** EDA 1 Bike Details (ML Module 2)**

In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

df = pd.read_csv('BIKE DETAILS.csv')
```

In [5]:
```
df
```

Out[5]:

	name	selling_price	year	seller_type	owner	km_driven	ex_showroom_price
0	Royal Enfield Classic 350	175000	2019	Individual	1st owner	350	NaN
1	Honda Dio	45000	2017	Individual	1st owner	5650	NaN
2	Royal Enfield Classic Gunmetal Grey	150000	2018	Individual	1st owner	12000	148114.0
3	Yamaha Fazer FI V 2.0 [2016-2018]	65000	2015	Individual	1st owner	23000	89643.0
4	Yamaha SZ [2013-2014]	20000	2011	Individual	2nd owner	21000	NaN
...	...	...	...	...	...	...	...
1056	Activa 3g	17000	2010	Individual	1st owner	500000	52000.0
1057	Honda CB twister	16000	2012	Individual	1st owner	33000	51000.0
1058	Bajaj Discover 125	15000	2013	Individual	2nd owner	35000	57000.0
1059	Honda CB Shine	12000	2009	Individual	1st owner	53000	58000.0
1060	Bajaj Pulsar 150	10000	2008	Individual	1st owner	92233	75000.0

1061 rows × 7 columns

In [ ]:
```
## Q1 What it is the range of selling prices in the dataset?

print(df.selling_price.min())
print(df.selling_price.max())
```

```
5000
760000
```

In [4]:
```
# Q2 What is the median selling price for bikes in the dataset?

print(df.selling_price.median())
```

```
45000.0
```

In [5]:
```
# Q3 What is the most common seller type in the dataset?

print(df.seller_type.mode())
```

```
0    Individual
Name: seller_type, dtype: object
```

In [9]:
```
# Q4 How many bike have driven more than 50000 kms?

print(df[df.km_driven > 50000].shape[0])
```

```
170
```

In [10]:
```
# Q5 What is the average km_driven value for each ownership type?

print(df.groupby('seller_type').km_driven.mean())
```

```
seller_type
Dealer        35258.833333
Individual    34354.720379
Name: km_driven, dtype: float64
```

In [6]:
```
# Q6 What proportion of bikes are from the year 2015 or older?

print(df[df.year <= 2015].shape[0] / df.shape[0])
```

```
0.5664467483506126
```

In [7]:
```
# Q7 What is the trend of missing values across the dataset?

print(df.isnull().sum())
```

```
name                  0
selling_price         0
year                  0
seller_type           0
owner                 0
km_driven             0
ex_showroom_price    435
dtype: int64
```

In [10]:
```
# Q8 What is the highest ex_showroom_price recorded, and for which bike?

print(df.ex_showroom_price.max())
#for which bike
print(df[df.ex_showroom_price == df.ex_showroom_price.max()].name)
```

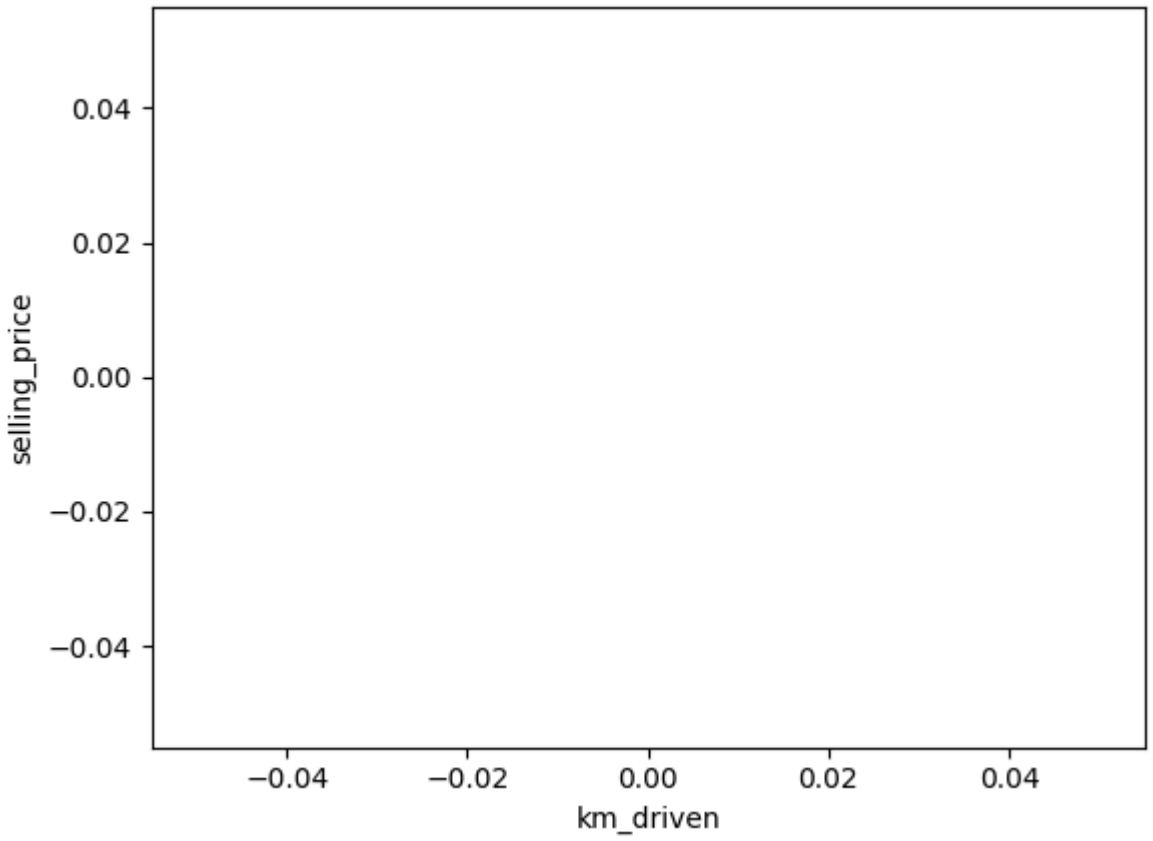
```
1278000.0
134    Harley-Davidson Street Bob
Name: name, dtype: object
```

In [11]:
```
# Q9 What is the total number of bikes listed by each seller type?

print(df.groupby('seller_type').name.count())
```

```
seller_type
Dealer           6
Individual    1055
Name: name, dtype: int64
```

In [13]:
```
# Q10 What is the relationship between Selling_price and km_driven for first-own

print(df[df.seller_type == 'First Owner'].plot.scatter(x='km_driven', y='selling
```

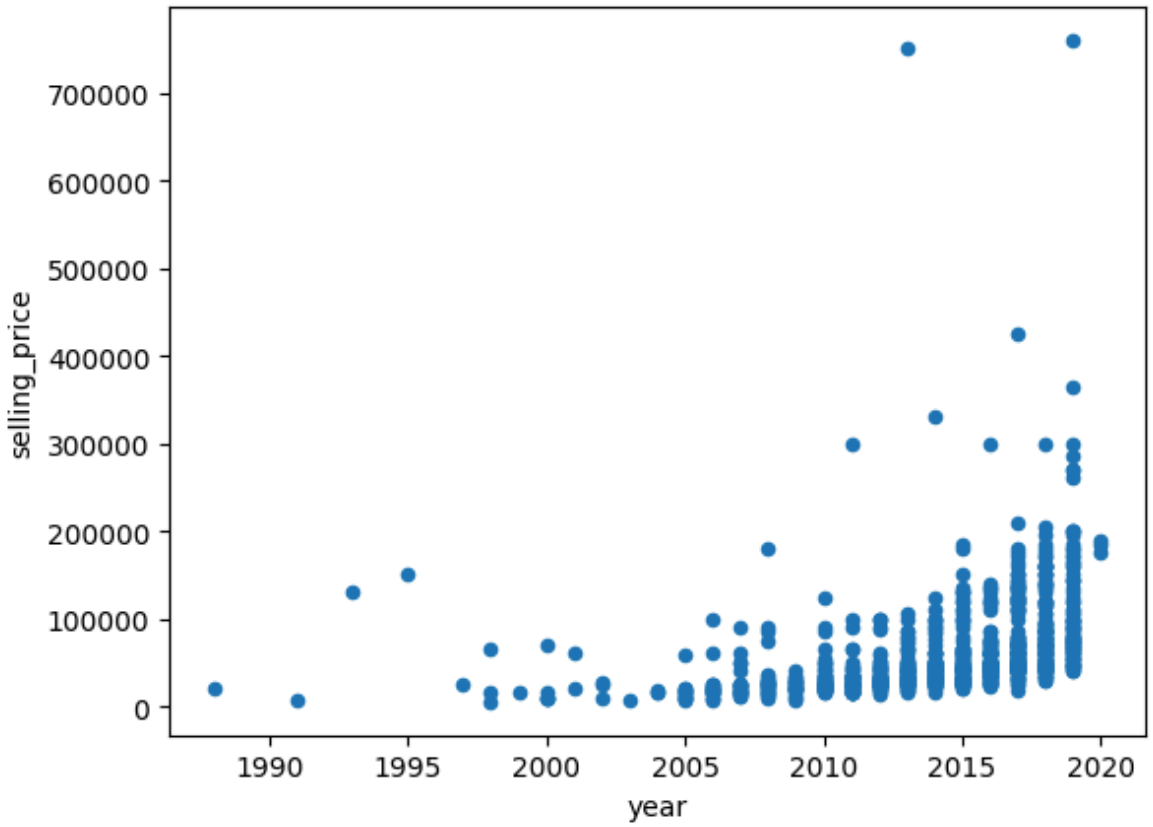
```
Axes(0.125,0.11;0.775x0.77)
```



In [16]:
```
# Q11 Identify and remove ourliers in the km_driven column using the IQR method.

Q1 = df.km_driven.quantile(0.25)
Q3 = df.km_driven.quantile(0.75)
IQR = Q3 - Q1

df = df[~((df.km_driven < (Q1 - 1.5 * IQR)) | (df.km_driven > (Q3 + 1.5 * IQR)))
```

In [17]:
```
# Q12 Perform a bivariate analysis to visualize the relationship between year an

print(df.plot.scatter(x='year', y='selling_price'))
```

```
Axes(0.125,0.11;0.775x0.77)
```



In [18]:
```
# Q13 What is the average depreciation in selling price based on the bike's age

print(df.year.max() - df.year.min())
```

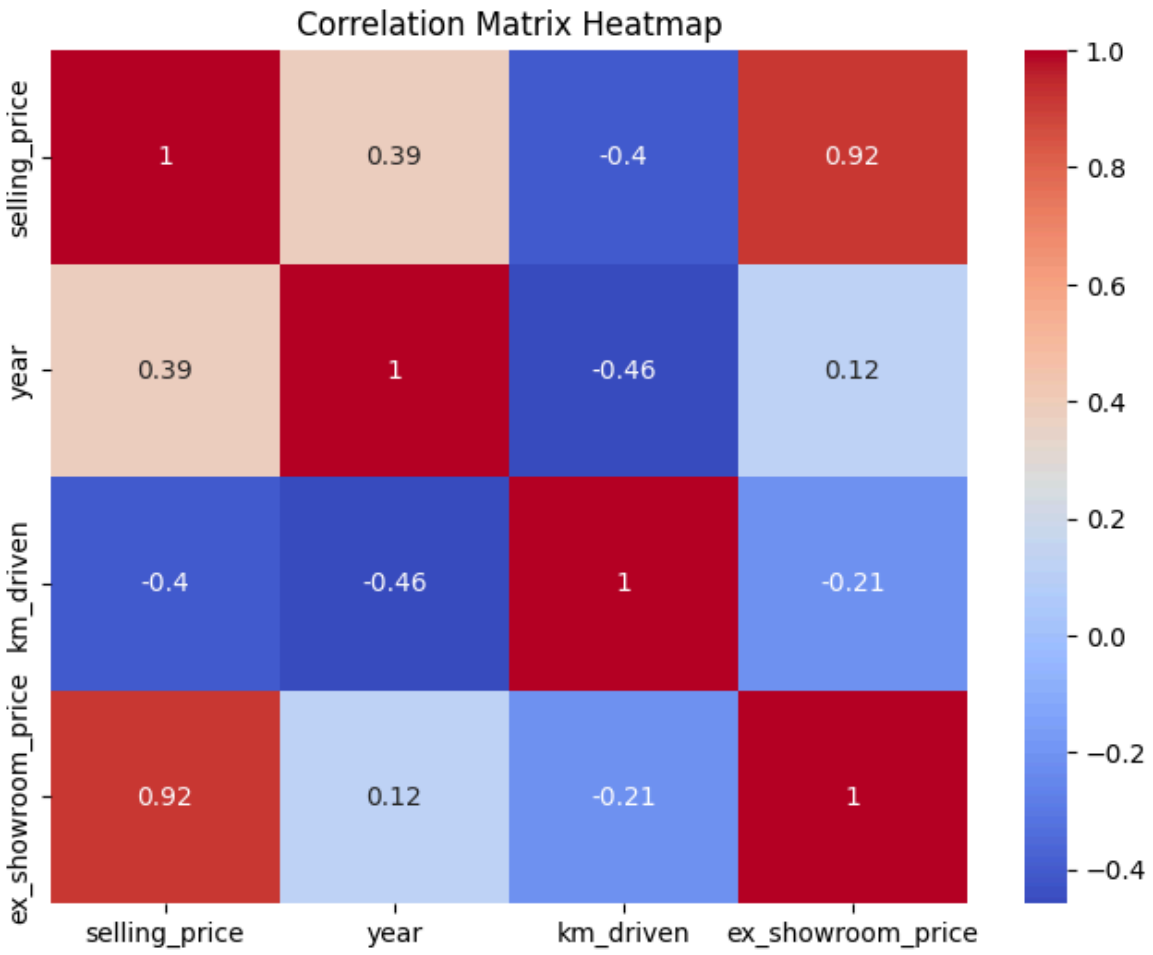
```
32
```

In [22]:
```
# Q14 Which bike names are priced significantly above the average price for thei

print(df.groupby('year').selling_price.mean())
# For which bike
mean_price_per_year = df.groupby('year')['selling_price'].transform('mean')
print(df[df.selling_price > mean_price_per_year].name)
```

```
year
1988     20000.000000
1991      6000.000000
1993    130000.000000
1995    150000.000000
1997     25000.000000
1998     28333.333333
1999     15000.000000
2000     20833.333333
2001     40000.000000
2002     20666.666667
2003      8000.000000
2004     16000.000000
2005     17669.230769
2006     23821.052632
2007     27768.181818
2008     37004.166667
2009     23295.454545
2010     32350.877193
2011     35859.631579
2012     36720.619048
2013     51802.816901
2014     49121.348315
2015     56313.131313
2016     57924.126214
2017     78962.121212
2018     87660.374046
2019    119689.511628
2020    183333.333333
Name: selling_price, dtype: float64
0                   Royal Enfield Classic 350
2         Royal Enfield Classic Gunmetal Grey
3           Yamaha Fazer FI V 2.0 [2016-2018]
7        Royal Enfield Bullet 350 [2007-2011]
13                             Yamaha YZF R3
                         ...
1004                    Bajaj Pulsar  NS 200
1005                      TVS Apache RTR 160
1008                      Bajaj Pulsar 220 F
1012                     Bajaj Pulsar NS 200
1023                   Bajaj Avenger 220 dtsi
Name: name, Length: 316, dtype: object
```

In [24]:
```
# Q15 Develop a correlation matrix for numeric columns and visualize it using a

corr_matrix = df.select_dtypes(include=[np.number]).corr()
print(corr_matrix)

plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix Heatmap')
plt.show()
```

```
                  selling_price      year  km_driven  ex_showroom_price
selling_price           1.000000  0.387068  -0.402240           0.919747
year                    0.387068  1.000000  -0.458364           0.117731
km_driven              -0.402240 -0.458364   1.000000          -0.205250
ex_showroom_price       0.919747  0.117731  -0.205250           1.000000
```


Correlation Matrix Heatmap
selling_price
year
km_driven
ex_showroom_price
1
0.39
-0.4
0.92
0.39
1
-0.46
0.12
-0.4
-0.46
1
-0.21
0.92
0.12
-0.21
1
selling_price
year
km_driven
ex_showroom_price
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

In [ ]: