The data used here is from year 1995 to 2019 of each different area. this data is available as a csv file , download from kaggle. we will analyze this data using the Pandas Dataframe Here, sets of questions are given for which we have to find correct results. this project is for begineers and for those who want to know how we analyzing big data with python .

In [1]:

```
#Import Pandas Library
import pandas as pd
```

In [2]:

```
#read csv file
london = pd.read_csv(r"C:\Users\USER\Downloads\5. London Housing Data.csv")
```

In [3]:

```
london
```

Out[3]:

	date	area	average_price	code	houses_sold	no_of_crimes
0	1/1/1995	city of london	91449	E09000001	17.0	NaN
1	2/1/1995	city of london	82203	E09000001	7.0	NaN
2	3/1/1995	city of london	79121	E09000001	14.0	NaN
3	4/1/1995	city of london	77101	E09000001	7.0	NaN
4	5/1/1995	city of london	84409	E09000001	10.0	NaN
...	...	...	...	...	...	...
13544	9/1/2019	england	249942	E92000001	64605.0	NaN
13545	10/1/2019	england	249376	E92000001	68677.0	NaN
13546	11/1/2019	england	248515	E92000001	67814.0	NaN
13547	12/1/2019	england	250410	E92000001	NaN	NaN
13548	1/1/2020	england	247355	E92000001	NaN	NaN

13549 rows × 6 columns

In [4]:

```
london.count()
#count() function is used to count non-NA cells for each column or rowcount() function is used to count non-NA cells for each column or row
```

Out[4]:

```
date              13549
area              13549
average_price     13549
code              13549
houses_sold       13455
no_of_crimes       7439
dtype: int64
```

In [5]:

```
#returns the number of missing values in the data set.
london.isnull().sum()
```

Out[5]:

```
date                0
area                0
```

```
average_price       0
code                0
houses_sold        94
no_of_crimes     6110
dtype: int64
```

In This Dataset , House_Sold and no_of_cries has null values present in it.

In [6]:

```
import seaborn as sns
```

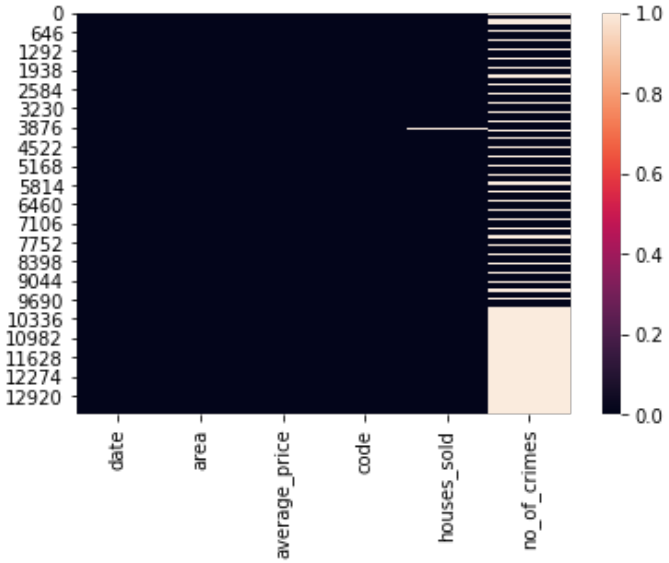
In [7]:

```
import matplotlib.pyplot as plt
```

Heatmaps visualise data through variations in colouring. heatmap is a data visualization technique that shows magnitude of a phenomenon as color in two dimensions.

In [8]:

```
sns.heatmap(london.isnull())
plt.show()
```



pyplot.show() function is used to represent the heatmap with proper formatting.

A) Convert the datatype of "date " column to date- time format .

In [9]:

```
london.dtypes
```

Out[9]:

```
date              object
area              object
average_price      int64
code              object
houses_sold      float64
no_of_crimes     float64
dtype: object
```

In [10]:

```
london.date =  pd.to_datetime(london.date) #change the dtype
#london['date']
```

In [11]:

```
london.dtypes
```

Out[11]:

```
date              datetime64[ns]
area                      object
average_price              int64
code                      object
houses_sold              float64
no_of_crimes             float64
dtype: object
```

B.1) Add a new column "Year" in the dataframe , which contains years only.

In [12]:

```
london['year'] = london.date.dt.year
```

In [13]:

```
london
```

Out[13]:

	date	area	average_price	code	houses_sold	no_of_crimes	year
0	1995-01-01	city of london	91449	E09000001	17.0	NaN	1995
1	1995-02-01	city of london	82203	E09000001	7.0	NaN	1995
2	1995-03-01	city of london	79121	E09000001	14.0	NaN	1995
3	1995-04-01	city of london	77101	E09000001	7.0	NaN	1995
4	1995-05-01	city of london	84409	E09000001	10.0	NaN	1995
...	...	...	...	...	...	...	...
13544	2019-09-01	england	249942	E92000001	64605.0	NaN	2019
13545	2019-10-01	england	249376	E92000001	68677.0	NaN	2019
13546	2019-11-01	england	248515	E92000001	67814.0	NaN	2019
13547	2019-12-01	england	250410	E92000001	NaN	NaN	2019
13548	2020-01-01	england	247355	E92000001	NaN	NaN	2020

13549 rows × 7 columns

B.2) Add a new column "Month" as 2nd column in the dataframe , which contains months only.

In [14]:

```
#london['month'] =london.date.dt.month
#used insert
london.insert(1,'month1', london.date.dt.month)
```

In [15]:

```
london
```

Out[15]:

	date	month1	area	average_price	code	houses_sold	no_of_crimes	year

	date	month1	area	average_price	code	houses_sold	no_of_crimes	year
0	1995-01-01	1	city of london	91449	E09000001	17.0	NaN	1995
1	1995-02-01	2	city of london	82203	E09000001	7.0	NaN	1995
2	1995-03-01	3	city of london	79121	E09000001	14.0	NaN	1995
3	1995-04-01	4	city of london	77101	E09000001	7.0	NaN	1995
4	1995-05-01	5	city of london	84409	E09000001	10.0	NaN	1995
...	...	...	...	...	...	...	...	...
13544	2019-09-01	9	england	249942	E92000001	64605.0	NaN	2019
13545	2019-10-01	10	england	249376	E92000001	68677.0	NaN	2019
13546	2019-11-01	11	england	248515	E92000001	67814.0	NaN	2019
13547	2019-12-01	12	england	250410	E92000001	NaN	NaN	2019
13548	2020-01-01	1	england	247355	E92000001	NaN	NaN	2020

13549 rows × 8 columns

c) Remove the columns 'year' and 'month''months1' from the dataframe.

In [16]:

```
london.drop(['year','month','month1'],axis=1,inplace = True)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-16-3859e6e930f3> in <module>
----> 1 london.drop(['year','month','month1'],axis=1,inplace = True)

~\AppData\Roaming\Python\Python39\site-packages\pandas\core\frame.py in drop(self, labels
, axis, index, columns, level, inplace, errors)
   4306                 weight  1.0     0.8
   4307         """
-> 4308         return super().drop(
   4309             labels=labels,
   4310             axis=axis,

~\AppData\Roaming\Python\Python39\site-packages\pandas\core\generic.py in drop(self, labe
ls, axis, index, columns, level, inplace, errors)
   4151         for axis, labels in axes.items():
   4152             if labels is not None:
-> 4153                 obj = obj._drop_axis(labels, axis, level=level, errors=errors)
   4154
   4155         if inplace:

~\AppData\Roaming\Python\Python39\site-packages\pandas\core\generic.py in _drop_axis(self
, labels, axis, level, errors)
   4186                 new_axis = axis.drop(labels, level=level, errors=errors)
   4187             else:
-> 4188                 new_axis = axis.drop(labels, errors=errors)
   4189             result = self.reindex(**{axis_name: new_axis})
   4190

~\AppData\Roaming\Python\Python39\site-packages\pandas\core\indexes\base.py in drop(self,
labels, errors)
   5589         if mask.any():
   5590             if errors != "ignore":
-> 5591                 raise KeyError(f"{labels[mask]} not found in axis")
   5592             indexer = indexer[~mask]
   5593         return self.delete(indexer)

KeyError: "['month'] not found in axis"
```

In [ ]:

```
london.head(2)
```

D) Show all the records where "no. of crime " is 0. And , how many such records are there?

In [ ]:

```
london.no_of_crimes == 0
```

In [ ]:

```
len(london[london.no_of_crimes == 0])
```

In [ ]:

```
london[london.no_of_crimes == 0]
```

E) What is the maximum & minimum 'Average_price' per year in england?

In [ ]:

```
london['year'] = london.date.dt.year
london.groupby('year').average_price.max()
```

F) What is the maximum & minimum no. of crime records per area?

In [ ]:

```
london.groupby('area').no_of_crimes.max().sort_values(ascending = False)
```

In [ ]:

```
london.groupby('area').no_of_crimes.min().sort_values(ascending = True)
```

G) Show the total count of records of each are, where average price is less than 100000.

In [ ]:

```
london[london.average_price < 100000] .area.value_counts()
```

In [ ]:

```

```

In [ ]:

```

```