# Car Sales Data Analysis Report


## 🎯 Goal of Project

**The primary goal of this project is to perform an in-depth Exploratory Data Analysis (EDA) on a vehicle sales dataset to uncover key insights, patterns, and trends related to vehicle sales. The analysis will focus on understanding various factors that influence vehicle prices, identifying the distribution of vehicle attributes (such as make, model, year, and interior color), and exploring relationships between these attributes and sales outcomes.**

## 📁 Source of Data : Kaggle

**The vehicle sales data used in this project was sourced from Kaggle, a well-known platform for datasets and data science competitions.**

## 🔄 Work Flow

- step 1 : importing all the related libraries
- step 2 : load the data set
- step 3 : Basic Understanding of Data

- step 4 : check Dtypes of columns
- step 5 : Basic information of Data
- step 6 : Data processing
- step 7 : Handling Missing Value
- step 8 : Feature Engneering Column Extraction
- step 9 : Data Quality Management: Addressing Duplicate Entries and Inconsistencies
- step 10 : Analysis and Insights
- step 11 : Conclusion

# 📚 Importing the required libraries

In [84]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime as dt
```

# 📥 Loading the Dataset

In [85]:
```
df=pd.read_csv("C:\\Users\\Rahul\\Desktop\\car24\\car_prices.csv\\car_prices.csv")
```

# 🔍 Basic Understanding of Data

## Preview Of Data 👀

**Previewing the data allows us to understand the structure of the dataset, including the number of columns, their names, and the type of data stored in each column. This understanding helps you plan your analysis effectively and choose appropriate techniques and tools.**

In [86]:
```
df.sample(5)
```

Out[86]:

| | year | make | model | trim | body | transmission | vin | state | conc |
|---|---|---|---|---|---|---|---|---|---|
| **433961** | 2013 | Ford | C-Max Energi | SEL | Wagon | automatic | 1fadp5cu8dl556866 | mo | |
| **472780** | 2011 | Jeep | Wrangler | Unlimited Sahara | suv | automatic | 1j4ba5h14bl619826 | tn | |
| **4530** | 2004 | Lexus | LS 430 | Base | Sedan | automatic | jthbn36f540164226 | ca | |
| **87094** | 2002 | Mercury | Sable | GS | Sedan | automatic | 1mefm50u62g641449 | ca | |
| **69588** | 2013 | Hyundai | Accent | GLS | Sedan | automatic | kmhct4ae6du346831 | ga | |

# How Big is the Data 📏

In [87]:
```
df.shape
```

Out[87]:
```
(558837, 16)
```

## 🔍 Inference :

- There are total **16 Attributes/column** available in the dataset.
- There are total **558837 Record/rows** available in the dataset.

# Fecthing the columns name 📝

In [88]:
```
for i in df.columns:
    print(i)
```

```
year
make
model
trim
body
transmission
vin
state
condition
odometer
color
interior
seller
mmr
sellingprice
saledate
```

### Columns Name :

**year , make , model , trim , body , transmission , vin , state , condition , odometer , color , interior , seller , mmr,sellingprice and saledate**

## Columns Description :

1. **year** 🗓️: The manufacturing year of the vehicle.
2. **make** 🏭: The manufacturer or brand of the vehicle.
3. **model** 🚗: The specific model name of the vehicle.
4. **trim** ✨: The trim level or variant of the vehicle model.
5. **body** 🚙: The type or style of the vehicle's body (e.g., SUV, sedan).
6. **transmission** ⚙️: The type of transmission (e.g., automatic, manual).
7. **vin** 🆔: The Vehicle Identification Number, a unique code for each vehicle.
8. **state** 📍: The U.S. state where the vehicle is located or registered.
9. **condition** 🛠️: The condition of the vehicle (e.g., new, used).
10. **odometer** 📏: The mileage the vehicle has traveled.
11. **color** 🎨: The exterior color of the vehicle.
12. **interior** 🛋️: The interior color or material of the vehicle.
13. **seller** 🤵: The person or entity selling the vehicle.
14. **mmr** 💲: The Manheim Market Report (MMR) value, an estimate of the vehicle's market value.
15. **sellingprice** 💵: The final sale price of the vehicle.
16. **saledate** 🗓️: The date when the vehicle was sold.

## 📝 Checking Dtypes of columns

In [89]:
```
df1=df.dtypes.to_frame()
```

In [90]:
```
df1.rename(columns={0:"Dtypes"}).T
```

Out[90]:

| | year | make | model | trim | body | transmission | vin | state | condition | odometer | o |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Dtypes** | int64 | object | object | object | object | object | object | object | float64 | float64 | o |

## 🔍 Inference :

- From the above output, we can see that all columns have the appropriate data types except for 'saledate,' which is currently an object. We need to convert it to datetime for better analysis.

# 📝 Basic information of Data

In [91]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 558837 entries, 0 to 558836
Data columns (total 16 columns):
 #   Column        Non-Null Count   Dtype  
---  ------        --------------   -----  
 0   year          558837 non-null  int64  
 1   make          548536 non-null  object 
 2   model         548438 non-null  object 
 3   trim          548186 non-null  object 
 4   body          545642 non-null  object 
 5   transmission  493485 non-null  object 
 6   vin           558833 non-null  object 
 7   state         558837 non-null  object 
 8   condition     547017 non-null  float64
 9   odometer      558743 non-null  float64
 10  color         558088 non-null  object 
 11  interior      558088 non-null  object 
 12  seller        558837 non-null  object 
 13  mmr           558799 non-null  float64
 14  sellingprice  558825 non-null  float64
 15  saledate      558825 non-null  object 
dtypes: float64(4), int64(1), object(11)
memory usage: 68.2+ MB
```

## 🔍 Inference :

- There are a few columns with null or missing values that we also need to handle.

# 🔄 ✨ 📊 Data Preprocessing

**Here's an outline of preprocessing steps which we are going to follow**

- Remove duplicates: Ensure there are no duplicate entries in the dataset.
- Handle missing values: Address missing data points.
- Imputation: Fill missing values with mean, median, mode, or other appropriate values.
- Dropping: Remove rows or columns with excessive missing values.
- Correct errors: Fix any inaccuracies or inconsistencies in the data.
- Convert data types: Ensure all columns have appropriate data types (e.g., dates as datetime objects).
- Create new features: Based on existing data to enhance the model

# 🚩⚠️Handling Missing Value📈🔮

## Checking the missing value of each columns

In [92]:
```
df.isnull().sum().to_frame().rename(columns={0:"Number of missing value"}).T
```

Out[92]:

| | year | make | model | trim | body | transmission | vin | state | condition | odometer |
|---|---|---|---|---|---|---|---|---|---|---|
| **Number of missing value** | 0 | 10301 | 10399 | 10651 | 13195 | 65352 | 4 | 0 | 11820 | 94 |

### 🔍 Inference :

- **Data Completeness**: The **year, state, and seller** columns have no missing values, indicating complete data for these features.
- **Moderate Missing Values**: Columns like **make, model, trim, body, and condition** have a moderate number of missing values, which may require imputation or exclusion depending on their importance in the analysis.
- **Significant Missing Values**: The **transmission** column has a large number of missing values (65,352), which could impact the analysis significantly. This might require either extensive imputation or the possibility of removing this column if it's not critical.
- **Minor Missing Values**: Columns like **vin, odometer, color, interior, mmr, sellingprice, and saledate** have relatively few missing values, making it easier to handle these through simple imputation or by excluding rows with missing data.

## Checking Missing Value Precentage of each Columns

In [93]:
```
print("-------------------------------------------------------------")
print("-------------------------------------------------------------")
print("|||Columns Name|||---------->>>|||Missing Value Percentage|||")
print("-------------------------------------------------------------")
print("-------------------------------------------------------------")
for i in df.columns:
    if df[i].isnull().sum()>0:
        print(i,"------------------------>>>",df[i].isnull().sum()*100/df.sha
```

```
------------------------------------------------------------
------------------------------------------------------------
|||Columns Name|||---------->>>|||Missing Value Percentage|||
------------------------------------------------------------
------------------------------------------------------------
make ------------------------->>> 1.8432924090566658
model ------------------------->>> 1.860828828441925
trim ------------------------->>> 1.905922478289734
body ------------------------->>> 2.361153610086662
transmission ------------------------->>> 11.69428652719845
vin ------------------------->>> 0.000715772219806491
condition ------------------------->>> 2.1151069095281807
odometer ------------------------->>> 0.016820647165452538
color ------------------------->>> 0.13402834815876544
interior ------------------------->>> 0.13402834815876544
mmr ------------------------->>> 0.006799836088161665
sellingprice ------------------------->>> 0.002147316659419473
saledate ------------------------->>> 0.002147316659419473
```

### 🔍 Inference :

- All columns with null values have a missing rate below 5%, except for the transmission column, which has 11% missing data. Given the richness of the dataset, the impact of these missing values on overall insights is very minimal.
- I've decided to drop rows with null values in all columns except for transmission, as the low percentage of missing data in these columns won't significantly affect the analysis.
- For the transmission column, we will handle the missing values separately, using domain knowledge to fill them in accurately, ensuring the integrity of the dataset.

# Taking transmission Columns

In [94]:
```
df['transmission'].nunique()
```

Out[94]:
```
4
```

In [95]:
```
df['transmission'].unique()
```

Out[95]:
```
array(['automatic', nan, 'manual', 'Sedan', 'sedan'], dtype=object)
```

- The **transmission** column currently shows **4 unique values**, likely due to inconsistencies in the data, which we will address later. Based on domain knowledge, there should only be **2 unique values: automatic and manual**. Since the transmission data is categorical, we will handle the null values by filling them with the most frequent value, using the mode().

In [96]:
```
df['transmission'].mode()
```

Out[96]:
```
0    automatic
Name: transmission, dtype: object
```

In [97]:
```
df['transmission'].fillna(df['transmission'].mode()[0],inplace=True)
```

In [98]:
```
df['transmission'].isnull().sum()
```

Out[98]:
```
0
```

# Taking All Null Values Columns Except transmission

In [99]:
```
df.dropna(axis=0,inplace=True)
```

# Verifying the Null Value replacements

In [100…
```
df.isnull().sum()
```

Out[100]:
```
year            0
make            0
model           0
trim            0
body            0
transmission    0
vin             0
state           0
condition       0
odometer        0
color           0
interior        0
seller          0
mmr             0
sellingprice    0
saledate        0
dtype: int64
```

In [101…
```
df.isnull().sum().sum()
```

Out[101]:
```
0
```

## 🔍 Inference :

- Now that we have addressed the null values in the dataset, our data is complete with no missing values.


# Checking the duplicate entries

In [102…
```
df.duplicated().sum()
```

Out[102]:
```
0
```

## 🔍 Inference :

- Their is no duplicates values in the data

# Detecting the inconsistency of the data

In [103…

```
for i in df.columns:
    print(i)
    print( i ,"has",df[i].nunique(),"unique values")
    print("----------------------------------------Unique Value------------
    print("------------------------------------------------------------------
    print(df[i].unique())
    #for j in df[i]:
        #print(j)
    print("==================================================================
    print("==================================================================
```

```
year
year has 26 unique values
----------------------------------------Unique Value----------------------
------------------------
------------------------------------------------------------------------------
------------------------
[2015 2014 2013 2012 2011 2010 2009 2008 2007 2006 2005 2004 2003 2002
 2001 2000 1999 1998 1996 1995 1997 1994 1993 1992 1991 1990]
==============================================================================
==========================
==============================================================================
==========================
make
make has 53 unique values
----------------------------------------Unique Value----------------------
------------------------
------------------------------------------------------------------------------
------------------------
['Kia' 'BMW' 'Volvo' 'Nissan' 'Chevrolet' 'Audi' 'Ford' 'Buick' 'Cadillac'
 'Acura' 'Lexus' 'Hyundai' 'Infiniti' 'Jeep' 'Mercedes-Benz' 'Mitsubishi'
 'Mazda' 'MINI' 'Land Rover' 'Lincoln' 'Jaguar' 'Volkswagen' 'Toyota'
 'Subaru' 'Scion' 'Porsche' 'Dodge' 'FIAT' 'Chrysler' 'Ferrari' 'Honda'
 'GMC' 'Ram' 'smart' 'Bentley' 'Pontiac' 'Saturn' 'Maserati' 'Mercury'
 'HUMMER' 'Saab' 'Suzuki' 'Oldsmobile' 'Geo' 'Rolls-Royce' 'Isuzu'
 'Plymouth' 'Tesla' 'Aston Martin' 'Fisker' 'Daewoo' 'Lamborghini' 'Lotus']
==============================================================================
==========================
==============================================================================
==========================
model
model has 772 unique values
----------------------------------------Unique Value----------------------
------------------------
------------------------------------------------------------------------------
------------------------
['Sorento' '3 Series' 'S60' '6 Series Gran Coupe' 'Altima' 'M5' 'Cruze'
 'A4' 'Camaro' 'A6' 'Optima' 'Fusion' 'Q5' '6 Series' 'Impala' '5 Series'
 'A3' 'XC70' 'SQ5' 'S5' 'Verano' 'Suburban' 'ELR' 'V60' 'X6' 'ILX' 'K900'
 'Malibu' 'RX 350' 'Versa' 'Elantra' 'Versa Note' 'A8' 'X1' 'Enclave'
 'TTS' '4 Series' 'Silverado 2500HD' 'MDX' 'Silverado 1500' 'SRX' 'X5'
 'G Coupe' 'G Sedan' 'FX' 'Santa Fe' 'Genesis' 'Equus' 'Sonata'
 'Sonata Hybrid' 'Accent' 'Veloster' 'Elantra Coupe' 'Azera' 'Tucson'
 'Genesis Coupe' 'Wrangler' 'S-Class' 'GS 350' 'Outlander' 'C-Class'
 'Mazda2' 'Rio' 'M' '370Z' 'Soul' 'Outlander Sport' 'SLK-Class' 'ES 350'
 'E-Class' 'Mazda3' 'Cooper Clubman' 'Cooper' 'CX-9' 'Forte' 'Compass'
 'JX' 'LR4' 'Mazda5' 'Range Rover Evoque' 'LS 460' 'GLK-Class' 'Sportage'
 'Grand Cherokee' 'MKX' 'XF' 'GL-Class' 'M-Class' 'Cooper Countryman'
 'Lancer' 'Range Rover Sport' 'Passat' 'Corolla' 'XC60' 'Sienna' 'Juke'
 'Yaris' 'Sentra' 'NV' 'CC' 'Leaf' 'Camry' 'Tacoma' 'Jetta' 'Impreza WRX'
 'FJ Cruiser' 'Beetle' 'Rogue' 'Avalon' 'FR-S' 'NV200' 'Quest' 'Tundra'
 'Maxima' 'Cayenne' '911' 'Xterra' 'Prius' 'S80' 'Frontier' 'Boxster'
 'Camry Hybrid' 'xB' 'Cube' 'Jetta SportWagen' '4Runner' 'Sequoia'
 'Legacy' 'Armada' 'Venza' 'Murano' 'Pathfinder' 'Panamera' 'Forester'
 'Highlander' 'TSX' '7 Series' '1 Series' 'TL' 'S4' 'A7' 'A5' 'RDX' 'M3'
 'Cooper Coupe' 'ZDX' 'R8' 'X3' 'Avenger' 'E-Series Wagon' 'Escape' 'Edge'
 'Focus' 'Z4' 'Traverse' 'F-350 Super Duty' 'Fiesta' '500' '200' 'Journey'
 'Charger' 'Flex' 'Equinox' '300' 'F-150' 'Explorer' 'Captiva Sport'
 'Escalade' 'Grand Caravan' 'CTS Coupe' 'E-Series Van' 'Volt'
 'Express Cargo' 'X5 M' 'Expedition' 'Colorado' 'Express' 'California'
 'Escalade ESV' 'Town and Country' 'Sonic' 'Accord' 'CR-V' 'Mustang'
 'Civic' 'Fit' 'Pilot' 'Odyssey' 'Crosstour' 'Transit Connect' 'Terrain'
 'Taurus' 'G Convertible' 'Yukon' 'Veracruz' 'XJ' 'Liberty' 'IS 250' 'XK'
 'QX' 'CT 200h' 'Mazda6' 'MKZ' 'Navigator' 'Range Rover' 'SL-Class'
 'Sedona' 'IS 350' 'Patriot' '1500' 'Impreza' 'GT-R' 'tC' '2500' 'Galant'
 'fortwo' 'GLI' '5 Series Gran Turismo' 'XC90' 'Tiguan' 'GTI' 'Q7'
 'Highlander Hybrid' 'RAV4' 'Prius Plug-in' 'CR-Z' 'EX' 'Sierra 1500'
 'LaCrosse' 'HHR' 'Accord Crosstour' 'CTS' 'Nitro' 'Tahoe' 'Challenger'
 'CTS-V' 'Escape Hybrid' 'X6 M' 'Ranger' 'Insight' 'Fusion Hybrid'
 'CTS-V Coupe' 'F-250 Super Duty' 'Acadia' 'Impala Limited' 'Dart' 'M37'
```

'Sprinter' 'Town Car' 'CX-7' 'RX 450h' 'MKT' 'QX56' 'Aveo' 'Outback'
'Caliber' 'Routan' 'Sebring' 'Corvette' 'Continental GT Speed' 'malibu'
'Land Cruiser' 'V50' 'Commander' 'Altima Hybrid' 'G37 Convertible'
'New Beetle' 'Golf' 'LR2' 'Lancer Sportback' 'G5' 'Yukon XL'
'Escalade Hybrid' 'Avalanche' 'Spectra' 'Rondo' 'Borrego' 'G-Class' 'MKS'
'CLK-Class' 'Tahoe Hybrid' 'Econoline Cargo' 'Econoline Wagon'
'CLS-Class' 'STS' 'Ridgeline' 'F-450 Super Duty' 'Magnum' 'Durango' 'S40'
'Malibu Classic' 'TT' 'Explorer Sport Trac' 'Ram Pickup 1500'
'PT Cruiser' 'Pacifica' 'S6' 'Rabbit' 'Sierra 2500HD' 'C30' 'VUE'
'GranTurismo' 'G6' 'Grand Prix' '350Z' 'Raider' 'Mazdaspeed Mazda3'
'Solstice' 'Milan' 'GX 470' 'RX 400h' 'Titan' 'H3' 'CL-Class' 'Outlook'
'G37' 'IS F' 'Touareg 2' 'Lancer Evolution' 'G35' 'xD' 'XJ-Series'
'Cobalt' 'H2' 'DTS' 'M45' 'Uplander' 'GS 450h' 'Silverado 1500 Classic'
'Rendezvous' 'Monte Carlo' 'FX35' 'ION' 'R-Class' 'Eclipse' 'Aura' 'RSX'
'Mariner' 'Five Hundred' 'Envoy XL' 'S-Type' 'Continental Flying Spur'
'S2000' 'FX45' 'Element' 'Cayman S' 'Mark LT' 'GS 300' 'Camry Solara'
'Touareg' 'allroad quattro' '9-3' '500L' 'Ram Pickup 3500'
'Sprinter Cargo' 'DeVille' 'H2 SUT' 'TrailBlazer' 'Canyon' 'Dakota'
'Continental GT' 'Neon' 'ES 330' 'Freestyle' 'Stratus' 'Q45' 'Freestar'
'XLR' 'Aviator' 'MPV' 'LS 430' 'RX 330' 'SC 430' 'Envoy XUV' 'Monterey'
'Excursion' 'Mountaineer' 'Malibu Maxx' 'Celica' 'Grand Am' 'Matrix'
'Envoy' 'Endeavor' 'Concorde' 'Marauder' 'C70' 'QX4' 'LS' 'Blazer'
'LeSabre' 'Montero' 'ES 300' 'Thunderbird' 'Century' 'Cavalier' 'Venture'
'S-10' 'XL-7' 'Silverado 1500HD' 'Savana Cargo' 'X-Type' 'Sonoma'
'IS 300' 'Protege5' 'Alero' 'Grand Vitara' 'RX 300' 'L-Series' 'Intrigue'
'XC' 'S-Series' 'Discovery Series II' 'Continental' 'Seville'
'Park Avenue' 'Millenia' 'V70' 'Z3' 'LHS' 'Regal' 'G20' 'Eldorado'
'Intrepid' 'Contour' 'S90' 'I30' 'Sunfire' 'Escort' '200SX'
'Mustang SVT Cobra' 'Grand Marquis' 'Prizm' 'LS 400' 'Cutlass Ciera'
'Santa Fe Sport' 'Cherokee' 'Cadenza' 'Q50' 'Elantra GT' 'F-TYPE'
'Shelby GT500' 'QX70' 'QX60' 'Q60 Convertible' 'Cooper Roadster' 'CX-5'
'Cooper Paceman' 'MX-5 Miata' 'Rogue Select' 'Cayman' 'CLA-Class'
'allroad' 'ATS' 'Prius v' 'Continental GTC' 'XV Crosstrek' '3500'
'C-Max Energi' 'C-Max Hybrid' 'Spark' 'RS 7' 'GX 460' 'CTS Wagon'
'SLS AMG' 'Ram Pickup 2500' 'Eclipse Spyder' 'Vibe' 'Eos' 'Caravan' 'SSR'
'Safari Cargo' 'Amanti' 'Tribute' 'Sable' 'Silverado 3500' 'Phaeton'
'R32' '9-5' 'Truck' 'Tercel' 'Roadmaster' 'SC 400' 'LX 570' 'QX80' 'RS 5'
'Jetta GLI' 'ES 300h' 'M4' 'SX4' 'iQ' 'Kizashi' 'C/V Cargo Van' 'Prius c'
'Lucerne' 'Escalade EXT' 'Silverado 3500HD' 'Crown Victoria'
'Sierra 3500HD' 'M56' 'IS 250 C' 'RL' 'ActiveHybrid X6' 'M35'
'Elantra Touring' 'G37 Coupe' 'HS 250h' 'Mazdaspeed3' 'Milan Hybrid'
'Ghost' 'Silverado 1500 Hybrid' 'Aspen' 'Taurus X' '6' 'Mariner Hybrid'
'G8' 'Torrent' 'VUE Hybrid' 'G3' '9-7X' 'Expedition EL' 'Yukon Hybrid'
'Tiburon' 'LR3' 'Navigator L' 'Astra' 'Tribeca' 'XL7' 'Forenza' 'Sky'
'M6' 'S8' 'Terraza' 'Silverado 2500HD Classic' 'Focus ST' 'Fusion Energi'
'Entourage' 'B-Series Truck' 'Mazdaspeed Mazda6' 'Montego' 'B9 Tribeca'
'Rainier' 'Crossfire' 'TrailBlazer EXT' 'Sierra 1500HD' 'Savana' 'Zephyr'
'Quattroporte' 'Relay' 'Verona' 'Classic' 'XG350' 'Q60 Coupe' 'Q70'
'QX50' 'GTO' 'Bonneville' 'Baja' 'ECHO' 'Aerio' 'Silverado 2500' '300M'
'Tracker' 'F-150 Heritage' 'accord' 'Rodeo' 'Ascender' 'Freelander'
'L300' 'MR2 Spyder' 'RS 6' 'Explorer Sport' 'Windstar' 'Axiom'
'Montero Sport' 'Diamante' 'Blackwood' 'Protege' 'Firebird' 'Silhouette'
'Aurora' 'CL' 'Astro' '626' 'GS 430' 'Prowler' 'Aztek' 'Cabrio' 'Integra'
'Tahoe Limited/Z71' 'Cirrus' 'Sephia' 'LX 470' 'Mirage' 'S70' 'Catera'
'Passport' 'Cougar' 'C/K 1500 Series' 'Discovery' 'XK-Series' 'Bravada'
'B-Series Pickup' 'SC 300' 'Lumina' 'Pickup' '500-Class'
'Murano CrossCabriolet' 'NV Cargo' 'Jetta Hybrid' 'S7' 'Encore' 'XTS'
'Black Diamond Avalanche' 'RLX' '2 Series' 'M6 Gran Coupe'
'C/V Tradesman' 'BRZ' 'Model S' 'Beetle Convertible' 'Golf R'
'TSX Sport Wagon' 'Corvette Stingray' 'SS' 'Mazdaspeed 3' 'i-MiEV' 'F430'
'I35' 'Ghibli' '960' 'Cutlass Supreme' 'IS 350 C' 'EX35' 'Reno'
'Astro Cargo' 'Montana' 'Mazdaspeed MX-5 Miata' 'V40' 'Ram Cargo'
'Safari' 'envoy' 'Prelude' 'Eighty-Eight' 'Legend' '850' 'J30'
'Promaster Cargo Van' 'Jimmy' 'Malibu Hybrid' 'Sierra 1500 Classic'
'Rodeo Sport' 'Cutlass' 'Spirit' '300-Class' 'Accord Hybrid'
'3 Series Gran Turismo' 'Viper' 'Sierra 2500' 'Trooper' 'Yukon Denali'
'Riviera' 'Avalon Hybrid' 'RX-8' 'V8 Vantage' '3' 'Equator'
'Sierra 2500HD Classic' '9-2X' 'xA' 'XG300' 'C/K 3500 Series' 'Mark VIII'
'GranTurismo Convertible' 'Vitara' 'Voyager' 'GS 400' 'Caprice'

```
 'Eighty-Eight Royale' 'ActiveHybrid 7' 'GS 460' 'Tribute Hybrid'
 'Aura Hybrid' '300ZX' 'Golf GTI' 'i-Series' 'MKZ Hybrid' 'Macan' 'FX50'
 'STS-V' 'Windstar Cargo' 'CTS-V Wagon' 'Karma' 'Z4 M' 'Villager' 'Coupe'
 'Ram Van' 'Tempo' 'Tracer' 'CV Tradesman' 'DB9' 'C/K 2500 Series' 'i8'
 'Rapide' 'Nubira' 'Mazdaspeed Protege' 'Montana SV6' 'Corsica'
 'NV Passenger' 'Spyder' 'LS 600h L' '400-Class' 'H3T' 'LX 450' 'WRX'
 'Esteem' 'Silverado 3500 Classic' 'GranSport' '500e'
 'Continental Supersports' 'EuroVan' 'Sierra 3500' 'Mystique'
 'F-150 SVT Lightning' 'E-150' 'Breeze' '190-Class' 'MKC' 'Aspire' '940'
 'Gallardo' 'Amigo' 'Continental Flying Spur Speed' '3000GT' 'TT RS'
 'B-Series' 'ActiveHybrid 5' 'Sierra 1500 Hybrid' 'ML55 AMG' 'S-10 Blazer'
 'RS 4' 'T100' 'Continental GTC Speed' 'mdx' 'Transit Van' 'F-250'
 'Sidekick' 'E-250' '8 Series' '420-Class' 'E-350' 'Achieva'
 'B-Class Electric Drive' 'Fleetwood' 'Paseo' 'Civic del Sol' 'Exige' 'X4'
 'Spark EV' 'Transit Wagon' 'H1' 'SLS AMG GT' 'Flying Spur' 'Metro'
 'Grand Cherokee SRT' 'RC F' 'Q3' '4 Series Gran Coupe' 'RC 350' '360'
 'GLA-Class' 'TLX' '458 Italia']
================================================================================
==========================
================================================================================
==========================
trim
trim has 1510 unique values
----------------------------------------Unique Value----------------------
------------------------
--------------------------------------------------------------------------------
------------------------
['LX' '328i SULEV' 'T5' ... 'pure' 'EWB' 'Power Wagon']
================================================================================
==========================
================================================================================
==========================
body
body has 86 unique values
----------------------------------------Unique Value----------------------
------------------------
--------------------------------------------------------------------------------
------------------------
['SUV' 'Sedan' 'Convertible' 'Coupe' 'Wagon' 'Hatchback' 'Crew Cab'
 'G Coupe' 'G Sedan' 'Elantra Coupe' 'Genesis Coupe' 'Minivan' 'Van'
 'Double Cab' 'CrewMax Cab' 'Access Cab' 'King Cab' 'SuperCrew'
 'CTS Coupe' 'Extended Cab' 'E-Series Van' 'SuperCab' 'Regular Cab'
 'G Convertible' 'Koup' 'Quad Cab' 'CTS-V Coupe' 'sedan' 'G37 Convertible'
 'Club Cab' 'Xtracab' 'Q60 Convertible' 'CTS Wagon' 'G37 Coupe' 'Mega Cab'
 'Cab Plus 4' 'Q60 Coupe' 'Beetle Convertible' 'TSX Sport Wagon'
 'Promaster Cargo Van' 'Cab Plus' 'GranTurismo Convertible' 'CTS-V Wagon'
 'Ram Van' 'convertible' 'minivan' 'suv' 'Transit Van' 'van' 'regular-cab'
 'g sedan' 'g coupe' 'hatchback' 'king cab' 'supercrew' 'g convertible'
 'coupe' 'crew cab' 'wagon' 'double cab' 'e-series van' 'regular cab'
 'quad cab' 'g37 convertible' 'supercab' 'extended cab' 'crewmax cab'
 'genesis coupe' 'access cab' 'mega cab' 'xtracab' 'beetle convertible'
 'cts coupe' 'koup' 'club cab' 'elantra coupe' 'q60 coupe' 'cts-v coupe'
 'transit van' 'granturismo convertible' 'tsx sport wagon'
 'promaster cargo van' 'q60 convertible' 'g37 coupe' 'cab plus 4'
 'cts wagon']
================================================================================
==========================
================================================================================
==========================
transmission
transmission has 2 unique values
----------------------------------------Unique Value----------------------
------------------------
--------------------------------------------------------------------------------
------------------------
['automatic' 'manual']
================================================================================
==========================
================================================================================
```

```
==========================
vin
vin has 525638 unique values
----------------------------------------Unique Value----------------------
------------------------
------------------------------------------------------------------------------
------------------------
['5xyktca69fg566472' '5xyktca69fg561319' 'wba3c1c51ek116351' ...
 '5uxzw0c58cl668465' '1n4al3ap0fc216050' '1ftfw1et2eke87277']
==============================================================================
==========================
==============================================================================
==========================
state
state has 38 unique values
----------------------------------------Unique Value----------------------
------------------------
------------------------------------------------------------------------------
------------------------
['ca' 'tx' 'pa' 'mn' 'az' 'wi' 'tn' 'md' 'fl' 'ne' 'oh' 'mi' 'nj' 'ga'
 'va' 'sc' 'in' 'il' 'co' 'ut' 'mo' 'nv' 'ma' 'pr' 'nc' 'ny' 'or' 'la'
 'wa' 'hi' 'qc' 'ab' 'on' 'ok' 'ms' 'nm' 'al' 'ns']
==============================================================================
==========================
==============================================================================
==========================
condition
condition has 41 unique values
----------------------------------------Unique Value----------------------
------------------------
------------------------------------------------------------------------------
------------------------
[ 5. 45. 41. 43.  1. 34.  2. 42.  3. 48. 49. 17. 19. 29. 38. 44. 47. 32.
  4. 25. 37. 39. 31. 28. 46. 36. 35. 26. 21. 22. 27. 24. 33. 23. 15. 16.
 18. 12. 14. 11. 13.]
==============================================================================
==========================
==============================================================================
==========================
odometer
odometer has 166557 unique values
----------------------------------------Unique Value----------------------
------------------------
------------------------------------------------------------------------------
------------------------
[ 16639.   9393.   1331. ... 204835. 111069. 262065.]
==============================================================================
==========================
==============================================================================
==========================
color
color has 20 unique values
----------------------------------------Unique Value----------------------
------------------------
------------------------------------------------------------------------------
------------------------
['white' 'gray' 'black' 'red' 'silver' 'brown' 'beige' 'blue' 'purple'
 'burgundy' '—' 'gold' 'yellow' 'green' 'charcoal' 'orange' 'off-white'
 'turquoise' 'pink' 'lime']
==============================================================================
==========================
==============================================================================
==========================
interior
interior has 17 unique values
----------------------------------------Unique Value----------------------
------------------------
------------------------------------------------------------------------------
------------------------
```

```
['black' 'beige' 'tan' 'brown' 'gray' '—' 'burgundy' 'white' 'silver'
 'off-white' 'red' 'yellow' 'green' 'purple' 'blue' 'orange' 'gold']
================================================================================
==========================
================================================================================
==========================
seller
seller has 12735 unique values
----------------------------------------Unique Value----------------------
------------------------
--------------------------------------------------------------------------------
------------------------
['kia motors america  inc' 'financial services remarketing (lease)'
 'volvo na rep/world omni' ... 'maserati north america inc'
 'alternative financial group inc' 'i -5 uhlmann rv']
================================================================================
==========================
================================================================================
==========================
mmr
mmr has 1101 unique values
----------------------------------------Unique Value----------------------
------------------------
--------------------------------------------------------------------------------
------------------------
[ 20500.  20800.  31900. ... 182000. 119000. 164000.]
================================================================================
==========================
================================================================================
==========================
sellingprice
sellingprice has 1852 unique values
----------------------------------------Unique Value----------------------
------------------------
--------------------------------------------------------------------------------
------------------------
[ 21500.  30000.  27750. ... 169000.  16550.  27840.]
================================================================================
==========================
================================================================================
==========================
saledate
saledate has 3687 unique values
----------------------------------------Unique Value----------------------
------------------------
--------------------------------------------------------------------------------
------------------------
['Tue Dec 16 2014 12:30:00 GMT-0800 (PST)'
 'Thu Jan 15 2015 04:30:00 GMT-0800 (PST)'
 'Thu Jan 29 2015 04:30:00 GMT-0800 (PST)' ...
 'Thu Jul 02 2015 13:20:00 GMT-0700 (PDT)'
 'Tue Jul 07 2015 09:50:00 GMT-0700 (PDT)'
 'Wed Jul 08 2015 09:45:00 GMT-0700 (PDT)']
================================================================================
==========================
================================================================================
==========================
```

# Taking Color Columns and Interior columns

**We've identified that the Color and Interior columns contain a '-' value, which doesn't represent a valid color. Since the actual color is unknown, I'll replace this placeholder with 'MultiColor'. This custom value will indicate uncertainty while still providing meaningful context to the data.**

In [104…
```
df['color'].replace('–', 'multicolor', inplace=True)
df['interior'].replace('–', 'multicolor', inplace=True)
```

In [105…
```
df['color'].unique()
```

Out[105]:
```
array(['white', 'gray', 'black', 'red', 'silver', 'brown', 'beige',
       'blue', 'purple', 'burgundy', 'multicolor', 'gold', 'yellow',
       'green', 'charcoal', 'orange', 'off-white', 'turquoise', 'pink',
       'lime'], dtype=object)
```

In [106…
```
df['interior'].unique()
```

Out[106]:
```
array(['black', 'beige', 'tan', 'brown', 'gray', 'multicolor', 'burgundy',
       'white', 'silver', 'off-white', 'red', 'yellow', 'green', 'purple',
       'blue', 'orange', 'gold'], dtype=object)
```

# Taking State columns

In [107…
```
df['state'].unique()
```

Out[107]:
```
array(['ca', 'tx', 'pa', 'mn', 'az', 'wi', 'tn', 'md', 'fl', 'ne', 'oh',
       'mi', 'nj', 'ga', 'va', 'sc', 'in', 'il', 'co', 'ut', 'mo', 'nv',
       'ma', 'pr', 'nc', 'ny', 'or', 'la', 'wa', 'hi', 'qc', 'ab', 'on',
       'ok', 'ms', 'nm', 'al', 'ns'], dtype=object)
```

Here is a list of U.S. states and territories corresponding to the abbreviations provided in dataset:

1. **CA** - California
2. **TX** - Texas
3. **PA** - Pennsylvania
4. **MN** - Minnesota
5. **AZ** - Arizona
6. **WI** - Wisconsin
7. **TN** - Tennessee
8. **MD** - Maryland
9. **FL** - Florida
10. **NE** - Nebraska
11. **OH** - Ohio
12. **MI** - Michigan
13. **NJ** - New Jersey
14. **GA** - Georgia
15. **VA** - Virginia
16. **SC** - South Carolina
17. **IN** - Indiana
18. **IL** - Illinois
19. **CO** - Colorado
20. **UT** - Utah
21. **MO** - Missouri
22. **NV** - Nevada
23. **MA** - Massachusetts
24. **PR** - Puerto Rico (U.S. territory)
25. **NC** - North Carolina
26. **NY** - New York
27. **OR** - Oregon

28. **LA** - Louisiana
29. **WA** - Washington
30. **HI** - Hawaii
31. **QC** - Quebec (Province in Canada)
32. **AB** - Alberta (Province in Canada)
33. **ON** - Ontario (Province in Canada)
34. **OK** - Oklahoma
35. **MS** - Mississippi
36. **NM** - New Mexico
37. **AL** - Alabama
38. **NS** - Nova Scotia (Province in Canada)

# Taking condition Columns

In [108…
```
df['condition'].unique()
```

Out[108]:
```
array([ 5., 45., 41., 43.,  1., 34.,  2., 42.,  3., 48., 49., 17., 19.,
       29., 38., 44., 47., 32.,  4., 25., 37., 39., 31., 28., 46., 36.,
       35., 26., 21., 22., 27., 24., 33., 23., 15., 16., 18., 12., 14.,
       11., 13.])
```

In [109…
```
df['condition'].dtypes
```

Out[109]:
```
dtype('float64')
```

- The values in the **condition** column are currently not in float format. We need to change the data type from float to integer.

In [110…
```
df['condition']=df['condition'].astype(int)
```

Verifying the results

In [111…
```
df['condition'].dtypes
```

Out[111]:
```
dtype('int32')
```

In [112…
```
df['condition'].unique()
```

Out[112]:
```
array([ 5, 45, 41, 43,  1, 34,  2, 42,  3, 48, 49, 17, 19, 29, 38, 44, 47,
       32,  4, 25, 37, 39, 31, 28, 46, 36, 35, 26, 21, 22, 27, 24, 33, 23,
       15, 16, 18, 12, 14, 11, 13])
```

# Taking Saledate

In [113…
```
df['saledate'].info()
```

```
<class 'pandas.core.series.Series'>
Index: 533648 entries, 0 to 558836
Series name: saledate
Non-Null Count   Dtype
--------------   -----
533648 non-null  object
dtypes: object(1)
memory usage: 8.1+ MB
```

**Since the data type of the 'saledate' column is currently object, we will convert it to datetime.**

In [114…
```
df["saledate"]=pd.to_datetime(df["saledate"],utc=True)
```

```
C:\Users\Rahul\AppData\Local\Temp\ipykernel_13880\4275068227.py:1: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.
  df["saledate"]=pd.to_datetime(df["saledate"],utc=True)
```

In [115…
```
df["saledate"].dtypes
```

Out[115]:
```
datetime64[ns, UTC]
```

## 🔍 Inference :

- The data is consistent overall, but the interior and color columns contain a '-' value, which we will replace with 'multicolor'.
- We will also add the full state names alongside their abbreviations.
- The data type of the condition column will be changed to int, and the salesdate column will be converted to datetime.
- Now, all inconsistencies have been addressed, and the data is accurate and consistant.

# ✨➕📊 Feature engineering column extraction

**Since we will extract sale_month, sale_day_name, sale_day, and sale_year columns from the saledate column**

In [116…
```
df["sale_month"]=df.saledate.dt.month_name()
df["sale_day_name"]=df.saledate.dt.day_name()
df["sale_day"]=df.saledate.dt.day
df["sale_year"]=df.saledate.dt.year
```

**Verifying the results**

In [117…
```
df.head(2)
```

Out[117]:

| | year | make | model | trim | body | transmission | vin | state | condition | c |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2015 | Kia | Sorento | LX | SUV | automatic | 5xyktca69fg566472 | ca | 5 | |
| **1** | 2015 | Kia | Sorento | LX | SUV | automatic | 5xyktca69fg561319 | ca | 5 | |

# 🔍 Descriptive Analysis 📊

In [118…
```
df.describe().T
```

Out[118]:

| | count | mean | std | min | 25% | 50% | 75% | |
|---|---|---|---|---|---|---|---|---|
| **year** | 533648.0 | 2010.231083 | 3.828460 | 1990.0 | 2008.0 | 2012.0 | 2013.0 | |
| **condition** | 533648.0 | 30.800012 | 13.368147 | 1.0 | 24.0 | 35.0 | 42.0 | |
| **odometer** | 533648.0 | 66254.095940 | 52093.614999 | 1.0 | 27746.0 | 50146.0 | 95814.0 | 99 |
| **mmr** | 533648.0 | 14013.458825 | 9599.502736 | 25.0 | 7575.0 | 12500.0 | 18500.0 | 18 |
| **sellingprice** | 533648.0 | 13860.066505 | 9673.553875 | 1.0 | 7300.0 | 12400.0 | 18400.0 | 23 |
| **sale_day** | 533648.0 | 14.553903 | 8.661093 | 1.0 | 7.0 | 15.0 | 22.0 | |
| **sale_year** | 533648.0 | 2014.923618 | 0.265608 | 2014.0 | 2015.0 | 2015.0 | 2015.0 | |

## 🔍 Inference :

**Statistical Analysis Inference:**

1. **Year:**
   - The dataset covers the years from **1990 to 2015**, with a mean year of 2010. The data is concentrated around recent years, with most entries dating from 2012 onwards.
2. **Condition:**
   - Condition values range from 1 to 49, with an average of 30.8. The median condition score is 35, indicating significant variability in condition scores across entries.
3. **Odometer:**
   - Odometer readings span from 1 to 999,999, with a mean of 66,254 and a median of 50,146. There is substantial variability, suggesting a wide range of usage among the vehicles.
4. **MMR (Market Value):**
   - Market values range from 25 to 182,000, with an average of 14,013 and a median of 12,500. The high standard deviation reflects a wide spread in market values.
5. **Selling Price:**
   - Selling prices vary from 1 to 230,000, with a mean of 13,860 and a median of 12,400. The data shows a broad range of selling prices, with several high-value outliers.
6. **Sale Day:**
   - Sale days range from 1 to 31, with an average of 14.55 and a median of 15, indicating that most sales occur around the middle of the month.
7. **Sale Year:**
   - The sale year is predominantly 2015, with very little variation, indicating that the data is highly concentrated for this year.

**This summary provides a snapshot of the central tendencies, variability, and range of the key variables in the dataset.**

In [120…
```
df.describe(exclude='number',include='object').T
```

Out[120]:

| | count | unique | top | freq |
|---|---|---|---|---|
| **make** | 533648 | 53 | Ford | 91907 |
| **model** | 533648 | 772 | Altima | 19159 |
| **trim** | 533648 | 1510 | Base | 54092 |
| **body** | 533648 | 86 | Sedan | 194608 |
| **transmission** | 533648 | 2 | automatic | 517286 |
| **vin** | 533648 | 525638 | wbanv13588cz57827 | 5 |
| **state** | 533648 | 38 | fl | 79626 |
| **color** | 533648 | 20 | black | 106599 |
| **interior** | 533648 | 17 | black | 238663 |
| **seller** | 533648 | 12735 | nissan-infiniti lt | 19677 |
| **sale_month** | 533648 | 8 | February | 158882 |
| **sale_day_name** | 533648 | 6 | Tuesday | 175201 |

## 🔍 Inference :

**Categorical Data Analysis Inference:**

1. **Make:**
   - There are 53 unique makes in the dataset. The most common make is **Ford**, appearing **91,907** times, indicating it is the most frequently represented brand.
2. **Model:**
   - The dataset includes 772 unique models. The top model is the **Altima**, with a frequency of **19,159**, showing it is the most common model.
3. **Trim:**
   - There are 1,510 unique trims. The most frequent trim is **Base**, appearing **54,092** times, reflecting its popularity.
4. **Body:**
   - The dataset contains 86 body types. The most common body type is **Sedan**, with **194,608** occurrences, indicating it is the most prevalent body style.
5. **Transmission:**
   - There are 2 types of transmissions: **automatic** (which is the most common, with **517,286** occurrences) and one other type, suggesting that automatic transmissions dominate the dataset.
6. **VIN (Vehicle Identification Number):**
   - With 525,638 unique VINs, the most common VIN is **wbanv13588cz57827**, appearing **5** times, indicating some repetition or

errors in VIN entries.

7. **State:**
    - There are 38 unique states. The most frequently represented state is **FL** (Florida), with **79,626** occurrences, indicating a high concentration of entries from this state.
8. **Color:**
    - The dataset includes 20 unique colors. The most common color is **black**, appearing **106,599** times, suggesting it is the most popular color among the vehicles.
9. **Interior:**
    - There are 17 unique interior colors. The most frequent interior color is **black**, with **238,663** occurrences, indicating it is the most commonly reported interior color.
10. **Seller:**
    - The dataset features 12,735 unique sellers. The top seller is **nissan-infiniti lt**, with **19,677** transactions, reflecting a significant presence.
11. **Sale Month:**
    - There are 8 unique sale months. The most frequent month is **February**, with **158,882** occurrences, suggesting a higher volume of sales during this month.
12. **Sale Day Name:**
    - The dataset includes 6 unique sale days. The most common day is **Tuesday**, with **175,201** occurrences, indicating that sales are most frequent on this day.

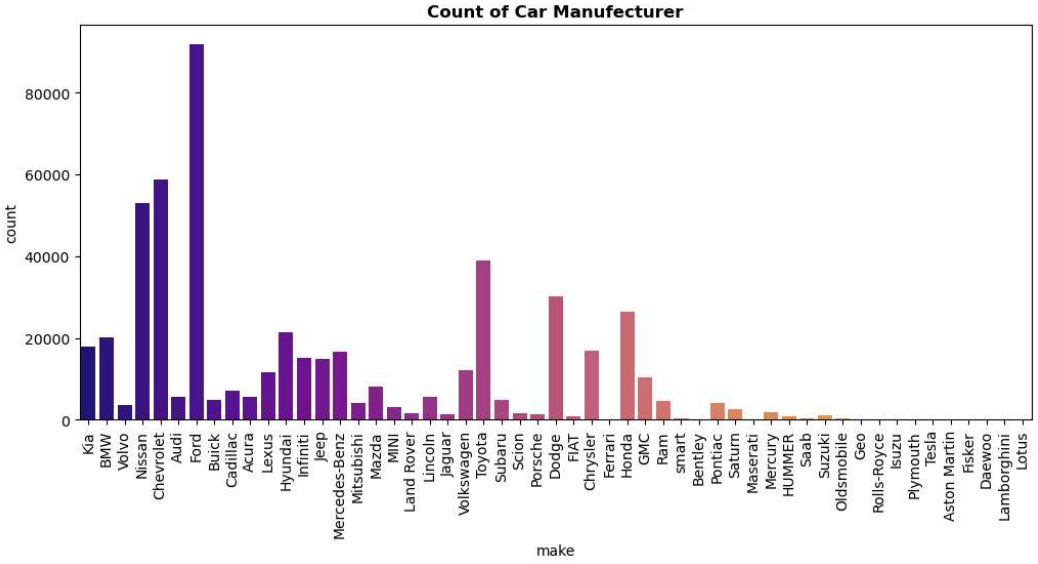
**This summary provides an overview of the most frequent values and their occurrences for each categorical variable in the dataset.**

## 🎨 Visualization and Insights 📊

In [121…
```
df['make'].value_counts().head()
```

Out[121]:
```
make
Ford          91907
Chevrolet     58817
Nissan        53085
Toyota        38991
Dodge         30100
Name: count, dtype: int64
```

In [122…
```
plt.figure(figsize=(12,5))
sns.countplot(x="make",data=df,palette='plasma')
plt.title("Count of Car Manufecturer",fontweight='bold');
plt.xticks(rotation=90);
```



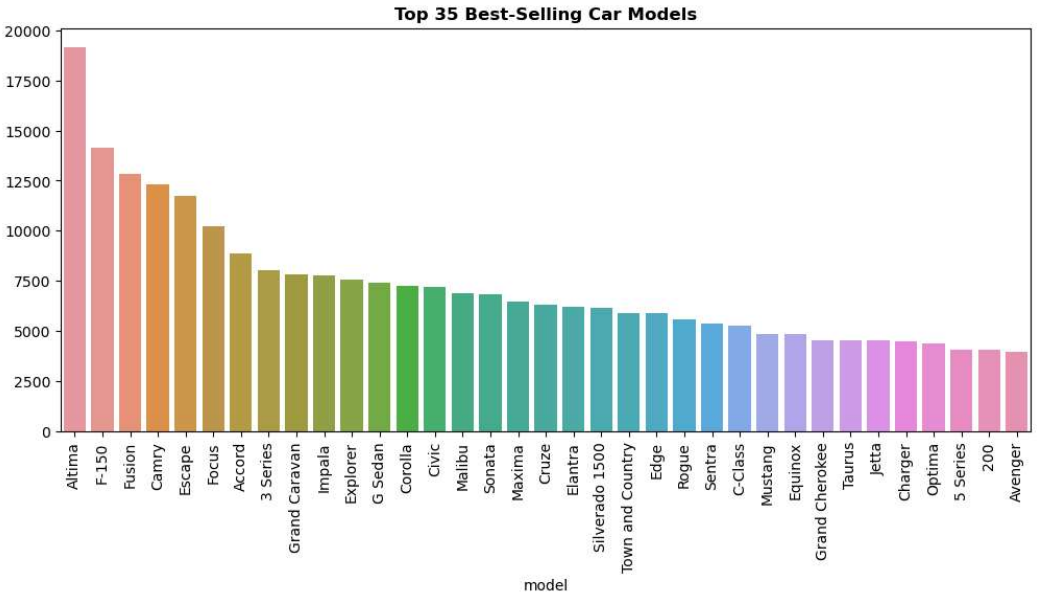

## 🔍 Inference :

- **The top five car manufacturers in the dataset are Ford (91,907 entries), Chevrolet (58,817), Nissan (53,085), Toyota (38,991), and Dodge (30,100). Ford leads significantly, indicating the highest representation, while Chevrolet and Nissan also show strong presence. Toyota and Dodge follow, reflecting their notable but lesser representation. This distribution highlights Ford's dominance and the competitive standing of the other brands in the market.**

In [123…
```
df['model'].value_counts().head(5)
```

Out[123]:
```
model
Altima    19159
F-150     14164
Fusion    12835
Camry     12326
Escape    11714
Name: count, dtype: int64
```

In [124…
```
plt.figure(figsize=(12,5))
sns.barplot(x=df['model'].value_counts().head(35).index,y=df['model'].va
plt.title("Top 35 Best-Selling Car Models",fontweight='bold');
plt.xticks(rotation=90);
```



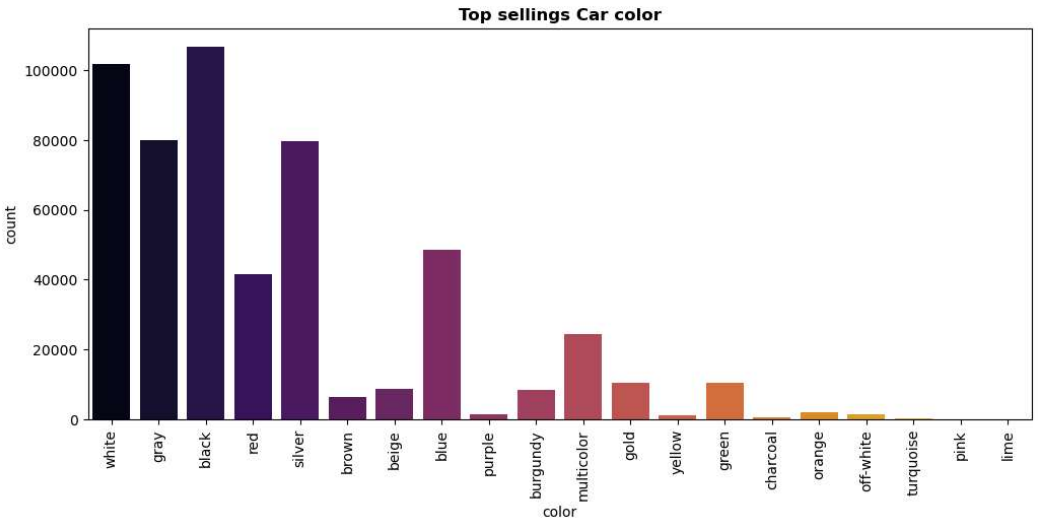

## 🔍 Inference :

- **The Altima is the top seller, leading by a significant margin. The F-150 follows as a strong second, with the Fusion, Camry, and Escape also showing notable sales figures, reflecting their strong market positions.**

In [125…
```
df['color'].value_counts()
```

Out[125]:
```
color
black         106599
white         101834
gray           80001
silver         79514
blue           48532
red            41635
multicolor     24371
gold           10471
green          10361
beige           8591
burgundy        8501
brown           6493
orange          1940
purple          1477
off-white       1398
yellow          1196
charcoal         464
turquoise        215
pink              41
lime              14
Name: count, dtype: int64
```

In [126…
```
plt.figure(figsize=(12,5))
sns.countplot(x="color",data=df,palette='inferno')
plt.title("Top sellings Car color",fontweight='bold');
plt.xticks(rotation=90);
```



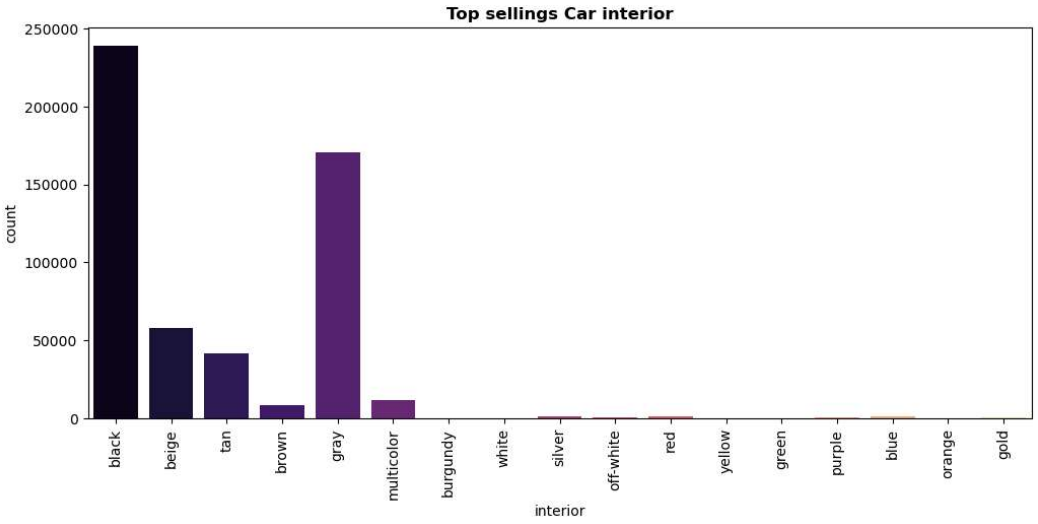

## 🔍 Inference :

- **The top-selling car colors are black, white, gray, silver, and blue. Black leads the market, followed by white, with gray and silver also showing strong popularity. Blue, while less prevalent, remains a notable choice among car buyers**

In [127…
```
df['interior'].value_counts()
```

Out[127]:
```
interior
black          238663
gray           170422
beige           57868
tan             41703
multicolor      11425
brown            8296
red              1298
silver           1053
blue             1013
off-white         469
purple            326
gold              313
white             239
green             223
burgundy          184
orange            134
yellow             19
Name: count, dtype: int64
```

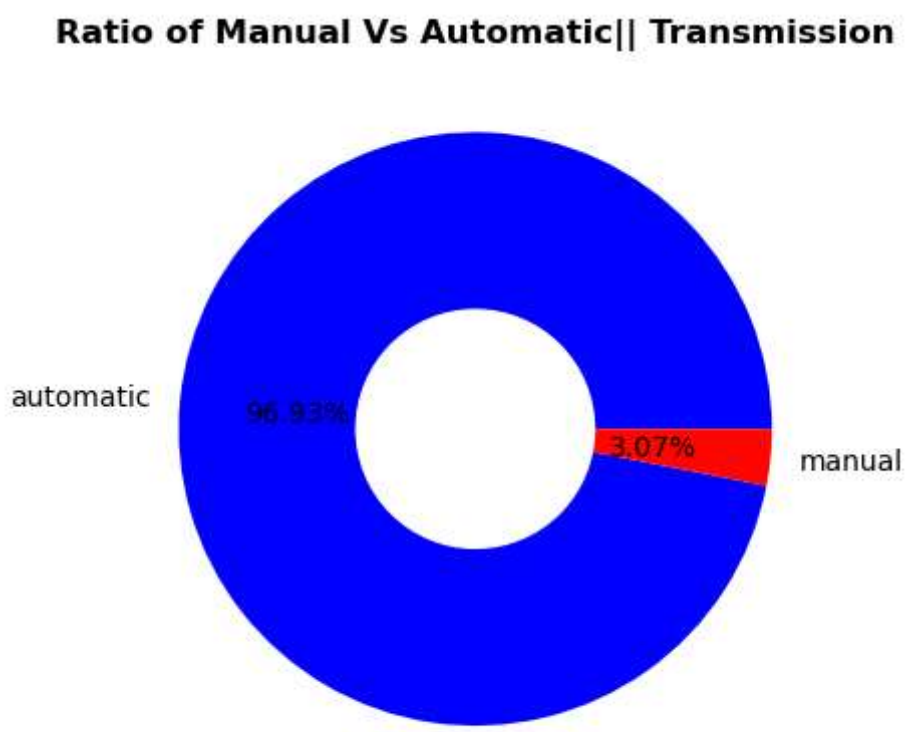
In [128…
```
plt.figure(figsize=(12,5))
sns.countplot(x="interior",data=df,palette='magma')
plt.title("Top sellings Car interior",fontweight='bold');
plt.xticks(rotation=90);
```




## 🔍 Inference :

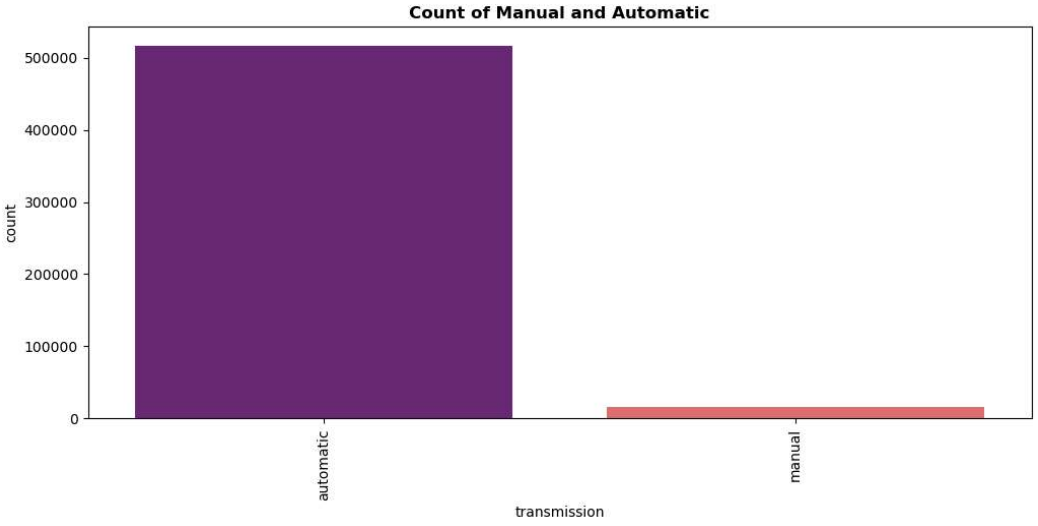
- **The top-selling car interior colors are black, gray, beige, tan, and multicolor. Black leads by a large margin, with gray following as a strong second. Beige and tan also show significant popularity, while multicolor reflects a distinctive but less common choice among buyers.**

In [129…
```
plt.pie(df["transmission"].value_counts(),labels=df["transmission"].valu
plt.title("Ratio of Manual Vs Automatic|| Transmission",fontweight='bold
circle = plt.Circle( (0,0), 0.4, color='white')
p=plt.gcf()
p.gca().add_artist(circle);
```




In [130…

```
plt.figure(figsize=(12,5))
sns.countplot(x="transmission",data=df,palette='magma')
plt.title("Count of Manual and Automatic",fontweight='bold');
plt.xticks(rotation=90);
```

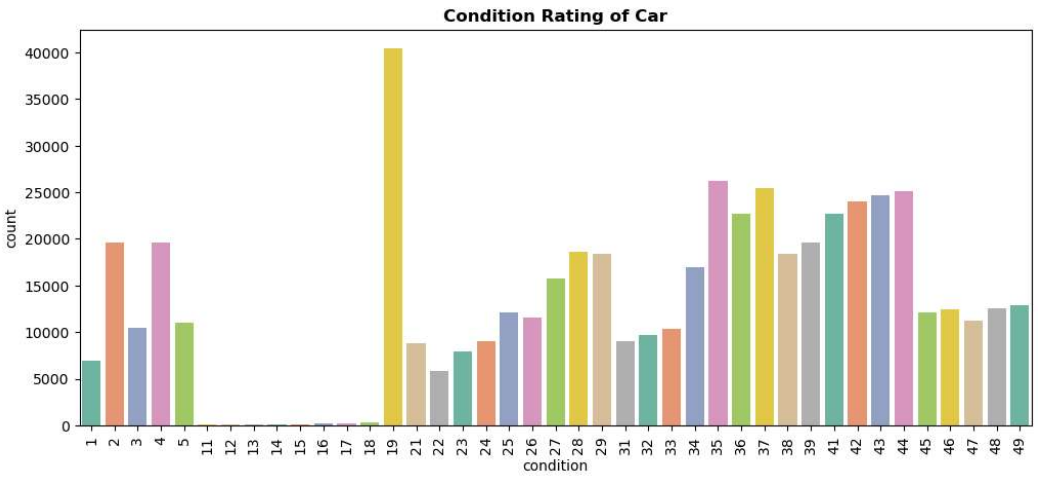



## 🔍 Inference :

- **In the transmission column, the data reveals that manual transmissions account for only 3% of the vehicles, while automatic transmissions dominate at 97%.**

In [131…

```
plt.figure(figsize=(12,5))
sns.countplot(x="condition",data=df,palette='Set2')
plt.title("Condition Rating of Car",fontweight='bold');
plt.xticks(rotation=90);
```

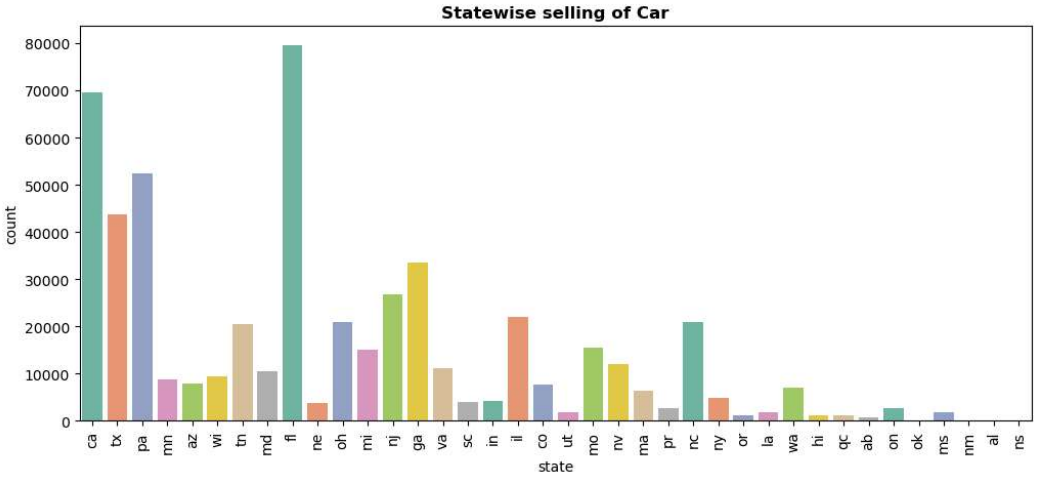



## 🔍 Inference :

- **The condition column shows a wide range of vehicle conditions, with code 19 being the most common, representing 40,399 vehicles. Other frequently observed conditions include codes 35, 37, 44, and 43, each with over 24,000 vehicles. In contrast, condition codes like 12, 11, 13, and 15 are rare, with fewer than 200 vehicles each. This distribution suggests that most vehicles fall within a standard range of conditions, with a few in either very high or very low condition categories.**

In [132…

```
plt.figure(figsize=(12,5))
sns.countplot(x="state",data=df,palette='Set2')
plt.title("Statewise selling of Car",fontweight='bold');
plt.xticks(rotation=90);
```



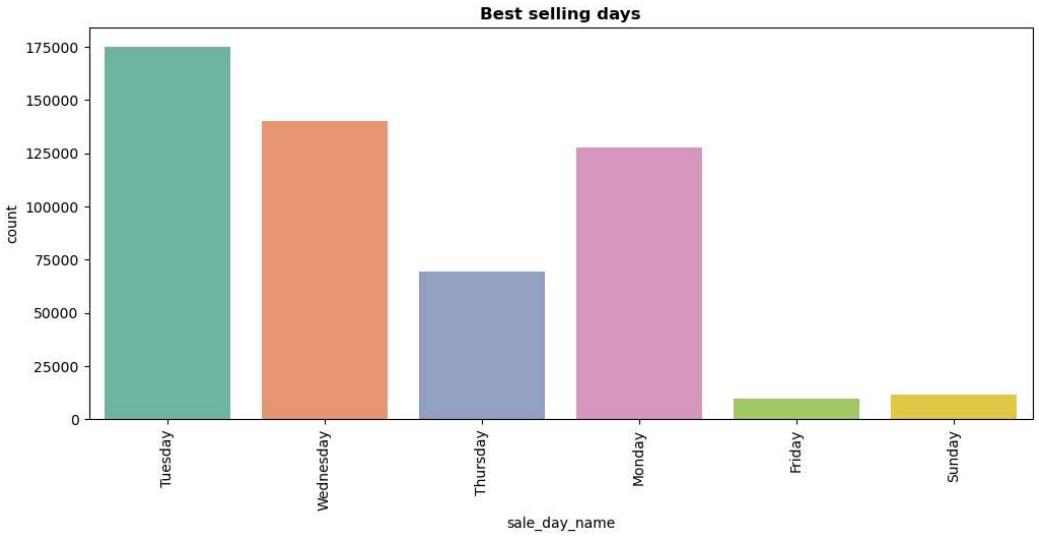

## 🔍 Inference :

- **Florida (FL) leads in car sales with 79,626 vehicles sold, followed by California (CA) with 69,636 vehicles, and Pennsylvania (PA) with 52,411 vehicles. These states represent the highest volume of car sales in the dataset. Texas (TX) and Georgia (GA) also show significant sales figures, with 43,703 and 33,486 vehicles sold, respectively.**

- **On the other end, states like Oklahoma (OK), Nova Scotia (NS), and Alabama (AL) have the lowest car sales, each with fewer than 100 vehicles sold. This suggests that car sales are highly concentrated in a few states, particularly in Florida and California, with a steep drop-off in sales volume as we move down the list. The data highlights regional**

**differences in car sales, with certain states playing a dominant role in the market.**

In [133…
```
plt.figure(figsize=(12,5))
sns.countplot(x="sale_day_name",data=df,palette='Set2')
plt.title("Best selling days",fontweight='bold');
plt.xticks(rotation=90);
```

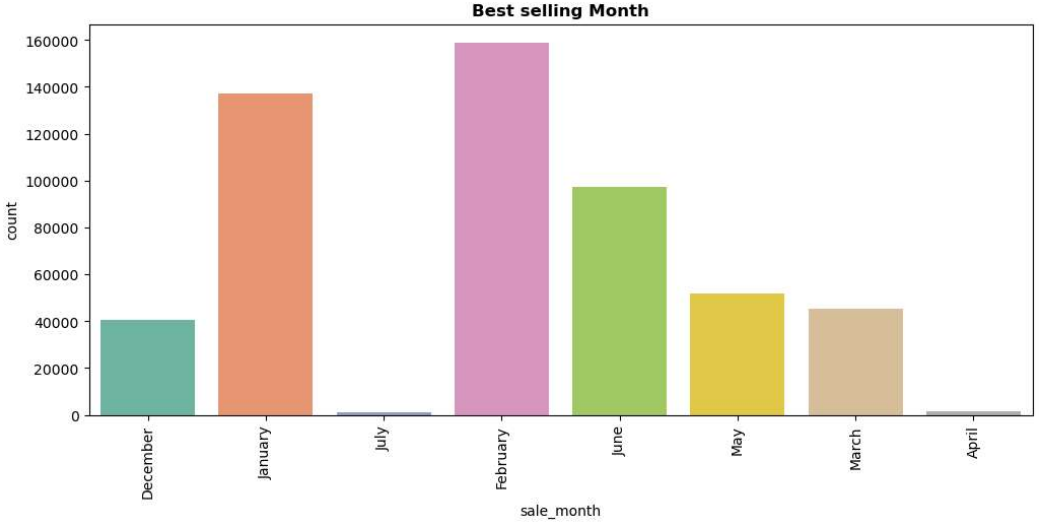



## 🔍 Inference :

**Tuesday is the best-selling day of the week, with 175,201 vehicles sold, significantly outperforming other days. Wednesday follows as the second-best day with 140,338 vehicles sold, while Monday ranks third with 127,518 vehicles sold.**

**Sales drop notably on Thursday, with 69,339 vehicles sold, and further decrease on Sunday, with only 11,478 vehicles sold. Friday has the lowest sales with 9,774 vehicles. This indicates a clear peak in sales on Tuesday, suggesting it is the most favorable day for car sales, with a marked decrease as the week progresses towards the weekend.**

In [134…
```
plt.figure(figsize=(12,5))
sns.countplot(x="sale_month",data=df,palette='Set2')
plt.title("Best selling Month",fontweight='bold');
plt.xticks(rotation=90);
```




## 🔍 Inference :

- **February is the best-selling month with 158,882 vehicles sold, followed by January with 137,293 vehicles. Sales drop significantly in June (97,238) and May (51,847), and further decrease in March and December. April and July have the lowest sales, with only 1,391 and 1,232 vehicles sold, respectively. This indicates a strong peak in early months and a notable decline as the year progresses.**

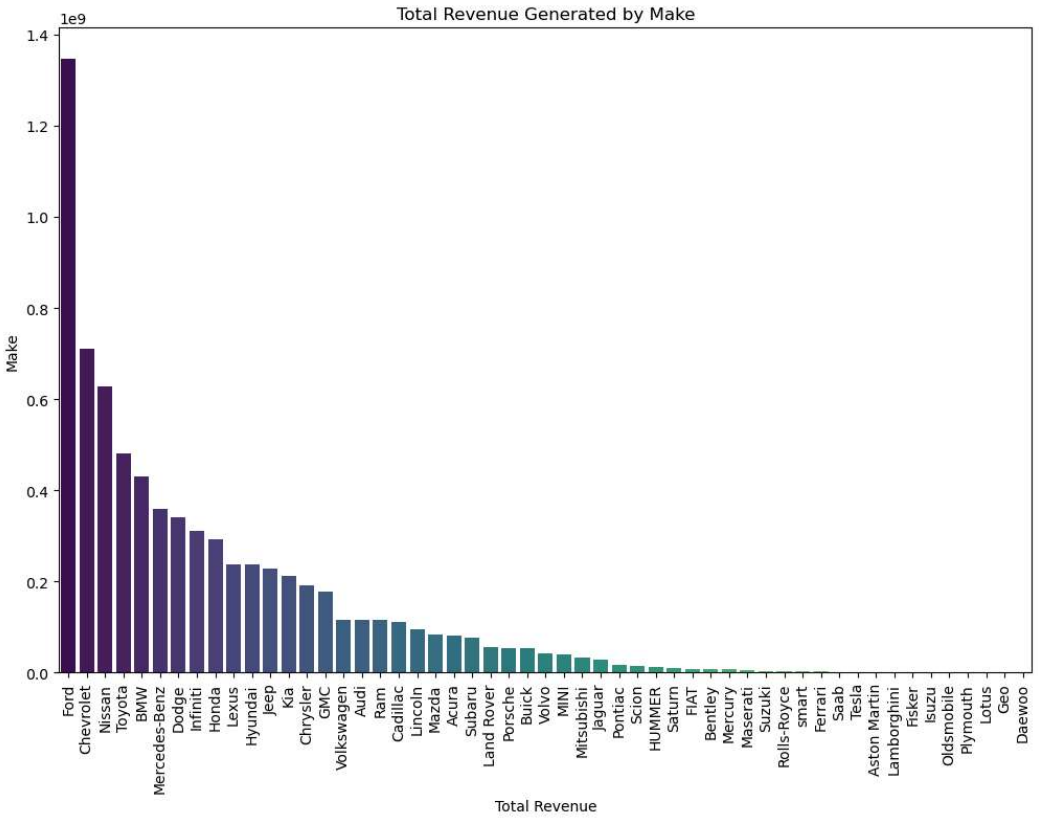
In [135…

```
revenue_by_make = df.groupby('make')['sellingprice'].sum().reset_index()

revenue_by_make = revenue_by_make.sort_values(by='sellingprice', ascendi

plt.figure(figsize=(12, 8))
sns.barplot(x='make', y='sellingprice', data=revenue_by_make, palette='v


plt.xticks(rotation=90);
plt.xlabel('Total Revenue')
plt.ylabel('Make')
plt.title('Total Revenue Generated by Make')
plt.show()
```




## 🔍 Inference :

**Ford generates the highest total revenue, amounting to approximately 1.35 billion, far surpassing other makes. Chevrolet and Nissan follow with revenues of about 711 million and 628 million, respectively. Toyota and BMW also contribute significantly, with revenues of 482 million and 431 million.**

**Other notable makes include Mercedes-Benz and Dodge, generating 360 million and 340 million, respectively. Brands such as Infiniti, Honda, and Lexus have revenues in the range of 238 million to 311 million.**

**In contrast, luxury and niche brands like Ferrari, Rolls-Royce, and Lamborghini show much lower revenue figures, ranging from 450,000 to**

**2.4 million. This data indicates a strong revenue concentration among a few leading brands, with a sharp decline as we move towards less common or luxury vehicles.**

# 📊 Conclusion

- **Best-Selling Day**: 🗓️ Tuesday tops the sales charts with 175,201 vehicles sold, making it the most favorable day for sales. Wednesday and Monday follow with 140,338 and 127,518 vehicles sold, respectively, while sales drop significantly towards the weekend, with Friday having the lowest sales at 9,774.
- **Best-Selling Month**: 🗓️ February leads in car sales with 158,882 vehicles sold, followed by January with 137,293 vehicles. Sales peak early in the year and decrease notably through the months, with April and July showing the lowest figures.
- **Transmission Types**: 🚗 Automatic transmissions dominate with 97% of the vehicles, while manual transmissions are much less common, representing only 3%. This indicates a strong preference for automatic transmissions in the dataset.
- **Condition Distribution**: 🚗 The condition column reveals a high concentration of vehicles in condition code 19 (40,399 vehicles). Other frequently occurring conditions include 35, 37, and 44, while conditions like 12 and 13 are much less common, indicating a concentration in average conditions with fewer extreme cases.
- **State-Wise Sales**: 🌎 Florida leads with 79,626 vehicles sold, followed by California with 69,636. The lowest sales are seen in Oklahoma and Nova Scotia, with fewer than 100 vehicles sold, showing a strong regional variation in car sales.
- **Total Revenue by Make**: 💰 Ford generates the highest total revenue at approximately 1.35 billion, far ahead of Chevrolet and Nissan with 711 million and 628 million, respectively. Luxury brands like Ferrari and Rolls-Royce have much lower revenue figures, indicating a significant concentration of revenue among leading makes.

In [ ]: