In [1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

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

Out[2]:

	Unnamed: 0	Id	year	brand	full_model_name	model_name	price	distance_travelled(kms)	fuel_type	city	...	new and less used	inv_car_price	inv_car_dist	inv_car_age	inv_brand	std_invpri
0	0	0	2016	Honda	Honda Brio S MT	Brio	425000.0	9680.0	Petrol	Mumbai	...	0	2.352941e-06	0.000103	0.200000	0.142857	0.1434
1	1	1	2012	Nissan	Nissan Sunny XV Diesel	Sunny	325000.0	119120.0	Diesel	Mumbai	...	0	3.076923e-06	0.000008	0.111111	0.090909	0.1888
2	2	2	2017	Toyota	Toyota Fortuner 2.8 4x2 MT [2016-2020]	Fortuner	2650000.0	64593.0	Diesel	Thane	...	0	3.773585e-07	0.000015	0.250000	1.000000	0.0194
3	3	3	2017	Mercedes-Benz	Mercedes-Benz E-Class E 220d Expression [2019-...	E-Class	4195000.0	25000.0	Diesel	Mumbai	...	1	2.383790e-07	0.000040	0.250000	0.500000	0.0106
4	4	4	2012	Hyundai	Hyundai Verna Fluidic 1.6 CRDi SX	Verna	475000.0	23800.0	Diesel	Mumbai	...	0	2.105263e-06	0.000042	0.111111	0.071429	0.1278
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1720	1720	1720	2015	Hyundai	Hyundai Eon Era +	Eon	290000.0	38000.0	Petrol	Pune	...	0	3.448276e-06	0.000026	0.166667	0.071429	0.2121
1721	1721	1721	2011	Bentley	Bentley Continental Flying Spur W12	Continental	7500000.0	36000.0	Petrol	Pune	...	0	1.333333e-07	0.000028	0.100000	0.022727	0.0040
1722	1722	1722	2008	Mahindra-Renault	Mahindra-Renault Logan DLE 1.5 dci	Logan	185000.0	142522.0	Diesel	Pune	...	0	5.405405e-06	0.000007	0.076923	0.041667	0.3350
1723	1723	1723	1990	Mahindra	Mahindra Jeep CJ 500 D	Jeep	325000.0	18581.0	Diesel	Pune	...	0	3.076923e-06	0.000054	0.032258	0.041667	0.1888
1724	1724	1724	2017	Hyundai	Hyundai Creta SX Plus 1.6 AT CRDI	Creta	1395000.0	31028.0	Diesel	Pune	...	0	7.168459e-07	0.000032	0.250000	0.071429	0.0407

1725 rows × 23 columns

In [3]:
```python
df.drop(['Unnamed: 0','Id','year','full_model_name','model_name','brand_rank','inv_car_age','inv_car_price','inv_car_dist','inv_brand','std_invprice','std_inv
df
```

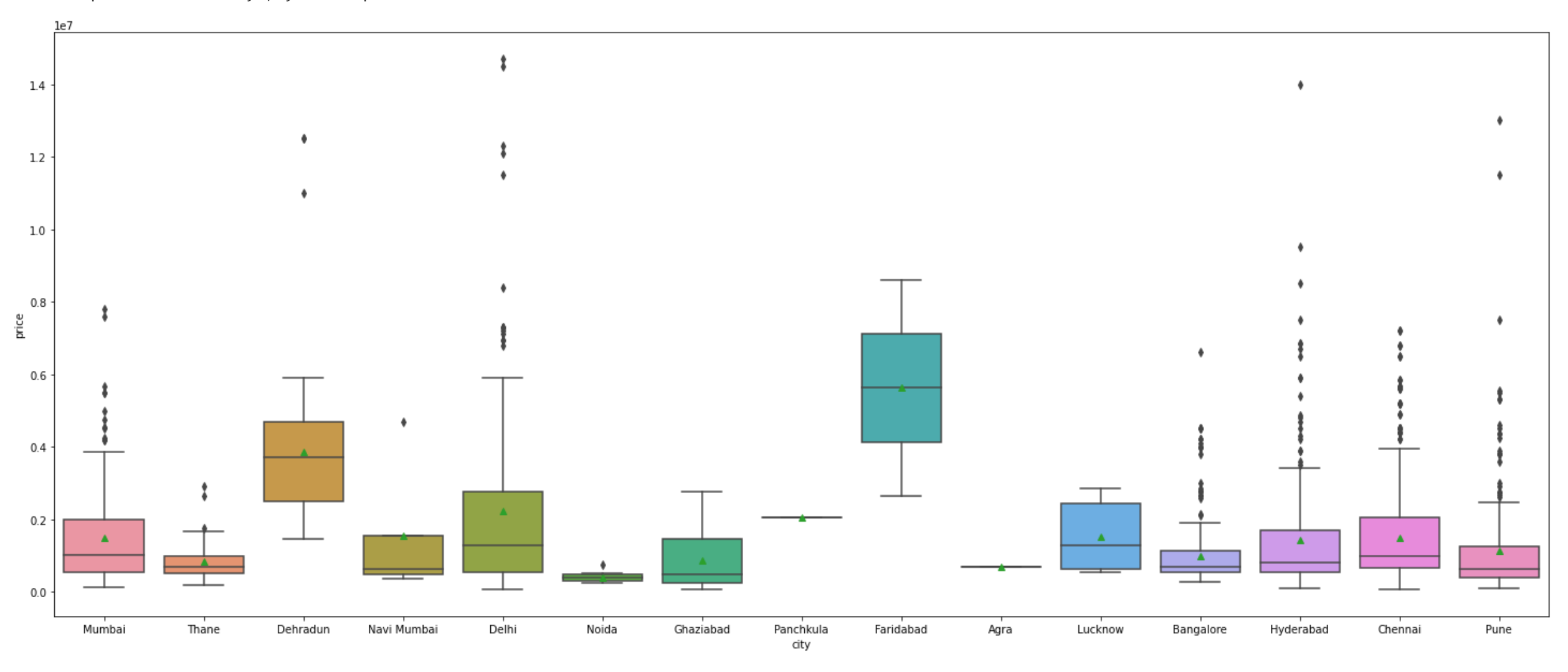
Out[3]:

	brand	price	distance_travelled(kms)	fuel_type	city	car_age	distance below 30k km	new and less used
0	Honda	425000.0	9680.0	Petrol	Mumbai	5.0	1	0
1	Nissan	325000.0	119120.0	Diesel	Mumbai	9.0	0	0
2	Toyota	2650000.0	64593.0	Diesel	Thane	4.0	0	0
3	Mercedes-Benz	4195000.0	25000.0	Diesel	Mumbai	4.0	1	1
4	Hyundai	475000.0	23800.0	Diesel	Mumbai	9.0	1	0
...	...	...	...	...	...	...	...	...
1720	Hyundai	290000.0	38000.0	Petrol	Pune	6.0	0	0
1721	Bentley	7500000.0	36000.0	Petrol	Pune	10.0	0	0
1722	Mahindra-Renault	185000.0	142522.0	Diesel	Pune	13.0	0	0
1723	Mahindra	325000.0	18581.0	Diesel	Pune	31.0	1	0
1724	Hyundai	1395000.0	31028.0	Diesel	Pune	4.0	0	0

1725 rows × 8 columns

In [4]:
```python
plt.figure(figsize=(25,10))
sns.boxplot(x='city',y='price',data=df, showmeans=True)
```

Out[4]: <AxesSubplot:xlabel='city', ylabel='price'>



In [5]:
```python
city_rank={'Noida':1,'Ghaziabad':2,'Pune':3,'Navi Mumbai':4,'Thane':5,'Bangalore':6,'Agra':7,
          'Hyderabad':8,'Chennai':9,'Mumbai':10,'Delhi':11,'Lucknow':12,'Panchkula':13,'Dehradun':14,
          'Faridabad':15}
df['city_rank']=df.city.map(city_rank)
df.drop('city',axis=1,inplace=True)
df
```

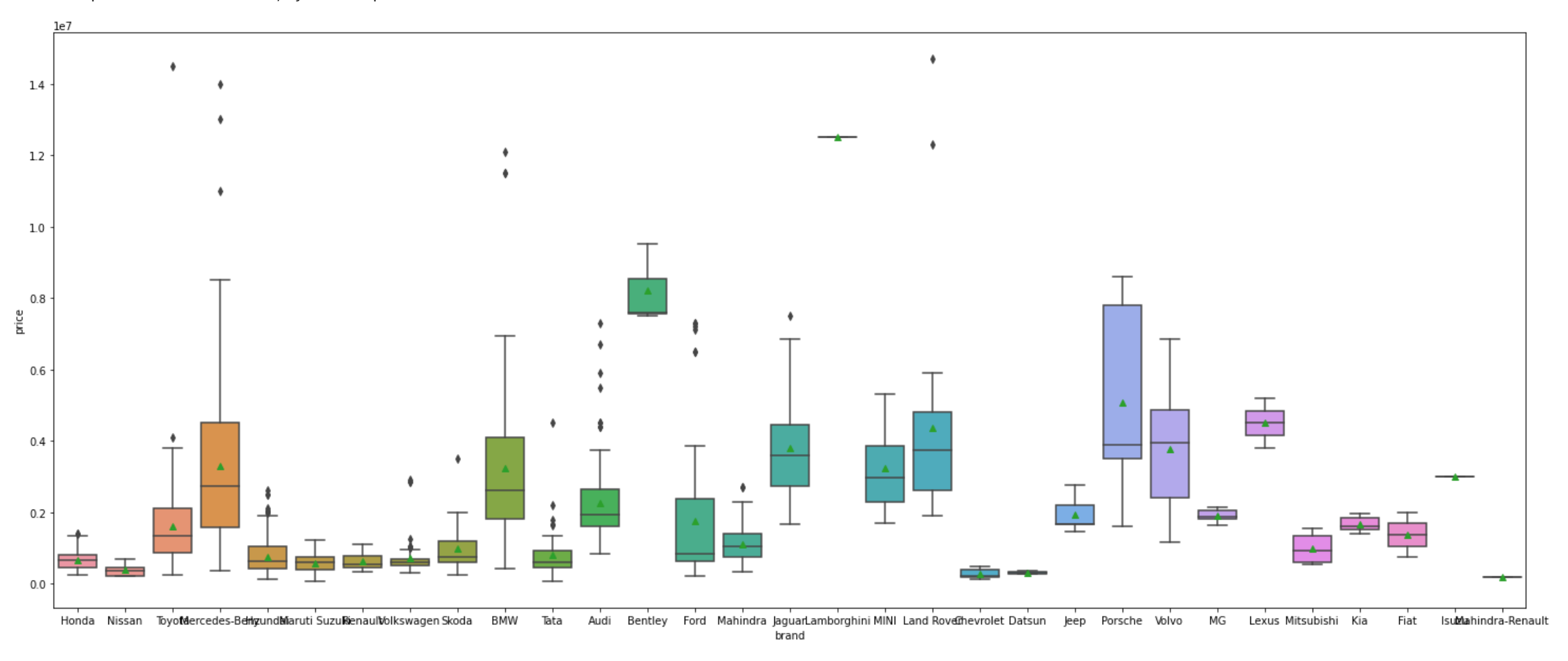
Out[5]:

	brand	price	distance_travelled(kms)	fuel_type	car_age	distance below 30k km	new and less used	city_rank
0	Honda	425000.0	9680.0	Petrol	5.0	1	0	10
1	Nissan	325000.0	119120.0	Diesel	9.0	0	0	10
2	Toyota	2650000.0	64593.0	Diesel	4.0	0	0	5
3	Mercedes-Benz	4195000.0	25000.0	Diesel	4.0	1	1	10
4	Hyundai	475000.0	23800.0	Diesel	9.0	1	0	10
...	...	...	...	...	...	...	...	...
1720	Hyundai	290000.0	38000.0	Petrol	6.0	0	0	3
1721	Bentley	7500000.0	36000.0	Petrol	10.0	0	0	3
1722	Mahindra-Renault	185000.0	142522.0	Diesel	13.0	0	0	3
1723	Mahindra	325000.0	18581.0	Diesel	31.0	1	0	3
1724	Hyundai	1395000.0	31028.0	Diesel	4.0	0	0	3

1725 rows × 8 columns

In [6]:
```python
plt.figure(figsize=(25,10))
sns.boxplot(x='brand',y='price',data=df, showmeans=True)
```

Out[6]: <AxesSubplot:xlabel='brand', ylabel='price'>



In [7]:
```python
df['brand'].value_counts()
```

Out[7]:
```
Hyundai             297
Maruti Suzuki       275
Honda               153
Mercedes-Benz       131
Toyota              117
BMW                 111
Audi                 98
Mahindra             93
Ford                 78
Volkswagen           70
Renault              51
Skoda                45
Tata                 44
Jaguar               30
Land Rover           28
Volvo                17
Nissan               13
Kia                  12
MG                   10
MINI                 10
Jeep                 10
Chevrolet             9
Porsche               5
Mitsubishi            4
Datsun                3
Bentley               3
Lamborghini           2
Fiat                  2
Lexus                 2
Mahindra-Renault      1
Isuzu                 1
Name: brand, dtype: int64
```

In [8]:
```python
brand_rank={'Mahindra-Renault':1,'Chevrolet':2,'Datsun':3,'Renault':4,'Volkswagen':5,'Maruti Suzuki':6,
            'Nissan':7,'Tata':8,'Hyundai':9,'Honda':10,'Skoda':11,'Ford':12,'Mitsubishi':13,
            'Mahindra':14,'Toyota':15,'Fiat':16,'Kia':17,'Jeep':18,'MG':19,'Audi':20,'BMW':21,
            'Mercedes-Benz':22,'MINI':23,'Isuzu':24,'Jaguar':25,'Land Rover':26,'Porsche':27,
            'Volvo':28,'Lexus':29,'Bentley':30,'Lamborghini':31}
df['brand_rank']=df.brand.map(brand_rank)
df.drop('brand',axis=1,inplace=True)
df
```

Out[8]:

	price	distance_travelled(kms)	fuel_type	car_age	distance below 30k km	new and less used	city_rank	brand_rank
0	425000.0	9680.0	Petrol	5.0	1	0	10	10
1	325000.0	119120.0	Diesel	9.0	0	0	10	7
2	2650000.0	64593.0	Diesel	4.0	0	0	5	15
3	4195000.0	25000.0	Diesel	4.0	1	1	10	22
4	475000.0	23800.0	Diesel	9.0	1	0	10	9
...	...	...	...	...	...	...	...	...
1720	290000.0	38000.0	Petrol	6.0	0	0	3	9
1721	7500000.0	36000.0	Petrol	10.0	0	0	3	30
1722	185000.0	142522.0	Diesel	13.0	0	0	3	1
1723	325000.0	18581.0	Diesel	31.0	1	0	3	14
1724	1395000.0	31028.0	Diesel	4.0	0	0	3	9

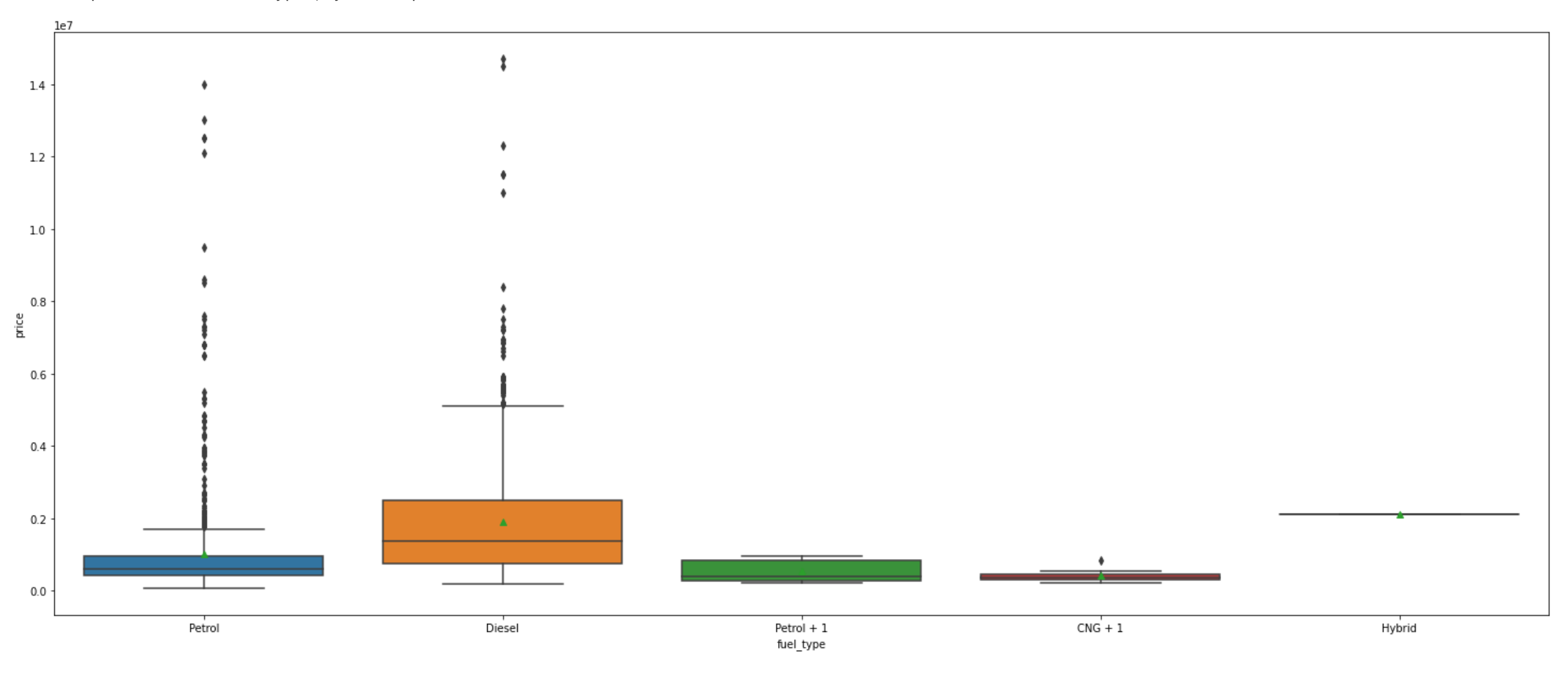
1725 rows × 8 columns

In [9]:
```python
df['fuel_type'].value_counts()
```

Out[9]:
```
Diesel        922
Petrol        788
CNG + 1         8
Petrol + 1      6
Hybrid          1
Name: fuel_type, dtype: int64
```

In [10]:
```python
plt.figure(figsize=(25,10))
sns.boxplot(x='fuel_type',y='price',data=df, showmeans=True)
```

Out[10]: <AxesSubplot:xlabel='fuel_type', ylabel='price'>



In [11]:
```python
fuel_rank={'CNG + 1':1,'Petrol + 1':2,'Petrol':3,'Diesel':4,'Hybrid':5}
df['fuel_rank']=df.fuel_type.map(fuel_rank)
df.drop('fuel_type',axis=1,inplace=True)
df
```

Out[11]:

	price	distance_travelled(kms)	car_age	distance below 30k km	new and less used	city_rank	brand_rank	fuel_rank
0	425000.0	9680.0	5.0	1	0	10	10	3
1	325000.0	119120.0	9.0	0	0	10	7	4
2	2650000.0	64593.0	4.0	0	0	5	15	4
3	4195000.0	25000.0	4.0	1	1	10	22	4
4	475000.0	23800.0	9.0	1	0	10	9	4
...	...	...	...	...	...	...	...	...
1720	290000.0	38000.0	6.0	0	0	3	9	3
1721	7500000.0	36000.0	10.0	0	0	3	30	3
1722	185000.0	142522.0	13.0	0	0	3	1	4
1723	325000.0	18581.0	31.0	1	0	3	14	4
1724	1395000.0	31028.0	4.0	0	0	3	9	4

1725 rows × 8 columns

In [ ]:

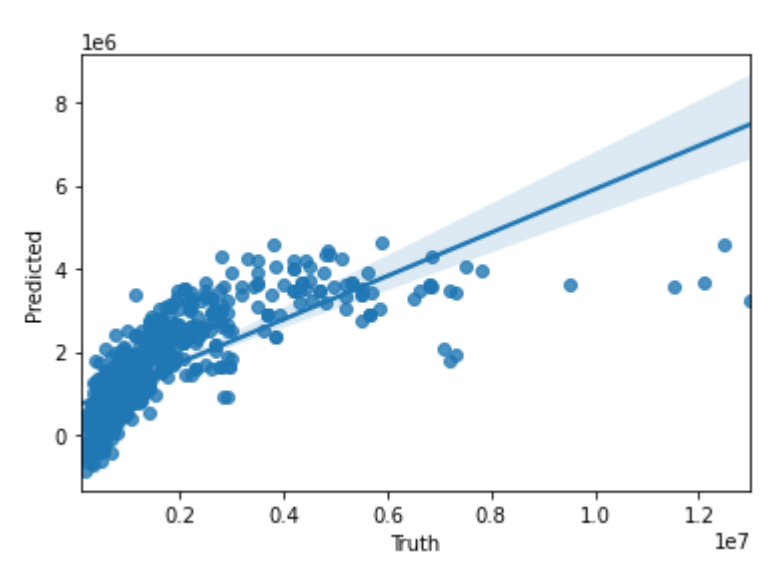
In [12]:
```python
from sklearn.model_selection import train_test_split as tts
x=df.drop('price',axis=1)
y=df['price']

x_train,x_test, y_train, y_test=tts(x,y,test_size=0.4, random_state=10)
```

In [ ]:

In [13]:
```python
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
y_lr_pred=lr.predict(x_test)
sns.regplot(y_test,y_lr_pred)
plt.xlabel('Truth')
plt.ylabel('Predicted')
```

Out[13]: Text(0, 0.5, 'Predicted')



In [14]:
```python
lr.score(x_train,y_train)
```

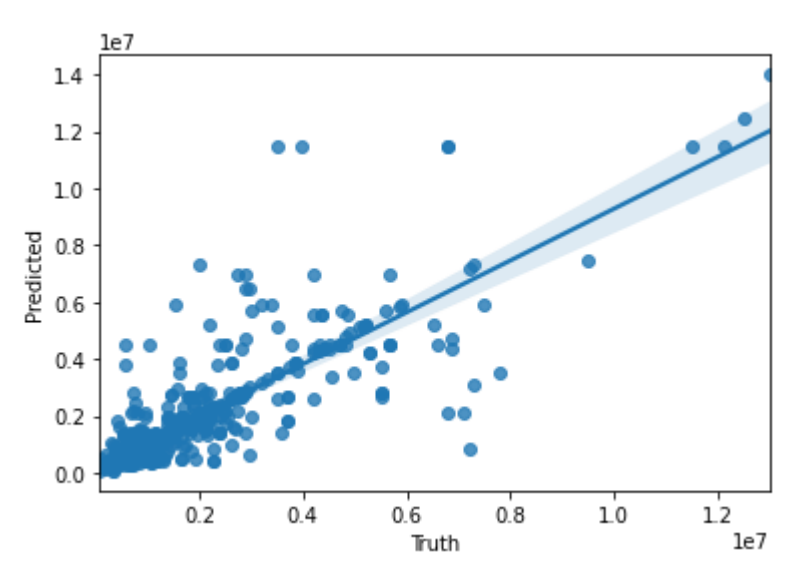
Out[14]: 0.5179056583341175

In [15]:
```python
lr.score(x_test,y_test)
```

Out[15]: 0.558751961574397

In [16]:
```python
from sklearn.tree import DecisionTreeRegressor
dtr=DecisionTreeRegressor()
dtr.fit(x_train,y_train)
y_dtr_pred=dtr.predict(x_test)
sns.regplot(y_test,y_dtr_pred)
plt.xlabel('Truth')
plt.ylabel('Predicted')
```

Out[16]: Text(0, 0.5, 'Predicted')



In [17]:
```python
dtr.score(x_train,y_train)
```

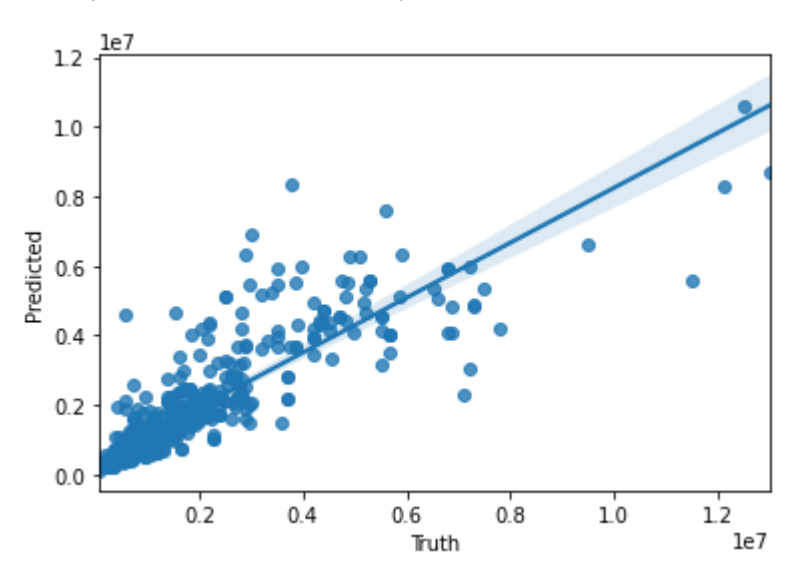
Out[17]: 0.999987584760231

In [18]:
```python
dtr.score(x_test,y_test)
```

Out[18]: 0.6494133587731714

In [19]:
```python
from sklearn.ensemble import RandomForestRegressor
rfr=RandomForestRegressor(n_estimators = 100,
                          criterion = 'mse',
                          random_state = 20)
rfr.fit(x_train,y_train)
y_rfr_pred=rfr.predict(x_test)
sns.regplot(y_test,y_rfr_pred)
plt.xlabel('Truth')
plt.ylabel('Predicted')
```

Out[19]: Text(0, 0.5, 'Predicted')



In [20]:
```python
rfr.score(x_train,y_train)
```

Out[20]: 0.9483256653123986

In [21]:
```python
rfr.score(x_test,y_test)
```

Out[21]: 0.7713400711391021

In [ ]:

In [ ]:

In [22]:
```python
lr.predict([[64000.0,4.0,0,0,5,15,4]])
```

Out[22]: array([1962017.52953527])

In [23]:
```python
dtr.predict([[64000.0,4.0,0,0,5,15,4]])
```

Out[23]: array([2650000.])

In [24]:
```python
rfr.predict([[64000.0,4.0,0,0,5,15,4]])
```

Out[24]: array([2445300.])

In [ ]: