Importing pandas and plotly express

In [1]:
```
import pandas as pd
import plotly.express as px
```

In [2]:
```
df = pd.read_csv('Airbnb_Open_Data.csv')
```

```
C:\Users\anaconda3\Lib\site-packages\IPython\core\interactiveshell.py:3146: DtypeWarning: Columns (25) have mixed types.Specify dtype option o
n import or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

In [3]:
```
df.columns
```

Out[3]:
```
Index(['id', 'NAME', 'host id', 'host_identity_verified', 'host name',
       'neighbourhood group', 'neighbourhood', 'lat', 'long', 'country',
       'country code', 'instant_bookable', 'cancellation_policy', 'room type',
       'Construction year', 'price', 'service fee', 'minimum nights',
       'number of reviews', 'last review', 'reviews per month',
       'review rate number', 'calculated host listings count',
       'availability 365', 'house_rules', 'license'],
      dtype='object')
```

Dropping columns not needed/ not significant

In [4]:
```
df.drop(columns=['host id','host name','lat','long', 'country code','house_rules','license'], inplace = True)
```

In [5]:
```
df.head(5)
```

Out[5]:

	id	NAME	host_identity_verified	neighbourhood group	neighbourhood	country	instant_bookable	cancellation_policy	room type	Construction year	price	service fee	minimum nights	numb revie
0	1001254	Clean & quiet apt home by the park	unconfirmed	Brooklyn	Kensington	United States	False	strict	Private room	2020.0	$966	$193	10.0	
1	1002102	Skylit Midtown Castle	verified	Manhattan	Midtown	United States	False	moderate	Entire homeapt	2007.0	$142	$28	30.0	4
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	NaN	Manhattan	Harlem	United States	True	flexible	Private room	2005.0	$620	$124	3.0	
3	1002755	NaN	unconfirmed	Brooklyn	Clinton Hill	United States	True	moderate	Entire homeapt	2005.0	$368	$74	30.0	27
4	1003689	Entire Apt: Spacious Studio/Loft by central park	verified	Manhattan	East Harlem	United States	False	moderate	Entire homeapt	2009.0	$204	$41	10.0	

Finding null values in each column

In [6]:
```
for i in df.columns:
    print(df[i].isnull().mean())
```

```
0.0
0.002436679922718545
0.002816795586062638
0.0003662118978966483
0.000235048395398687
0.005185257235645664
0.001023401787541789
0.008740747960504377
0.0
0.002085793369847543
0.002447430871649224
0.000366211897896648
0.003988393962954754
0.001783641115429075
0.154599483899653
0.104767008327911
0.000366211897896648
0.001289192987388832
0.004365514293511633
```

Filling null values with 'unknown' for string based columns

In [7]:
```
df.iloc[:,1:8] = df.iloc[:,1:8].fillna('unknown')
```

Removing all rows where 'last review' is null

In [8]:
```
df = df[df['last review'].notna()]
```

In [9]:
```
df.head(3)
```

Out[9]:

	id	NAME	host_identity_verified	neighbourhood group	neighbourhood	country	instant_bookable	cancellation_policy	room type	Construction year	price	service fee	minimum nights	number of reviews	
0	1001254	Clean & quiet apt home by the park	unconfirmed	Brooklyn	Kensington	United States	False	strict	Private room	2020.0	$966	$193	10.0	9.0	10
1	1002102	Skylit Midtown Castle	verified	Manhattan	Midtown	United States	False	moderate	Entire homeapt	2007.0	$142	$28	30.0	45.0	5
3	1002755	Unknown	unconfirmed	Brooklyn	Clinton Hill	United States	True	moderate	Entire homeapt	2005.0	$368	$74	30.0	270.0	

For the numeric columns, each null value in each column is filled with the mean of the column. For the column of 'price' and'service fee', first the '$','.' characthers are removed after which they are converted into float data types so to be used in the mean calculation

In [10]:
```
df.iloc[:,9:14]
```

Out[10]:

	Construction year	price	service fee	minimum nights	number of reviews
0	2020.0	$966	$193	10.0	9.0
1	2007.0	$142	$28	30.0	45.0
3	2005.0	$368	$74	30.0	270.0
4	2009.0	$204	$41	10.0	9.0
5	2013.0	$577	$115	3.0	74.0
...	...	...	...	...	...
102588	2016.0	$618	$124	1.0	177.0
102591	2005.0	$1,139	$228	5.0	17.0
102593	2017.0	$1,099	$220	1.0	6.0
102595	2016.0	$837	$167	1.0	1.0
102597	2015.0	$546	$109	2.0	5.0

86706 rows × 5 columns

In [11]:
```
df['price'] = df['price'].str.replace('$', '')
```

In [12]:
```
df['price'] = df['price'].str.replace(',', '')
```

In [13]:
```
df['service fee'] = df['service fee'].str.replace('$', '')
```

In [14]:
```
df['service fee'] = df['service fee'].str.replace(',', '')
```

In [15]:
```
df['price'] = df['price'].astype('float')
```

In [16]:
```
df['service fee'] = df['service fee'].astype('float')
```

In [17]:
```
df.iloc[:,9:14] = df.iloc[:,9:14].fillna(df.iloc[:,9:14].mean())
```

For the reviews and availability columns, similar method is used to fill the null values with the mean of the column.

In [18]:
```
df.iloc[:,15:] = df.iloc[:,15:].fillna(df.iloc[:,15:].mean())
```

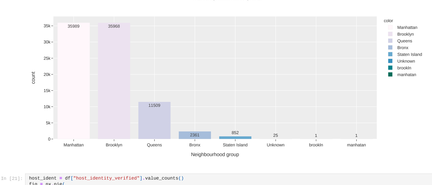
Each column is again checked for null values

In [19]:
```
for i in df.columns:
    print(df[i].isnull().mean())
```

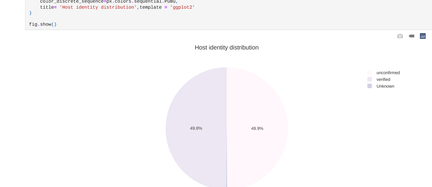
```
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
```

The following are some visulisations based on the data

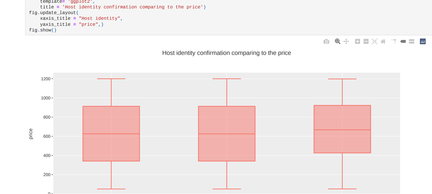
In [20]:
```
nb = df["neighbourhood group"].value_counts()
fig = px.bar(y = nb.values,
             x = nb.index,
             color = nb.index,
             color_discrete_sequence=px.colors.sequential.PuBuGn,
             text = nb.values,
             title = "Top Neighbourhood group",
             template = 'ggplot2')
fig.update_layout(
    xaxis_title = "Neighbourhood group",
    yaxis_title = "count")
fig.show()
```



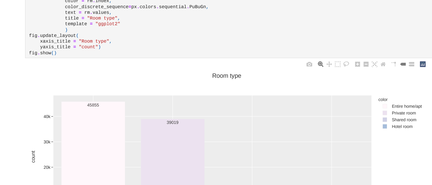
In [21]:
```
host_ident = df["host_identity_verified"].value_counts()
fig = px.pie(
             values = host_ident.values,
             names = host_ident.index,
             color_discrete_sequence=px.colors.sequential.PuBu,
             title= 'Host identity distribution',template = 'ggplot2'
)

fig.show()
```



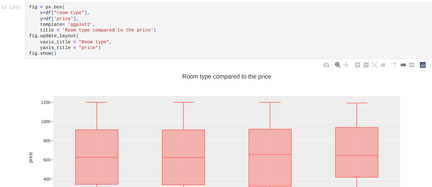
In [22]:
```
fig = px.box(
    x=df["host_identity_verified"],
    y=df['price'],
    template= 'ggplot2',
    title = 'Host identity confirmation comparing to the price')
fig.update_layout(
    xaxis_title = 'Host identity",
    yaxis_title = "price",)
fig.show()
```



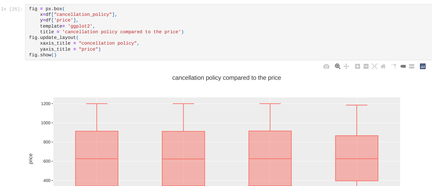
In [23]:
```
rm = df["room type"].value_counts()
fig = px.bar(y = rm.values,
             x = rm.index,
             color = rm.index,
             color_discrete_sequence=px.colors.sequential.PuBuGn,
             text = rm.values,
             title = "Room type",
             template = 'ggplot2'
             )
fig.update_layout(
    xaxis_title = "Room type",
    yaxis_title = "count")
fig.show()
```



In [24]:
```
fig = px.box(
    x=df["room type"],
    y=df['price'],
    template= 'ggplot2',
    title = 'Room type compared to the price')
fig.update_layout(
    xaxis_title = "Room type",
    yaxis_title = "price")
fig.show()
```



In [25]:
```
fig = px.box(
    x=df["cancellation_policy"],
    y=df['price'],
    template= 'ggplot2',
    title = 'cancellation policy compared to the price')
fig.update_layout(
    xaxis_title = "cancellation policy",
    yaxis_title = "price")
fig.show()
```



In [1]: