STUDYING THE RAW DATA

In [1]:

```
# importing required libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

```
#importing the data
dataset=pd.read_csv('CAR DETAILS FROM CAR DEKHO.csv')
```

In [3]:

```
#Fetching  first 50 records for training
dataset.head(50)
```

Out[3]:

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner
1	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner
2	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner
3	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner
4	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner
5	Maruti Alto LX BSIII	2007	140000	125000	Petrol	Individual	Manual	First Owner
6	Hyundai Xcent 1.2 Kappa S	2016	550000	25000	Petrol	Individual	Manual	First Owner
7	Tata Indigo Grand Petrol	2014	240000	60000	Petrol	Individual	Manual	Second Owner
8	Hyundai Creta 1.6 VTVT S	2015	850000	25000	Petrol	Individual	Manual	First Owner
9	Maruti Celerio Green VXI	2017	365000	78000	CNG	Individual	Manual	First Owner
10	Chevrolet Sail 1.2 Base	2015	260000	35000	Petrol	Individual	Manual	First Owner
11	Tata Indigo Grand Petrol	2014	250000	100000	Petrol	Individual	Manual	First Owner
12	Toyota Corolla Altis 1.8 VL CVT	2018	1650000	25000	Petrol	Dealer	Automatic	First Owner

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
13	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner
14	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner
15	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner
16	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner
17	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner
18	Maruti Alto LX BSIII	2007	140000	125000	Petrol	Individual	Manual	First Owner
19	Hyundai Xcent 1.2 Kappa S	2016	550000	25000	Petrol	Individual	Manual	First Owner
20	Tata Indigo Grand Petrol	2014	240000	60000	Petrol	Individual	Manual	Second Owner
21	Hyundai Creta 1.6 VTVT S	2015	850000	25000	Petrol	Individual	Manual	First Owner
22	Maruti Celerio Green VXI	2017	365000	78000	CNG	Individual	Manual	First Owner
23	Chevrolet Sail 1.2 Base	2015	260000	35000	Petrol	Individual	Manual	First Owner
24	Tata Indigo Grand Petrol	2014	250000	100000	Petrol	Individual	Manual	First Owner
25	Toyota Corolla Altis 1.8 VL CVT	2018	1650000	25000	Petrol	Dealer	Automatic	First Owner
26	Maruti Ciaz VXi Plus	2015	585000	24000	Petrol	Dealer	Manual	First Owner
27	Hyundai Venue SX Opt Diesel	2019	1195000	5000	Diesel	Dealer	Manual	First Owner
28	Chevrolet Enjoy TCDi LTZ 7 Seater	2013	390000	33000	Diesel	Individual	Manual	Second Owner
29	Jaguar XF 2.2 Litre Luxury	2014	1964999	28000	Diesel	Dealer	Automatic	First Owner
30	Mercedes-Benz New C-Class 220 CDI AT	2013	1425000	59000	Diesel	Dealer	Automatic	First Owner
31	Maruti Vitara Brezza ZDi Plus AMT	2018	975000	4500	Diesel	Dealer	Automatic	First Owner
32	Audi Q5 2.0 TDI	2011	1190000	175900	Diesel	Dealer	Automatic	First Owner
33	Honda City V MT	2018	930000	14500	Petrol	Dealer	Manual	First Owner
34	Tata Tigor 1.2 Revotron XT	2018	525000	15000	Petrol	Individual	Manual	First Owner
35	Audi A6 2.0 TDI Design Edition	2013	1735000	50000	Diesel	Dealer	Automatic	First Owner
36	Mercedes-Benz New C-Class C 220 CDI Avantgarde	2012	1375000	33800	Diesel	Dealer	Automatic	Second Owner
37	Skoda Superb Ambition 2.0 TDI CR AT	2011	450000	130400	Diesel	Dealer	Automatic	Second Owner

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
38	Toyota Corolla Altis G AT	2016	900000	50000	Petrol	Individual	Automatic	First Owner
39	Toyota Innova 2.5 G (Diesel) 7 Seater	2015	1300000	80000	Diesel	Individual	Manual	First Owner
40	Jeep Compass 1.4 Sport Plus BSIV	2019	1400000	10000	Petrol	Individual	Manual	First Owner
41	Mercedes-Benz E-Class E 200 CGI Elegance	2010	850000	119000	Petrol	Dealer	Automatic	First Owner
42	Hyundai i10 Magna 1.1L	2014	229999	60000	Petrol	Individual	Manual	Fourth & Above Owner
43	BMW 3 Series 320d Sport Line	2013	1550000	75800	Diesel	Dealer	Automatic	Second Owner
44	Audi Q7 35 TDI Quattro Premium	2009	1250000	78000	Diesel	Dealer	Automatic	Third Owner
45	Hyundai Elantra CRDi S	2012	625000	40000	Diesel	Individual	Manual	First Owner
46	Mahindra Scorpio 1.99 S10	2014	1050000	50000	Diesel	Individual	Manual	First Owner
47	Honda City i DTEC V	2014	560000	74000	Diesel	Individual	Manual	Second Owner
48	Maruti Wagon R VXI BS IV with ABS	2014	290000	64000	Petrol	Individual	Manual	Second Owner
49	Maruti Wagon R VXI BS IV	2012	275000	60000	Petrol	Individual	Manual	Second Owner

In [4]:

```
#size of the dataset
dataset.shape
```

Out[4]:

```
(4340, 8)
```

In [5]:

```
#information on each columns
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4340 entries, 0 to 4339
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   name           4340 non-null   object
 1   year           4340 non-null   int64 
 2   selling_price  4340 non-null   int64 
 3   km_driven      4340 non-null   int64 
 4   fuel           4340 non-null   object
 5   seller_type    4340 non-null   object
 6   transmission   4340 non-null   object
 7   owner          4340 non-null   object
dtypes: int64(3), object(5)
memory usage: 271.4+ KB
```

In [6]:

```python
#checking for null values
dataset.isnull().sum()
```

Out[6]:

```
name             0
year             0
selling_price    0
km_driven        0
fuel             0
seller_type      0
transmission     0
owner            0
dtype: int64
```

In [7]:

```python
#to show no null values
dataset.isnull()
```

Out[7]:

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
4335	False	False	False	False	False	False	False	False
4336	False	False	False	False	False	False	False	False
4337	False	False	False	False	False	False	False	False
4338	False	False	False	False	False	False	False	False
4339	False	False	False	False	False	False	False	False

4340 rows × 8 columns

In [8]:

```
#description of the data
dataset.describe()
```

Out[8]:

	year	selling_price	km_driven
count	4340.000000	4.340000e+03	4340.000000
mean	2013.090783	5.041273e+05	66215.777419
std	4.215344	5.785487e+05	46644.102194
min	1992.000000	2.000000e+04	1.000000
25%	2011.000000	2.087498e+05	35000.000000
50%	2014.000000	3.500000e+05	60000.000000
75%	2016.000000	6.000000e+05	90000.000000
max	2020.000000	8.900000e+06	806599.000000

In [9]:

```
#column tag
dataset.columns
```

Out[9]:

```
Index(['name', 'year', 'selling_price', 'km_driven', 'fuel', 'seller_type',
       'transmission', 'owner'],
      dtype='object')
```

In [10]:

```
#counting frequency
print(dataset['year'].value_counts())
print(dataset['km_driven'].value_counts())
print(dataset['fuel'].value_counts())
print(dataset['seller_type'].value_counts())
print(dataset['transmission'].value_counts())
print(dataset['owner'].value_counts())
print(dataset['selling_price'].value_counts())
```

```
2017    466
2015    421
2012    415
2013    386
2014    367
```

```
2018    366
2016    357
2011    271
2010    234
2019    195
2009    193
2008    145
2007    134
2006    110
2005     85
2020     48
2004     42
2003     23
2002     21
2001     20
1998     12
2000     12
1999     10
1997      3
1996      2
1995      1
1992      1
Name: year, dtype: int64
70000     236
80000     228
50000     222
120000    220
60000     215
         ...
19107       1
32077       1
6480        1
118400      1
112198      1
Name: km_driven, Length: 770, dtype: int64
Diesel      2153
Petrol      2123
CNG           40
LPG           23
Electric       1
Name: fuel, dtype: int64
Individual          3244
Dealer               994
Trustmark Dealer     102
Name: seller_type, dtype: int64
```

```
Manual       3892
Automatic     448
Name: transmission, dtype: int64
First Owner             2832
Second Owner            1106
Third Owner              304
Fourth & Above Owner      81
Test Drive Car            17
Name: owner, dtype: int64
300000     162
250000     125
350000     122
550000     107
600000     103
          ...
2100000      1
828999       1
1119000      1
746000       1
865000       1
Name: selling_price, Length: 445, dtype: int64
```

In [11]:

```
#importing required libraries
from sklearn import preprocessing
#converting KM Driven in the ranfe of 0 to 1
dataset=pd.read_csv('CAR DETAILS FROM CAR DEKHO.csv')
dataset["km_driven"]=dataset["km_driven"]/dataset["km_driven"].max()
dataset
```

Out[11]:

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	Maruti 800 AC	2007	60000	0.086784	Petrol	Individual	Manual	First Owner
1	Maruti Wagon R LXI Minor	2007	135000	0.061989	Petrol	Individual	Manual	First Owner
2	Hyundai Verna 1.6 SX	2012	600000	0.123977	Diesel	Individual	Manual	First Owner
3	Datsun RediGO T Option	2017	250000	0.057030	Petrol	Individual	Manual	First Owner
4	Honda Amaze VX i-DTEC	2014	450000	0.174808	Diesel	Individual	Manual	Second Owner
...	...	...	...	...	...	...	...	...
4335	Hyundai i20 Magna 1.4 CRDi (Diesel)	2014	409999	0.099182	Diesel	Individual	Manual	Second Owner
4336	Hyundai i20 Magna 1.4 CRDi	2014	409999	0.099182	Diesel	Individual	Manual	Second Owner

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
4337	Maruti 800 AC BSIII	2009	110000	0.102901	Petrol	Individual	Manual	Second Owner
4338	Hyundai Creta 1.6 CRDi SX Option	2016	865000	0.111580	Diesel	Individual	Manual	First Owner
4339	Renault KWID RXT	2016	225000	0.049591	Petrol	Individual	Manual	First Owner

4340 rows × 8 columns

In [12]:

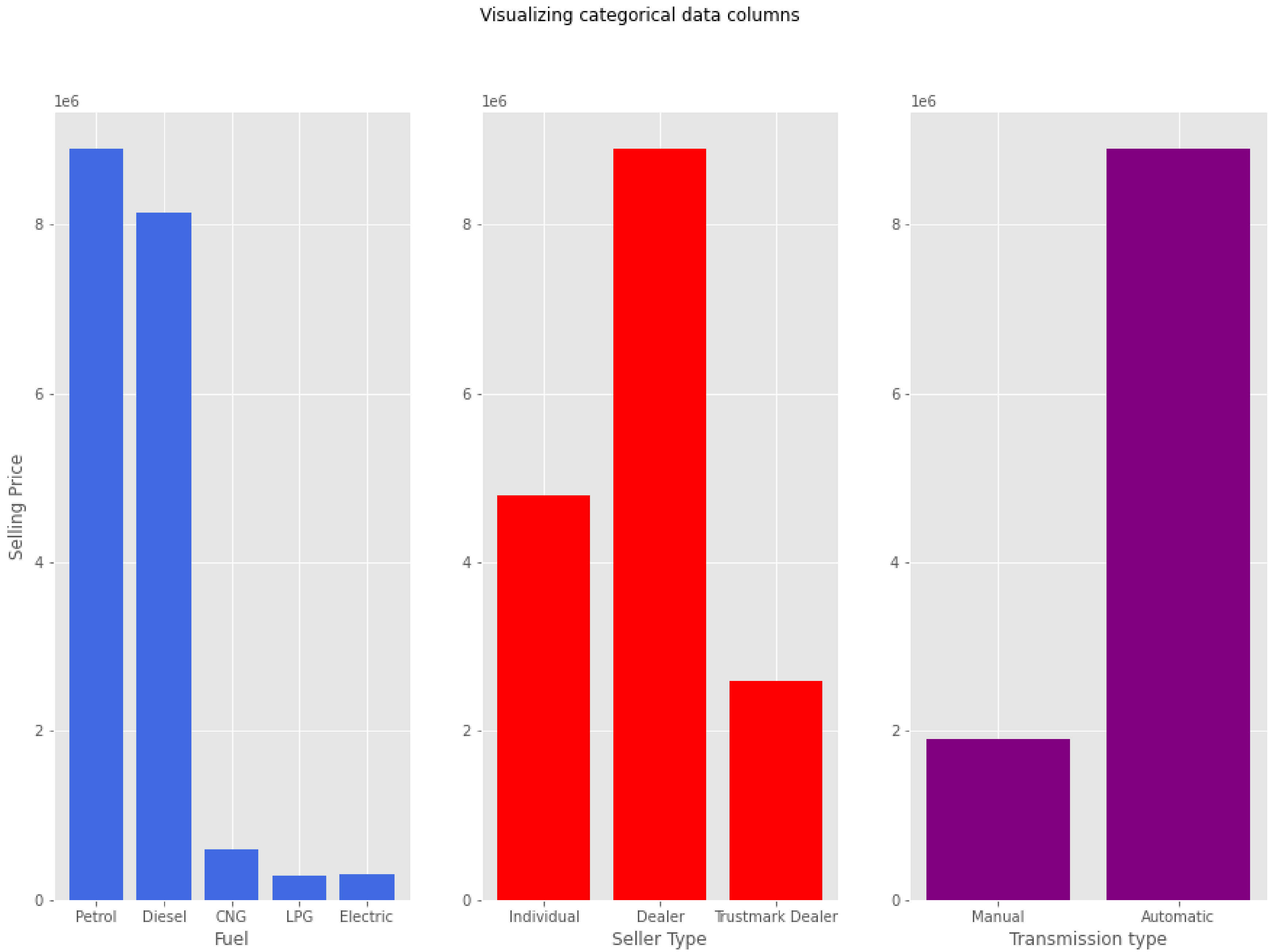
```
#Assignment of values
year=dataset['year']
km_driven=dataset['km_driven']
seller_type = dataset['seller_type']
transmission_type = dataset['transmission']
owner=dataset['owner']
fuel=dataset['fuel']
selling_price=dataset['selling_price']
```

In [13]:

```
# importing required libraries
from matplotlib import style
```

In [14]:

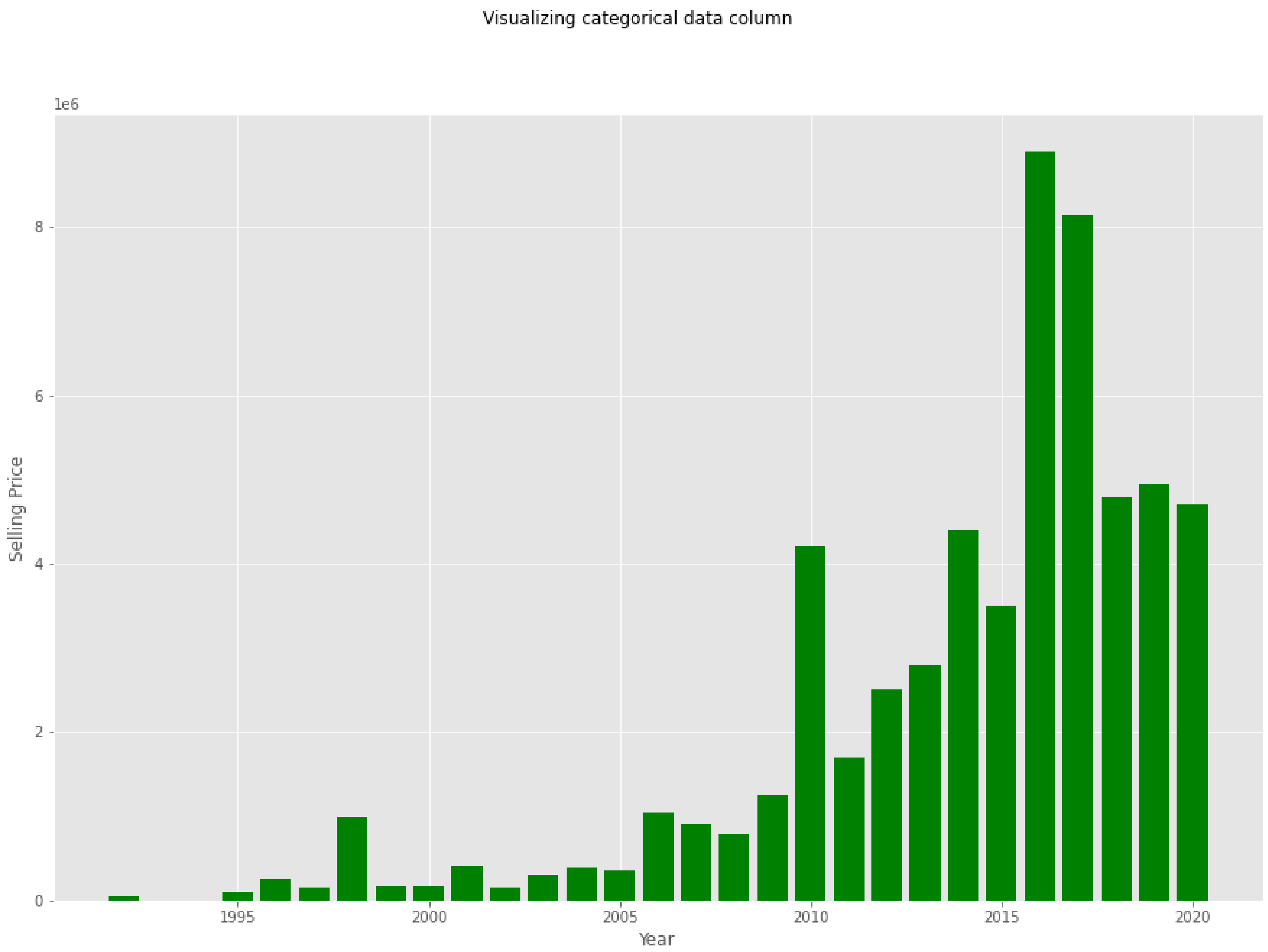
```
#Visualizing categorical data columns
style.use('ggplot')
fig = plt.figure(figsize=(15,10))
fig.suptitle('Visualizing categorical data columns')
plt.subplot(1,3,1)
plt.bar(fuel,selling_price, color='royalblue')
plt.xlabel("Fuel")
plt.ylabel("Selling Price")
plt.subplot(1,3,2)
plt.bar(seller_type, selling_price, color='red')
plt.xlabel("Seller Type")
plt.subplot(1,3,3)
plt.bar(transmission_type, selling_price, color='purple')
plt.xlabel('Transmission type')
plt.show()
```



In [15]:

```
#Visualizing categorical data column
fig = plt.figure(figsize=(15,10))
fig.suptitle('Visualizing categorical data column')
```

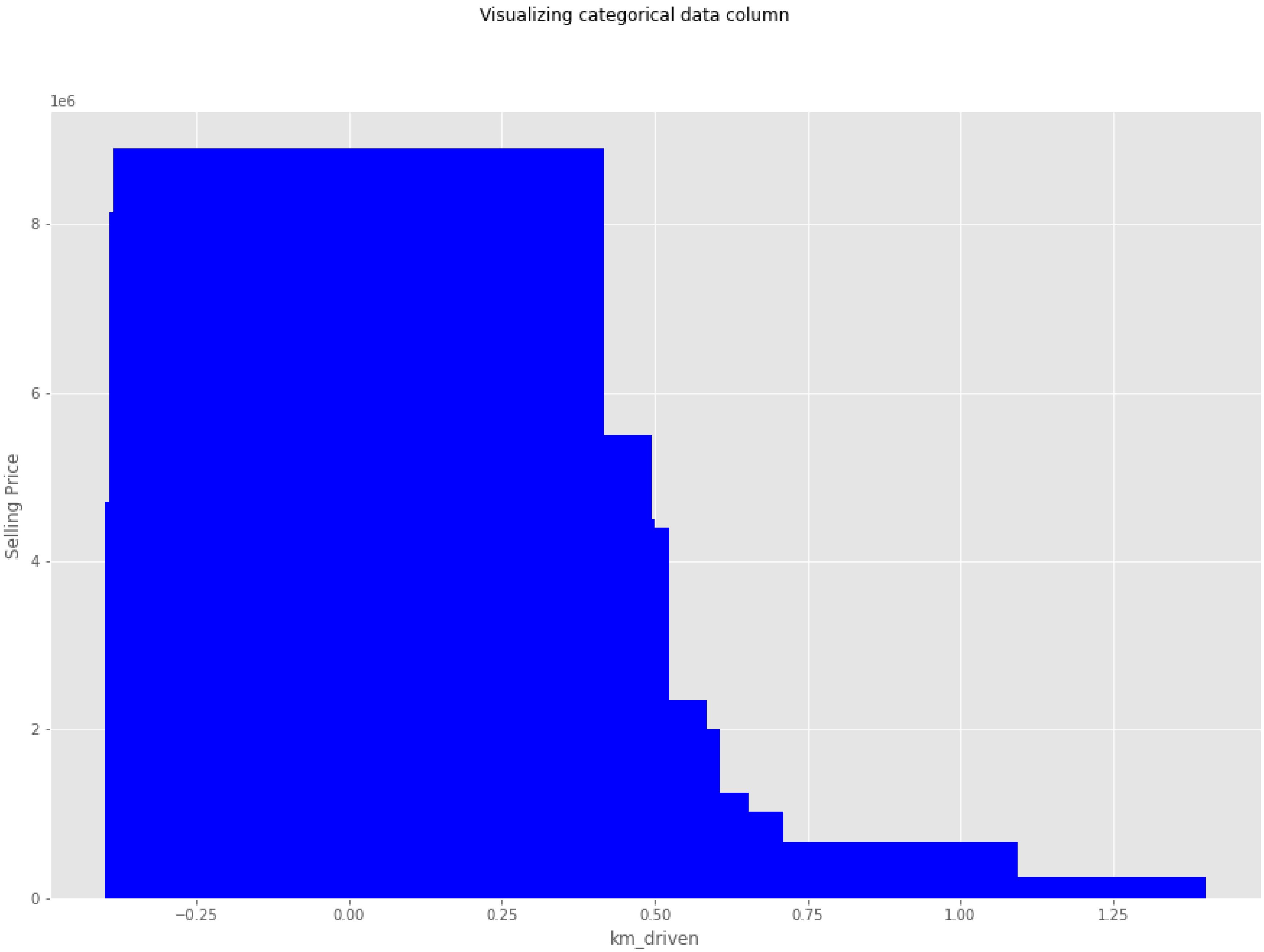
```
plt.subplot(1,1,1)
plt.bar(year,selling_price, color='green')
plt.xlabel("Year")
plt.ylabel("Selling Price")
plt.show()
```



In [16]:

```
#Visualizing categorical data column
fig = plt.figure(figsize=(15,10))
fig.suptitle('Visualizing categorical data column')
```

```
plt.subplot(1,1,1)
plt.bar(km_driven,selling_price, color='blue')
plt.xlabel("km_driven")
plt.ylabel("Selling Price")
plt.show()
```



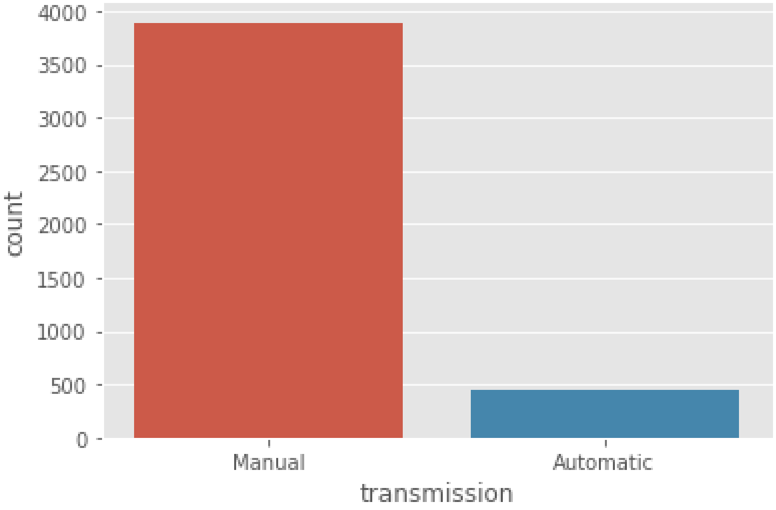
In [17]:

```
#Bar Graph
gear=dataset['transmission']
sns.countplot(gear)
```

```
D:\ANACONDA\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x.
From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

Out[17]:

```
<AxesSubplot:xlabel='transmission', ylabel='count'>
```



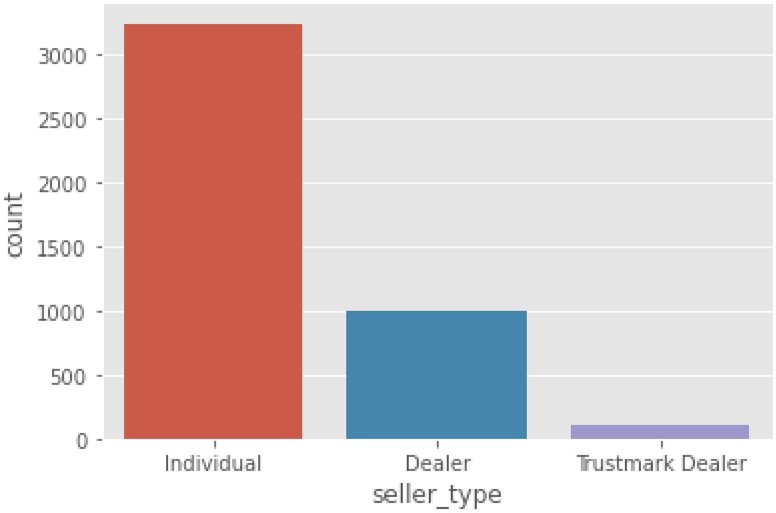
In [18]:

```python
#Bar Graph
authority=dataset['seller_type']
sns.countplot(authority)
```

```
D:\ANACONDA\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x.
From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

Out[18]:

```
<AxesSubplot:xlabel='seller_type', ylabel='count'>
```



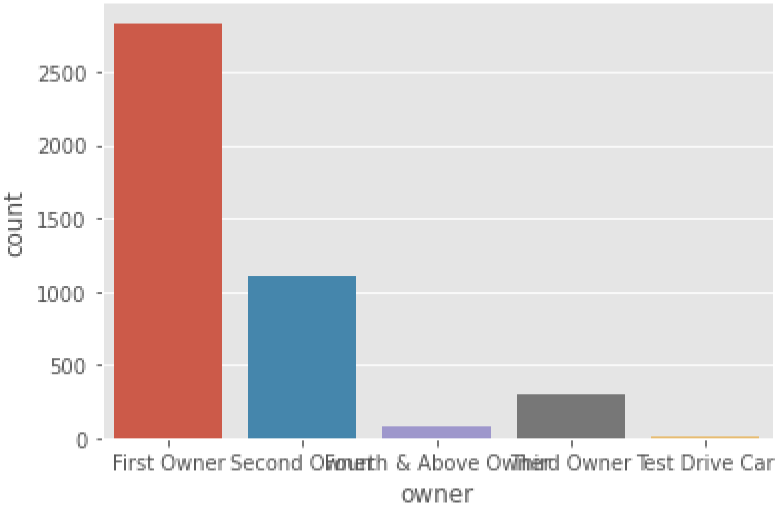
In [19]:

```
#Bar Graph
owned=dataset['owner']
sns.countplot(owned)
```

```
D:\ANACONDA\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x.
From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

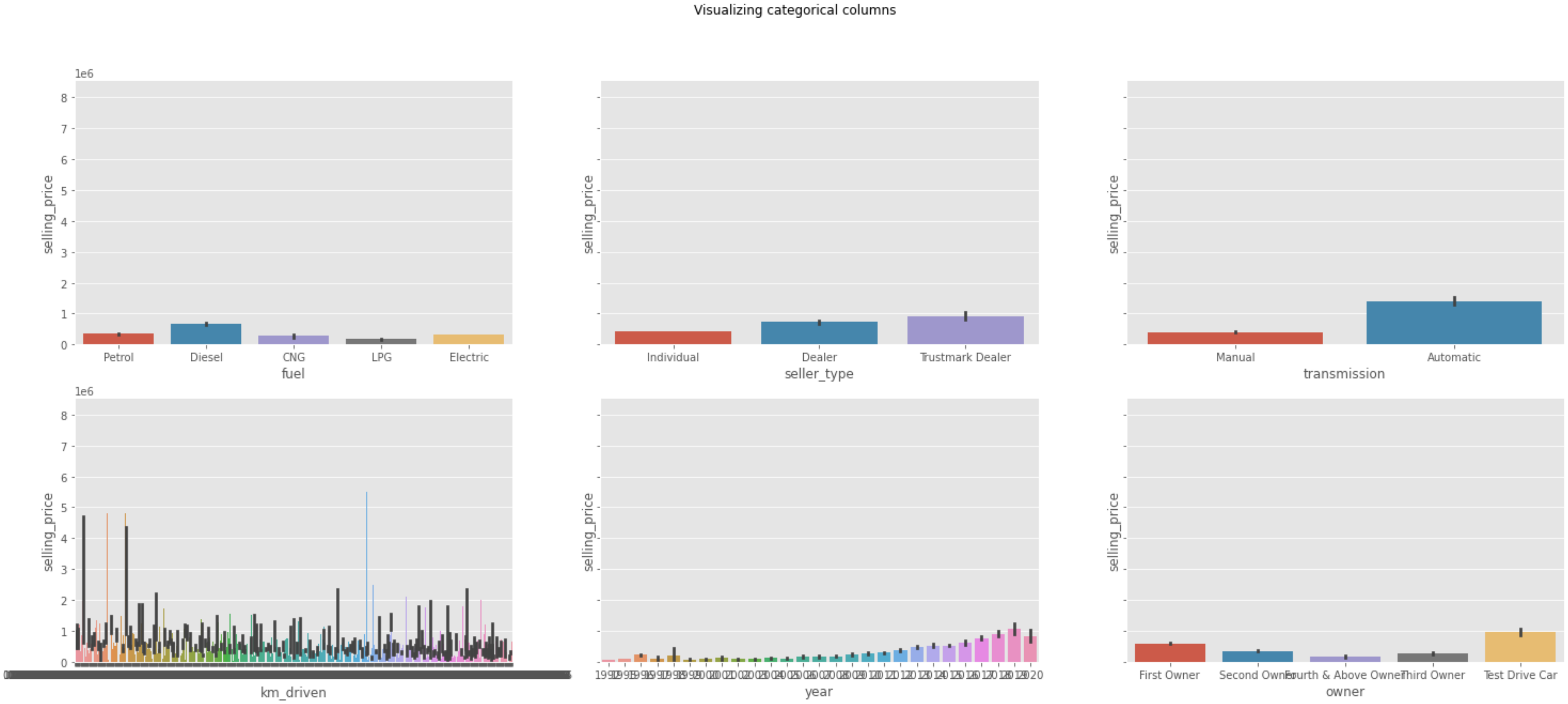
Out[19]:

```
<AxesSubplot:xlabel='owner', ylabel='count'>
```



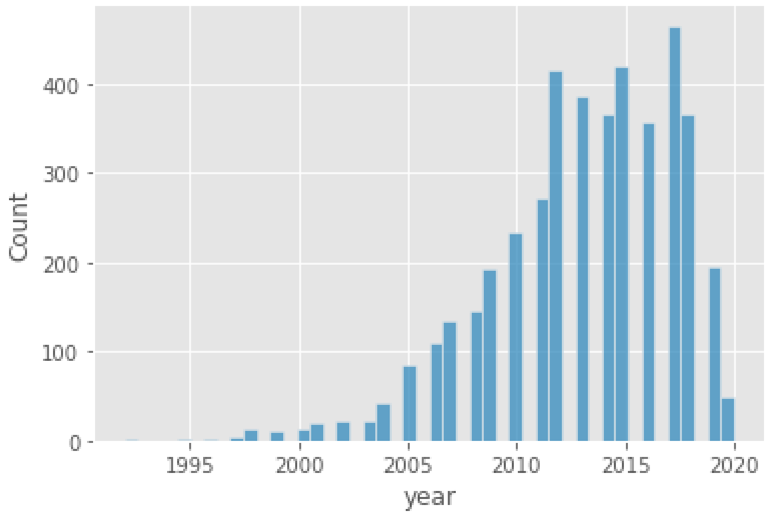
In [20]:
```
#Bar Graph
fig, axes = plt.subplots(2,3,figsize=(25,10), sharey=True)
fig.suptitle('Visualizing categorical columns')
sns.barplot(x=fuel, y=selling_price, ax=axes[0][0])
sns.barplot(x=seller_type, y=selling_price, ax=axes[0][1])
sns.barplot(x=transmission_type, y=selling_price, ax=axes[0][2])
sns.barplot(x=km_driven, y=selling_price, ax=axes[1][0])
sns.barplot(x=year, y=selling_price, ax=axes[1][1])
sns.barplot(x=owner, y=selling_price, ax=axes[1][2])
```

Out[20]:
```
<AxesSubplot:xlabel='owner', ylabel='selling_price'>
```



In [21]:
```
#histogram
sns.histplot(year)
```

Out[21]:
```
<AxesSubplot:xlabel='year', ylabel='Count'>
```

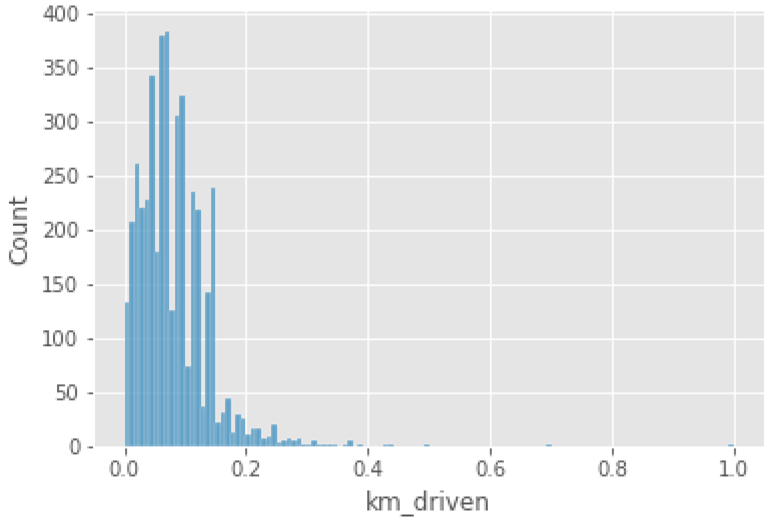


In [22]:

```
#histogram
sns.histplot(km_driven)
```

Out[22]:

```
<AxesSubplot:xlabel='km_driven', ylabel='Count'>
```

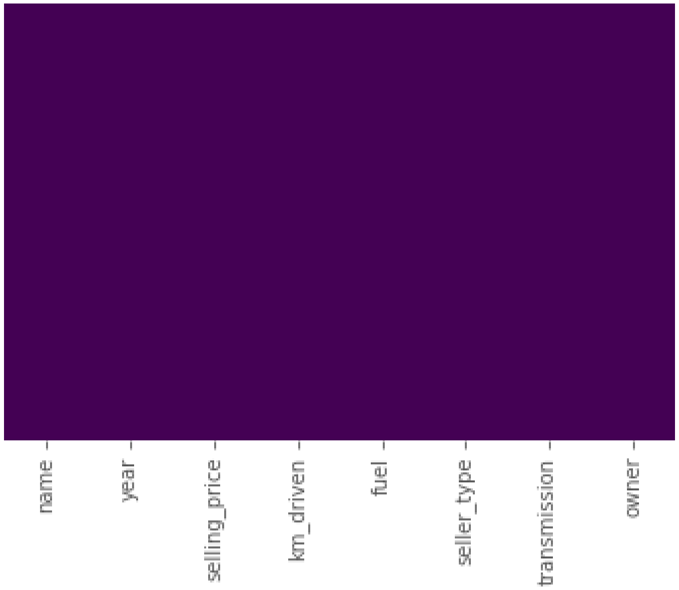


In [23]:

```
#Heat map plot
sns.heatmap(dataset.isnull(), cbar=False, yticklabels=False , cmap='viridis')
```

```
<AxesSubplot:>
```

Out[23]:



In [24]:

```
#Selling Price
y=dataset['selling_price']
```

In [25]:

```
y
```

Out[25]:

```
0        60000
1       135000
2       600000
3       250000
4       450000
         ...  
4335    409999
4336    409999
4337    110000
4338    865000
4339    225000
Name: selling_price, Length: 4340, dtype: int64
```

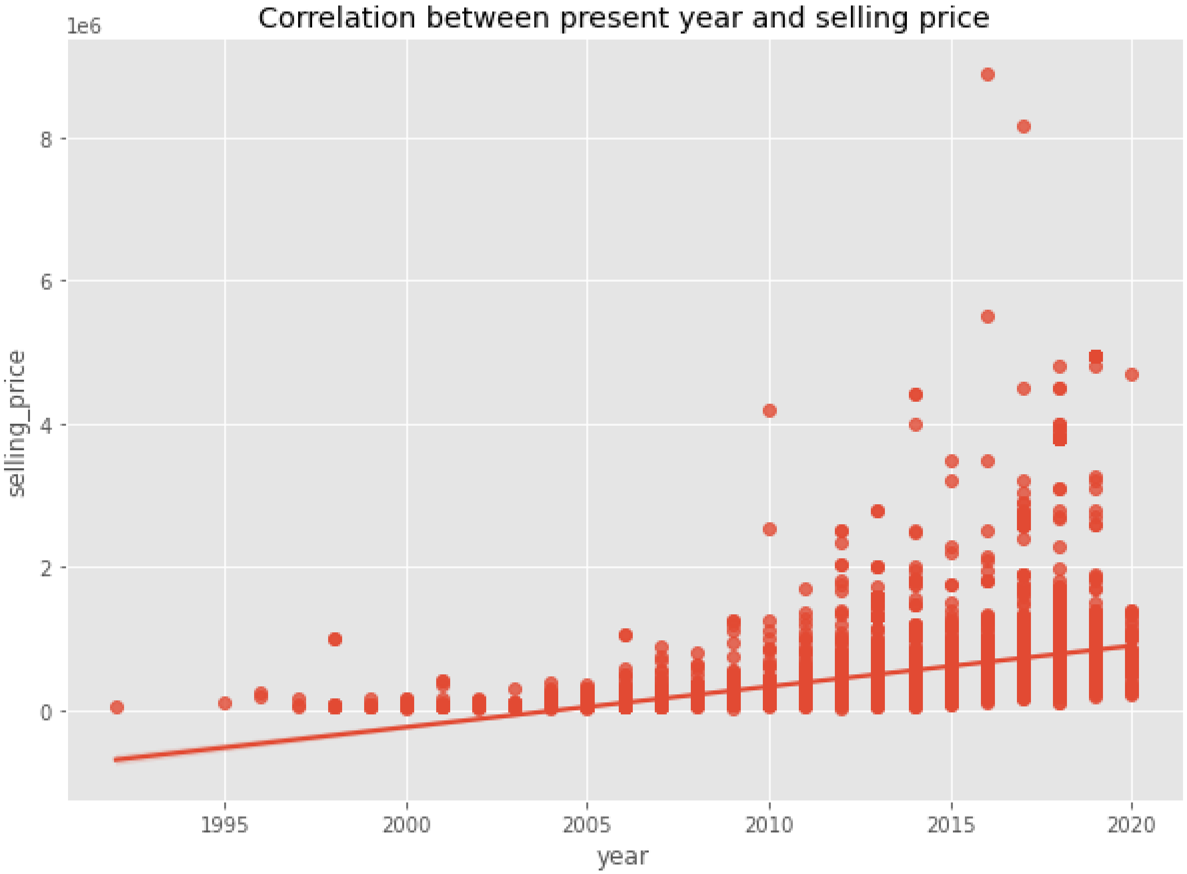
In [26]:

```
#Correlation between present year and selling price
fig=plt.figure(figsize=(10,7))
```

```
plt.title('Correlation between present year and selling price')
sns.regplot(x='year', y='selling_price', data=dataset)
```

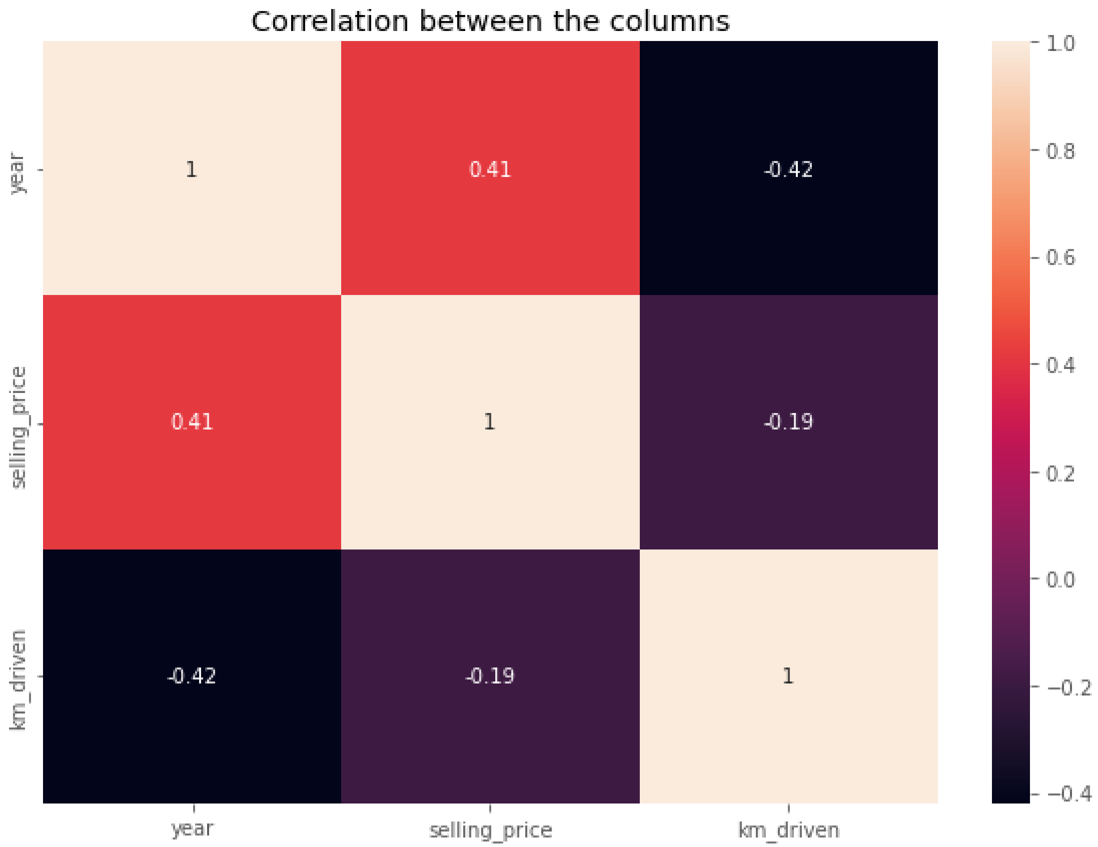
Out[26]:

```
<AxesSubplot:title={'center':'Correlation between present year and selling price'}, xlabel='year', ylabel='selling_pric
e'>
```



In [27]:

```
#Correlation between the columns
plt.figure(figsize=(10,7))
sns.heatmap(dataset.corr(), annot=True)
plt.title('Correlation between the columns')
plt.show()
#Ligher color relate a high value of corelation
```


Correlation between the columns
year
selling_price
km_driven
1
0.41
-0.42
0.41
1
-0.19
-0.42
-0.19
1
year
selling_price
km_driven
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4