In [3]:
```
import sqlite3
import numpy as np
```

In [4]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:
```
import pandas as pd
```

In [6]:
```
data=pd.read_csv(r'Book1.csv', encoding= 'unicode_escape')
```

In [45]:
```
df=pd.DataFrame(data ,columns=["Model"])
```

In [69]:
```
df2=pd.DataFrame(data,columns=["Brand","Model"])
df2
```

Out[69]:

	Brand	Model
0	Toyota	Prado
1	Suzuki	Bolan
2	Suzuki	Bolan
3	Suzuki	Alto
4	Toyota	Corolla XLI
...	...	...
24964	Toyota	Corolla XE
24965	Daihatsu	Cuore
24966	Other Brands	Other
24967	Suzuki	Alto
24968	Toyota	Corolla GLI

24969 rows × 2 columns

```
In [72]: df2.drop_duplicates()
```

```
Out[72]:
              Brand            Model
          0   Toyota           Prado
          1   Suzuki           Bolan
          3   Suzuki            Alto
          4   Toyota     Corolla XLI
          5   Toyota  Corrolla Altis
        ...      ...             ...
      22217   Toyota             MR2
      22551    Honda            HR-V
      22666   Nissan          Sylphy
      23151    Lexus       Is Series
      24932  Hyundai        Shehzore

340 rows × 2 columns
```

```
In [8]: Model=df["Model"].unique()
```

In [64]:

```
Model
```

Out[64]:

```
array(['Prado', 'Bolan', 'Alto', 'Corolla XLI', 'Corrolla Altis',
       'Cultus VXL', 'Civic VTi', 'Khyber', 'Liana', 'Passo',
       'Civic Prosmetic', 'Civic EXi', 'Charade', 'Pajero Mini',
       'Margalla', 'City IVTEC', 'Classic', 'Other', 'Corolla GLI',
       'Cultus VXR', 'Dayz Highway Star', 'Mehran VX', 'Vitz',
       'Mehran VXR', 'Carry', 'Cultus VX', 'Baleno', 'Mira',
       'Civic VTi Oriel Prosmatec', 'Cuore', 'Corolla 2.0 D',
       'Corolla XE', 'Surf', 'FX', 'City IDSI', 'Premio', 'Sprinter',
       '3 Series', 'Hilux', 'Lancer', 'Swift', 'Estima', 'Vamos',
       'Starlet', 'Prius', 'Joy', '323', 'Racer', 'Sunny', 'Accord',
       'Zest', 'AD Van', 'APV', 'March', 'Pride', 'Sportage',
       'Terios Kid', 'Santro', 'Fit', 'V2', 'City Aspire',
       'Civic VTi Oriel', 'BR-V', 'Kei', 'Probox', 'Hijet', '86', 'Yaris',
       'Aygo', 'Rush', 'Dayz', 'Fortuner', 'Every', 'Camry', 'Aqua',
       'Corolla Assista', 'Sera', 'Acty', 'Moco', 'Wagon R', 'Smart',
       'Demio', 'Corona', 'Every Wagon', 'Civic Hybrid', 'Duet', 'Vezel',
       'Corolla Axio', 'City Vario', 'Corolla Fielder', 'Palette Sw',
       'Platz', 'Lancer Evolution', 'Move', 'Gx Series', '5 Series',
       '240 Gd', 'Mira Cocoa', 'Wingroad', 'Belta', 'Atrai Wagon',
```

In [68]:

```
len(Model)
```

Out[68]:

```
304
```

In [10]:

```
car_data_frame=pd.DataFrame(Model)
```

In [11]:

```
conn1=sqlite3.connect('Assignment3.db(1)')
```

In [12]:

```
conn1.commit()
```

In [13]:
```
sales_data_Frame=pd.DataFrame(data)
sales_data_Frame
```

Out[13]:

	Brand	Condition	Fuel	KMs Driven	Model	Price	Registered City	Transaction Type	Year
0	Toyota	Used	Diesel	1.0	Prado	2100000	Karachi	Cash	1997.0
1	Suzuki	Used	Petrol	100000.0	Bolan	380000	Karachi	Cash	2006.0
2	Suzuki	Used	CNG	12345.0	Bolan	340000	Karachi	Cash	1998.0
3	Suzuki	Used	Petrol	94000.0	Alto	535000	Karachi	Cash	2010.0
4	Toyota	Used	Petrol	100000.0	Corolla XLI	1430000	Karachi	Cash	2013.0
...	...	...	...	...	...	...	...	...	...
24964	Toyota	Used	CNG	200000.0	Corolla XE	1070000	Lahore	Cash	2001.0
24965	Daihatsu	New	Petrol	10000.0	Cuore	390000	Karachi	Cash	2004.0
24966	Other Brands	Used	CNG	158715.0	Other	180000	NaN	Cash	2000.0
24967	Suzuki	Used	Petrol	1.0	Alto	470000	Rawalpindi	Cash	2003.0
24968	Toyota	Used	Petrol	48500.0	Corolla GLI	2050000	Lahore	Cash	2017.0

24969 rows × 9 columns

In [14]:
```
clean_dataFrame=sales_data_Frame.dropna()
```

In [15]:
```
clean_dataFrame
```

Out[15]:

	Brand	Condition	Fuel	KMs Driven	Model	Price	Registered City	Transaction Type	Year
0	Toyota	Used	Diesel	1.0	Prado	2100000	Karachi	Cash	1997.0
1	Suzuki	Used	Petrol	100000.0	Bolan	380000	Karachi	Cash	2006.0
2	Suzuki	Used	CNG	12345.0	Bolan	340000	Karachi	Cash	1998.0
3	Suzuki	Used	Petrol	94000.0	Alto	535000	Karachi	Cash	2010.0
4	Toyota	Used	Petrol	100000.0	Corolla XLI	1430000	Karachi	Cash	2013.0
...	...	...	...	...	...	...	...	...	...
24963	Toyota	Used	Petrol	76190.0	Avanza	1175000	Lahore	Cash	2011.0
24964	Toyota	Used	CNG	200000.0	Corolla XE	1070000	Lahore	Cash	2001.0
24965	Daihatsu	New	Petrol	10000.0	Cuore	390000	Karachi	Cash	2004.0
24967	Suzuki	Used	Petrol	1.0	Alto	470000	Rawalpindi	Cash	2003.0
24968	Toyota	Used	Petrol	48500.0	Corolla GLI	2050000	Lahore	Cash	2017.0

20333 rows × 9 columns

In [16]:
```
table_car='''CREATE TABLE IF NOT EXISTS CAR (NAME TEXT PRIMARY KEY)'''
table_sale='''CREATE TABLE IF NOT EXISTS SALE(brand TEXT,condition TEXT, fuel type TEXT, KMs_Driven float,
            model TEXT foriegn key, price INT, registered_city Text,
            tranaction_type TEXT, year INT)'''
```

In [17]:
```
conn1.execute(table_car)
```

Out[17]:
```
<sqlite3.Cursor at 0x1b976871810>
```

In [18]:
```
conn1.execute(table_sale)
```

Out[18]:
```
<sqlite3.Cursor at 0x1b9768719d0>
```

In [19]:

```
#inserting data in car table
for row in car_data_frame.itertuples():
    insert_car=f"INSERT INTO CAR VALUES('{row[1]}')"
    conn1.execute(insert_car)
```

```
---------------------------------------------------------------------------
IntegrityError                            Traceback (most recent call last)
<ipython-input-19-ff5ec707a557> in <module>
      2 for row in car_data_frame.itertuples():
      3     insert_car=f"INSERT INTO CAR VALUES('{row[1]}')"
----> 4     conn1.execute(insert_car)

IntegrityError: UNIQUE constraint failed: CAR.NAME
```

In [20]:

```
car=conn1.execute('select * from car')
```

In [21]:

```
for c in car:
    print(c)
```

```
('Prado',)
('Bolan',)
('Alto',)
('Corolla XLI',)
('Corrolla Altis',)
('Cultus VXL',)
('Civic VTi',)
('Khyber',)
('Liana',)
('Passo',)
('Civic Prosmetic',)
('Civic EXi',)
('Charade',)
('Pajero Mini',)
('Margalla',)
('City IVTEC',)
('Classic',)
('Other',)
('Corolla GLI',)
```

In [22]:
```
for row in clean_dataFrame.itertuples():
    insert_sales=f'''INSERT INTO SALE VALUES('{row[1]}','{row[2]}','{row[3]}',{row[4]},'{row[5]}',{row[6]},
    '{row[7]}','{row[8]}',{row[9]})'''
    conn1.execute(insert_sales)
```

In [23]:
```
sale=conn1.execute('select * from sale')
```

In [24]:
```
for s in sale:
    print(s)
```

```
('Toyota', 'New', 'Petrol', 15500.0, 'Corolla GLI', 1820000, 'Karachi', 'Cash', 2015)
('Toyota', 'Used', 'Diesel', 100000.0, 'Hilux', 2650000, 'Karachi', 'Cash', 2007)
('Daewoo', 'New', 'CNG', 2550.0, 'Racer', 160000, 'Karachi', 'Cash', 1996)
('Suzuki', 'Used', 'Petrol', 100000.0, 'Alto', 380000, 'Karachi', 'Cash', 2001)
('Honda', 'Used', 'Petrol', 80182.0, 'City Vario', 685000, 'Karachi', 'Cash', 2005)
('Classic & Antiques', 'Used', 'Petrol', 123456.0, 'Other', 650000, 'Karachi', 'Cash', 1964)
('Suzuki', 'Used', 'CNG', 50000.0, 'FX', 90000, 'Karachi', 'Cash', 1987)
('Suzuki', 'Used', 'Petrol', 69000.0, 'Cultus VXR', 710000, 'Karachi', 'Cash', 2012)
('Toyota', 'New', 'Petrol', 43946.0, 'Corolla GLI', 1810000, 'Karachi', 'Cash', 2016)
('Mitsubishi', 'Used', 'CNG', 5000.0, 'Lancer Evolution', 85000, 'Karachi', 'Cash', 1986)
('Suzuki', 'Used', 'Petrol', 13800.0, 'Cultus VXR', 1025000, 'Karachi', 'Cash', 2016)
('Honda', 'Used', 'CNG', 79250.0, 'Civic EXi', 775000, 'Karachi', 'Cash', 2005)
('Toyota', 'Used', 'Petrol', 85000.0, 'Corolla GLI', 1150000, 'Karachi', 'Cash', 2010)
('Honda', 'Used', 'Petrol', 75000.0, 'City IDSI', 670000, 'Karachi', 'Cash', 2005)
('Suzuki', 'Used', 'Petrol', 75000.0, 'Alto', 690000, 'Karachi', 'Cash', 2008)
('Honda', 'New', 'Petrol', 21900.0, 'Civic VTi Oriel Prosmatec', 2725000, 'Karachi', 'Cash', 2017)
('Suzuki', 'Used', 'CNG', 100000.0, 'Other', 220000, 'Karachi', 'Cash', 1984)
('Honda', 'Used', 'Petrol', 123456.0, 'Civic VTi Oriel Prosmatec', 1020000, 'Karak', 'Cash', 2007)
('Honda', 'Used', 'Petrol', 60986.0, 'City IDSI', 950000, 'Karachi', 'Cash', 2006)
('Suzuki', 'Used', 'Petrol', 38000.0, 'Mehran VXR', 510000, 'Karachi', 'Cash', 2014)
```

In [25]:
```
year=conn1.execute("select year from sale")
```

In [26]:

```
for years in year:
    print(years)
```

```
(1997,)
(2006,)
(1998,)
(2010,)
(2013,)
(2012,)
(2006,)
(2017,)
(2009,)
(1997,)
(1994,)
(2006,)
(2006,)
(1997,)
(1984,)
(2005,)
(1988,)
(1995,)
(1990,)
```

In [27]:

```
conn1.commit()
```

In [28]:
```
year_data=conn1.execute('''SELECT Model,year from Sale INNER JOIN Car on Car.Name=Sale.Model''')
car_year_data=pd.DataFrame(year_data,columns=['Model','year'])
car_year_data
```

Out[28]:
```
             Model  year
    0        Prado  1997
    1        Bolan  2006
    2        Bolan  1998
    3         Alto  2010
    4  Corolla XLI  2013
  ...          ...   ...
60994       Avanza  2011
60995   Corolla XE  2001
60996        Cuore  2004
60997         Alto  2003
60998  Corolla GLI  2017

60999 rows × 2 columns
```

In [29]:
```
for data in year_data:
    print(data)
```

In [30]:
```
year=car_year_data['year']
model=car_year_data['Model']
```

In [31]:
```
d1=conn1.execute('SELECT Brand,year from sale INNER JOIN Car on Car.Name=sale.Model AND (sale.year=1997)')
data1=pd.DataFrame(d1,columns=['Brand','year'])
data1
```

Out[31]:

	Brand	year
0	Toyota	1997
1	Honda	1997
2	Honda	1997
3	Suzuki	1997
4	BMW	1997
...	...	...
838	Toyota	1997
839	Suzuki	1997
840	Suzuki	1997
841	Suzuki	1997
842	Suzuki	1997

843 rows × 2 columns

In [32]:
```
count=conn1.execute('select count(Brand) from sale where year=1997 ')
count.fetchall()
```

Out[32]:
```
[(843,)]
```

In [33]:
```
for c in count:
    print(c)
```

In [34]:

```
i=1997
sales_count=[]
sales_year=[]
while(i!=2021):
    count=conn1.execute(f'select count(Brand) from sale where year={i} ')
    for c in count:
        sales_count.append(c)
        sales_year.append(i)
    i=i+1
```

In [36]:

```
for i in sales_count:
    print(i)
```

```
(843,)
(915,)
(768,)
(870,)
(945,)
(1083,)
(1680,)
(2352,)
(3114,)
(3885,)
(4065,)
(2994,)
(2043,)
(3003,)
(2952,)
(3114,)
(2901,)
(3150,)
(2688,)
(1881,)
(2148,)
(2523,)
(3,)
(6,)
```

In [37]:

```
sales_year
```

Out[37]:

```
[1997,
 1998,
 1999,
 2000,
 2001,
 2002,
 2003,
 2004,
 2005,
 2006,
 2007,
 2008,
 2009,
 2010,
 2011,
 2012,
 2013,
 2014,
 2015,
 2016,
 2017,
 2018,
 2019,
 2020]
```

In [38]:

```
sales_record=pd.DataFrame(list(zip(sales_year,sales_count)),columns=['Year','Sales'])
```

In [39]:

```
sales_record
```

Out[39]:

```
    Year    Sales
 0  1997   (843,)
 1  1998   (915,)
 2  1999   (768,)
 3  2000   (870,)
 4  2001   (945,)
 5  2002  (1083,)
 6  2003  (1680,)
 7  2004  (2352,)
 8  2005  (3114,)
 9  2006  (3885,)
10  2007  (4065,)
11  2008  (2994,)
12  2009  (2043,)
13  2010  (3003,)
14  2011  (2952,)
15  2012  (3114,)
16  2013  (2901,)
17  2014  (3150,)
18  2015  (2688,)
19  2016  (1881,)
20  2017  (2148,)
21  2018  (2523,)
22  2019     (3,)
23  2020     (6,)
```

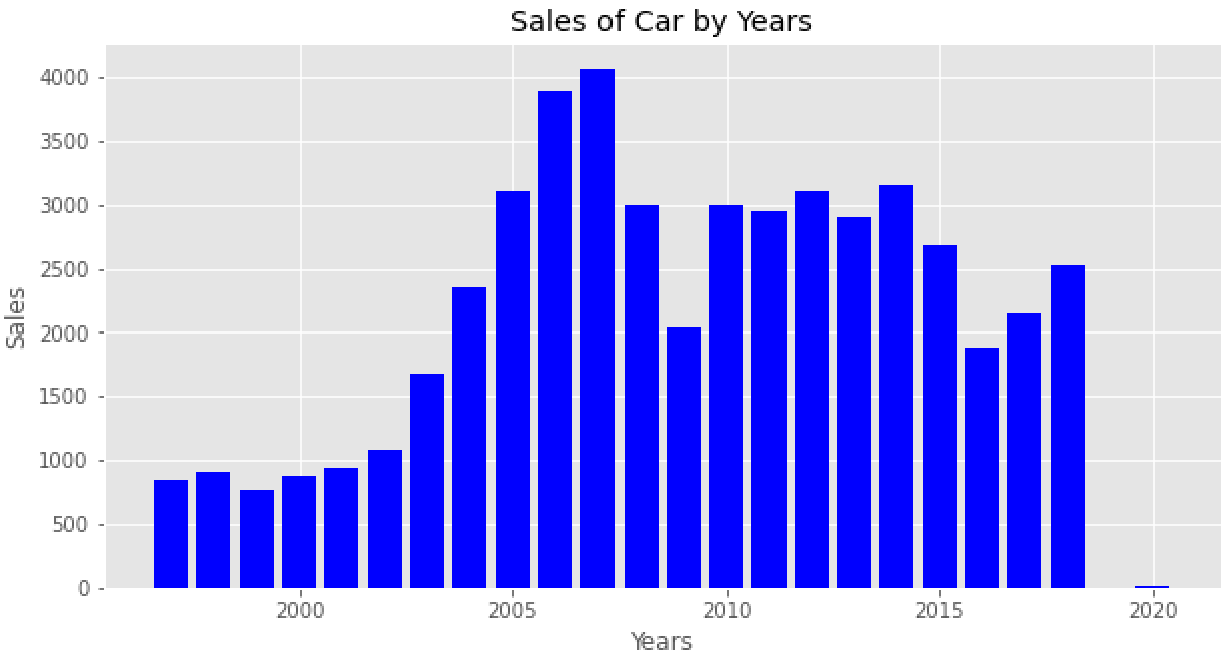
In [40]:
```
sales_record
```

Out[40]:
```
    Year   Sales
0   1997  (843,)
1   1998  (915,)
2   1999  (768,)
3   2000  (870,)
4   2001  (945,)
5   2002 (1083,)
6   2003 (1680,)
7   2004 (2352,)
8   2005 (3114,)
9   2006 (3885,)
10  2007 (4065,)
11  2008 (2994,)
12  2009 (2043,)
13  2010 (3003,)
14  2011 (2952,)
15  2012 (3114,)
16  2013 (2901,)
17  2014 (3150,)
18  2015 (2688,)
19  2016 (1881,)
20  2017 (2148,)
21  2018 (2523,)
22  2019    (3,)
23  2020    (6,)
```

In [41]:

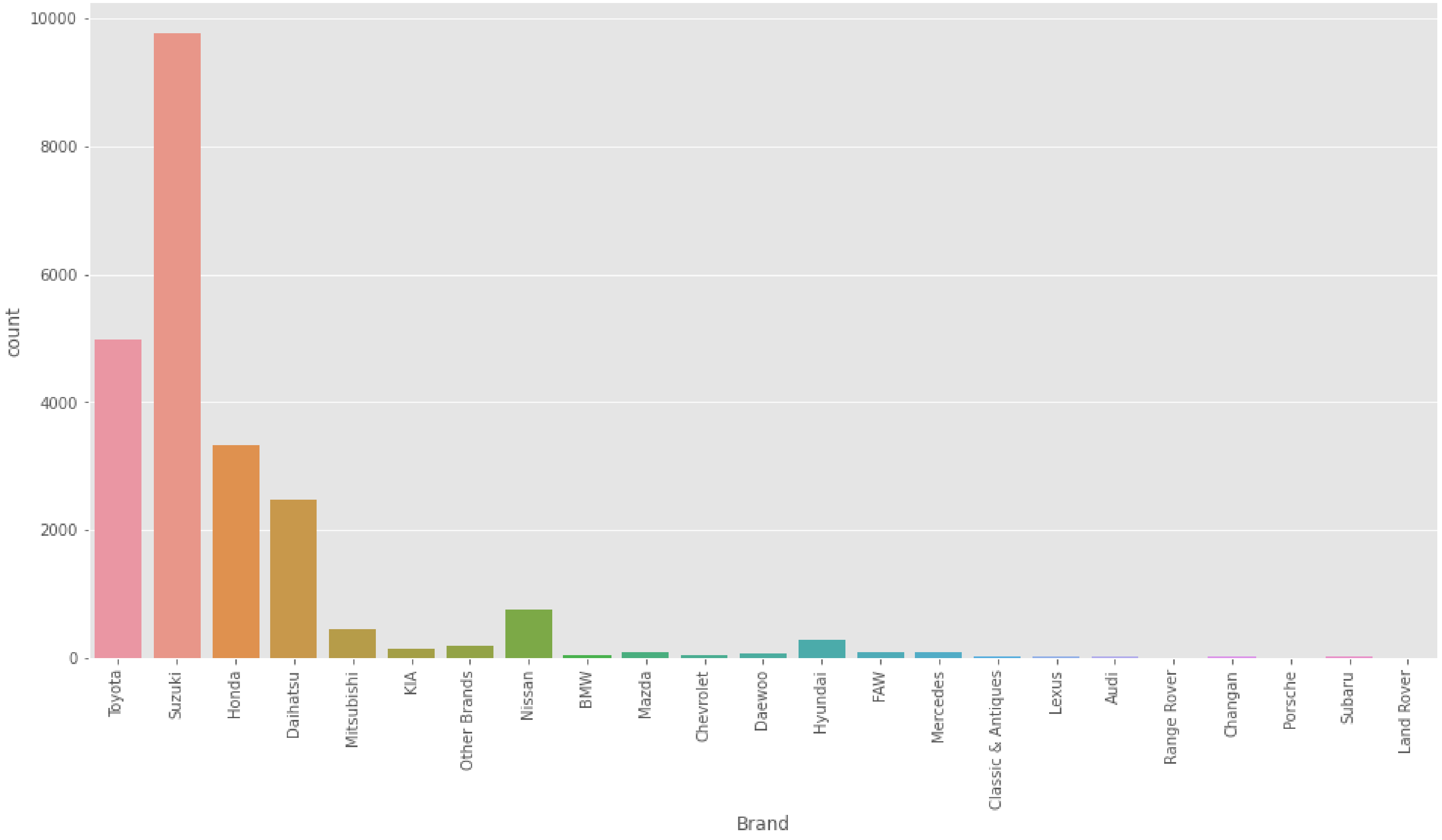
```
x = np.array(sales_record['Sales'])
y = np.array(sales_record['Year'])
plt.style.use('ggplot')
fig=plt.figure(figsize=(10,5))
plt.xlabel('Years')
plt.ylabel('Sales')
plt.title('Sales of Car by Years')
plt.bar(y,height=x,color='blue')
plt.show()
```

```
C:\Users\HP\anaconda3\lib\site-packages\numpy\core\_asarray.py:83: VisibleDeprecationWarning: Creating an ndar
ray from ragged nested sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays with different lengt
hs or shapes) is deprecated. If you meant to do this, you must specify 'dtype=object' when creating the ndarra
y
  return array(a, dtype, copy=False, order=order)
```

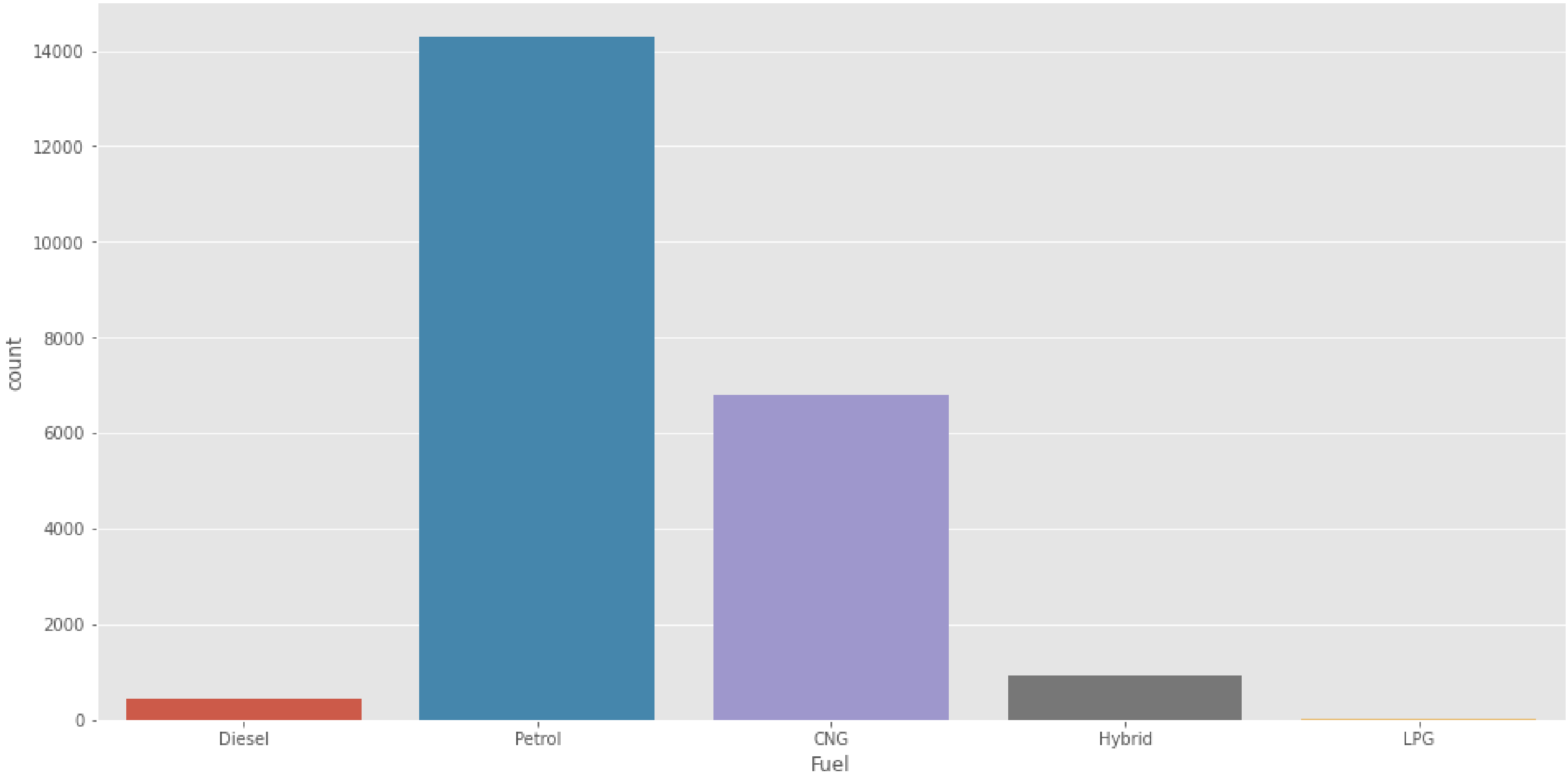

Sales of Car by Years

In [42]:

```
figure=plt.figure(figsize=(16,8))
sns.countplot(x=sales_data_Frame['Brand'],data=data)
plt.xticks(rotation=90)
plt.show()
```

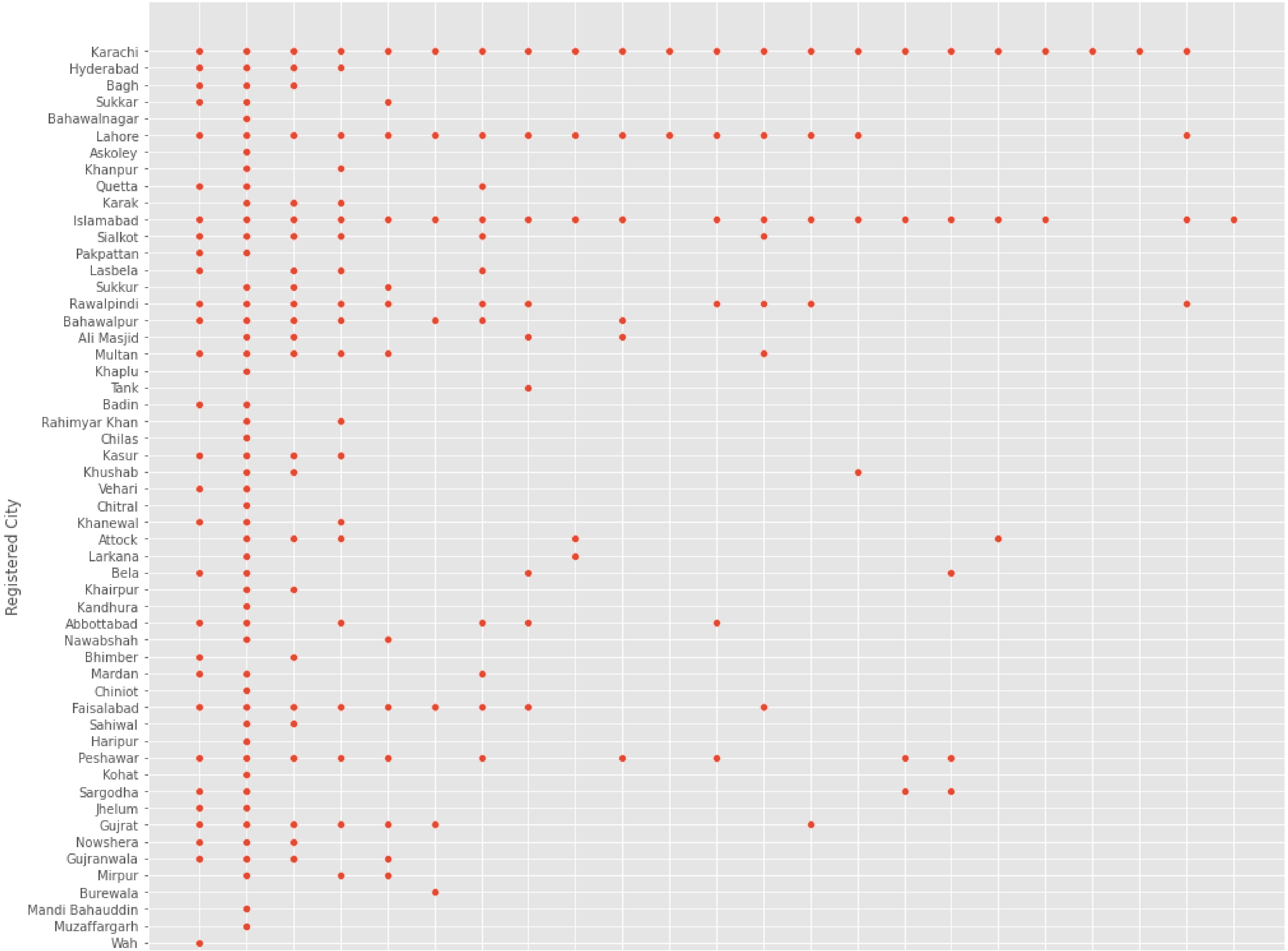


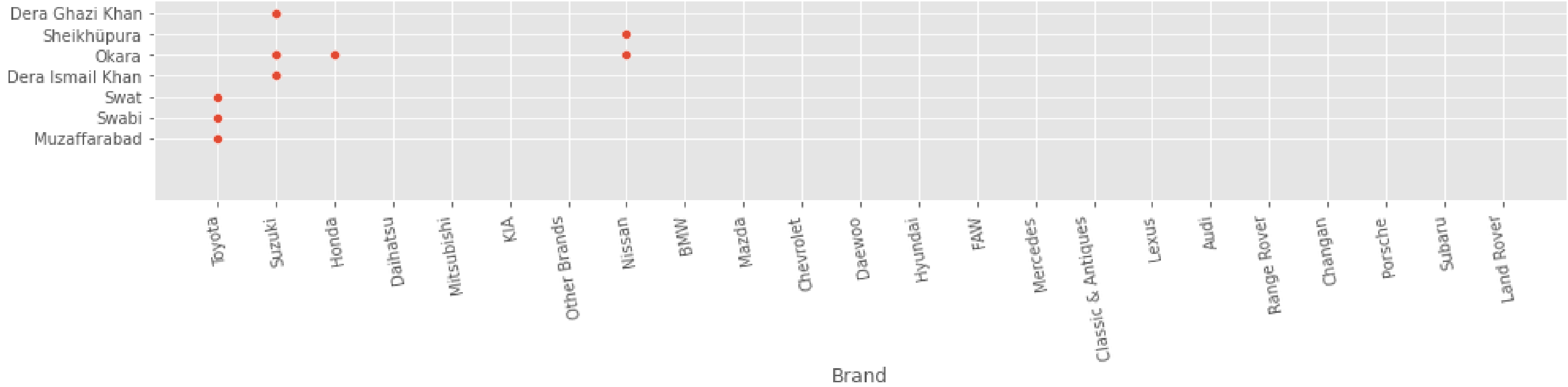
```
In [54]: figure=plt.figure(figsize=(16,8))
         sns.countplot(x=sales_data_Frame['Fuel'],data=data)
         # plt.xticks(rotation=100)
         plt.show()
```



In [55]:
```
figure=plt.figure(figsize=(16,16))
plt.xticks(rotation=100)
sns.scatterplot(data=data,x=sales_data_Frame['Brand'],y=sales_data_Frame['Registered City'])
```

Out[55]:
```
<AxesSubplot:xlabel='Brand', ylabel='Registered City'>
```





Comments:

we have taken the data of the car sales of olx from 1997 to 2021 and through visualization we found how many car soled and of which Brand in a particaular year and we found the average fuel type of car that whcih fuel type is soled more. We have drawn different chart to visualize the results.

```
In [ ]:
```

```
In [ ]:
```