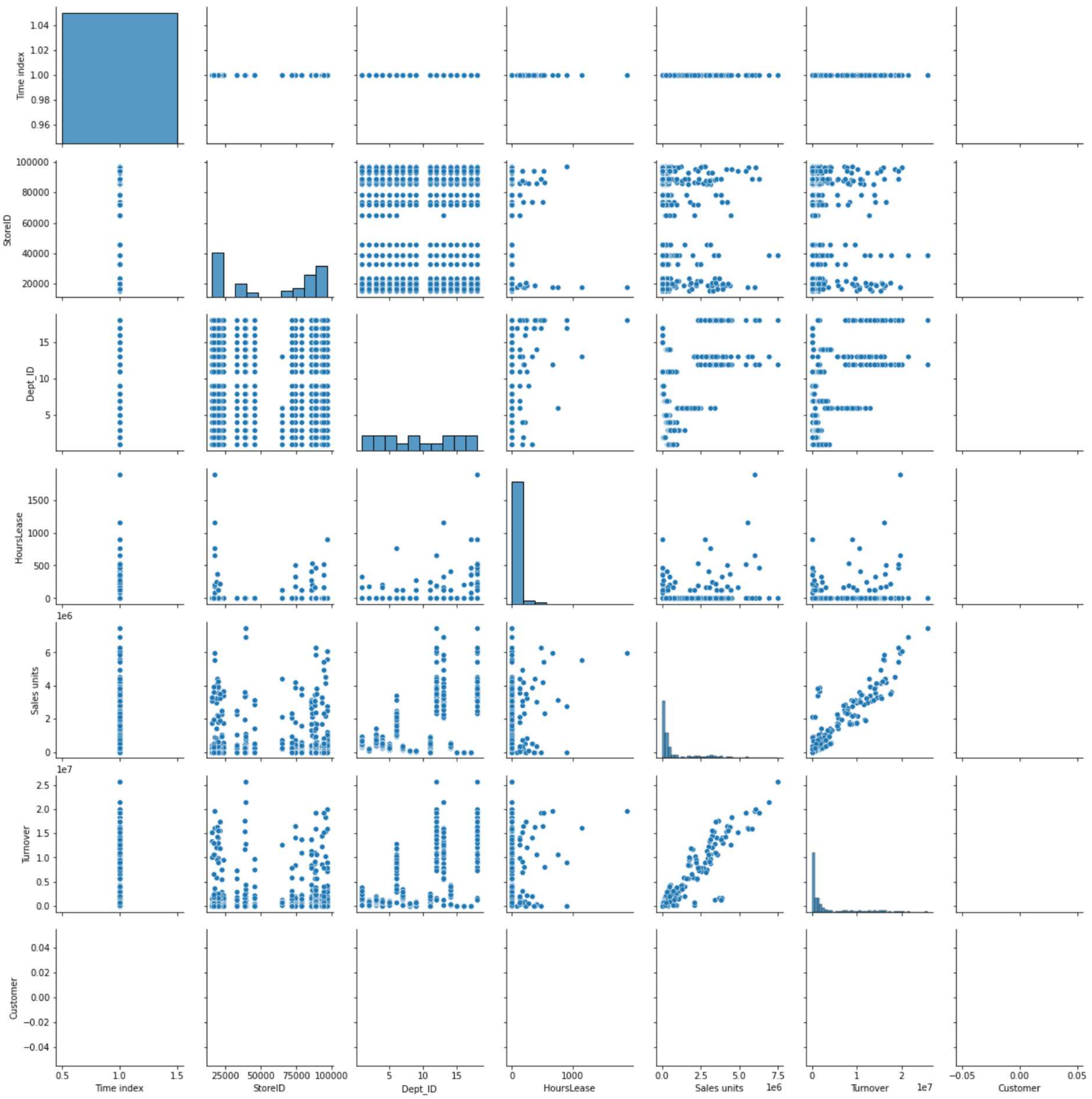
	Time index	StoreID	Dept_ID	HoursLease	Sales units	Turnover	Customer
count	500.0	500.000000	500.000000	500.000000	5.000000e+02	5.000000e+02	0.0
mean	1.0	57412.764000	9.406000	31.520000	9.397837e+05	3.153113e+06	NaN
std	0.0	32104.273482	5.350366	142.134408	1.486945e+06	5.165524e+06	NaN
min	1.0	15552.000000	1.000000	0.000000	0.000000e+00	0.000000e+00	NaN
25%	1.0	20891.000000	5.000000	0.000000	5.200250e+04	2.345122e+05	NaN
50%	1.0	71991.000000	9.000000	0.000000	2.555375e+05	7.053345e+05	NaN
75%	1.0	88253.000000	14.000000	0.000000	8.903900e+05	2.542147e+06	NaN
max	1.0	96857.000000	18.000000	1896.000000	7.476680e+06	2.571973e+07	NaN

In [7]:
```
df1.columns
```

Out[7]:
```
Index(['MonthYear', 'Time index', 'Country', 'StoreID', 'City', 'Dept_ID',
       'Dept. Name', 'HoursOwn', 'HoursLease', 'Sales units', 'Turnover',
       'Customer', 'Area (m2)', 'Opening hours'],
      dtype='object')
```

```
In [8]: sns.pairplot(df1)
```

```
Out[8]: <seaborn.axisgrid.PairGrid at 0x22d2f6d62b0>
```



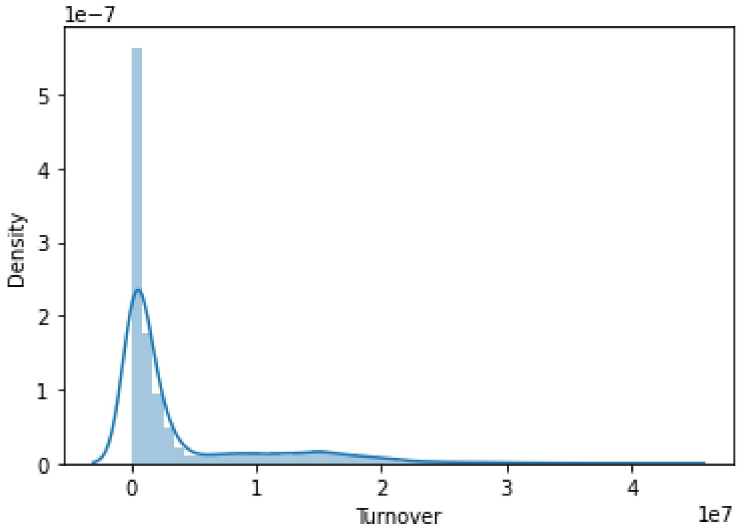
In [9]:

```
sns.distplot(df['Turnover'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level fu
nction with similar flexibility) or `histplot` (an axes-level function for hi
stograms).
  warnings.warn(msg, FutureWarning)
```

Out[9]:

```
<AxesSubplot:xlabel='Turnover', ylabel='Density'>
```

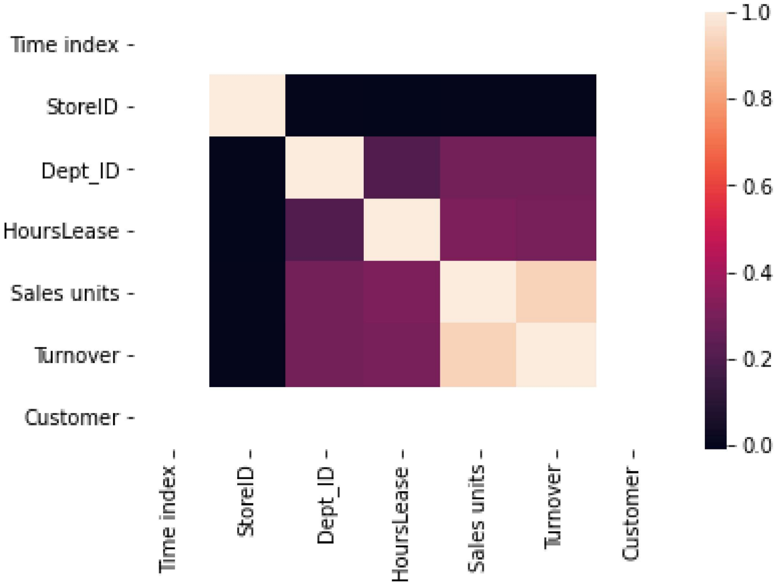


In [10]:

```
df2 = df1[['HoursOwn', 'HoursLease', 'Sales units', 'Turnover']]
sns.heatmap(df1.corr())
```

Out[10]:

```
<AxesSubplot:>
```



In [11]:

```
x = df2[['HoursOwn', 'HoursLease', 'Sales units']]
y = df2['Turnover']
```

In [12]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [13]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

Out[13]:
```
LinearRegression()
```

In [14]:
```
print(lr.intercept_)
```

```
134452.1869701813
```

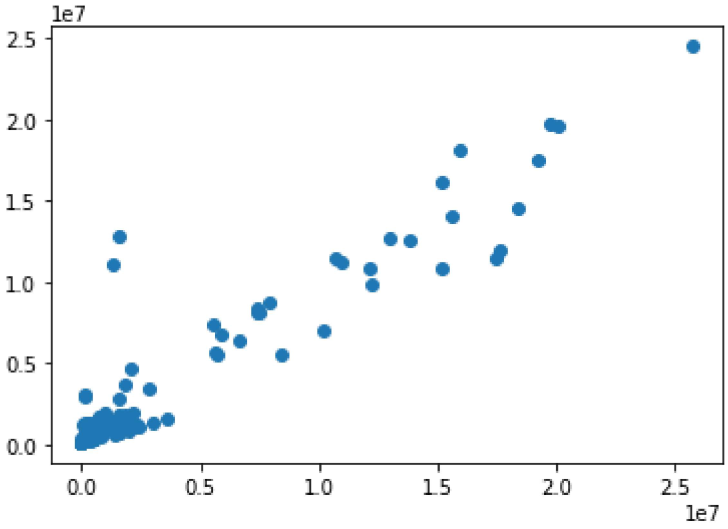
In [15]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[15]:

	Co-efficient
HoursOwn	6.463504
HoursLease	44.636122
Sales units	3.197188

In [16]:
```
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[16]:
```
<matplotlib.collections.PathCollection at 0x22d382c0a60>
```



In [17]:
```
print(lr.score(x_test,y_test))
```

```
0.8932584486612198
```

In [19]:
```
lr.score(x_train,y_train)
```

Out[19]:
```
0.8668062477556112
```

```
In [20]: from sklearn.linear_model import Ridge,Lasso
```

```
In [21]: rr = Ridge(alpha=10)
         rr.fit(x_train,y_train)
         rr.score(x_test,y_test)
```

```
Out[21]: 0.8932584487019442
```

```
In [22]: rr.score(x_train,y_train)
```

```
Out[22]: 0.8668062477556112
```

```
In [23]: ls = Lasso(alpha=10)
         ls.fit(x_train,y_train)
         ls.score(x_train,y_train)
```

```
Out[23]: 0.866806247755611
```

```
In [24]: ls.score(x_test,y_test)
```

```
Out[24]: 0.893258448982297
```

```
In [ ]: 
```