Name : K.Bhaskar

Rollno: 2211CS010273

group:4

--

Dataset Description:

The dataset consists of 329 records and 7 attributes, containing both numerical and categorical data. It is well-structured, with no missing or duplicate values, ensuring data integrity. The dataset primarily includes five numerical columns and two categorical columns, suggesting a combination of quantitative and qualitative information. Given its small size, with a memory usage of approximately 0.055 MB, it is lightweight and easy to process. The catalog contains portability transactions data under the One Nation One Ration Card (ONORC) plan, which facilitates seamless access to ration benefits across states. This dataset has been released under the National Data Sharing and Accessibility Policy (NDSAP), ensuring open access to relevant public data. The dataset is contributed by the **Ministry of Consumer**

Affairs, Food, and Public Distribution, Department of Food and Public Distribution, further emphasizing its authenticity and relevance for public welfare analysis. The absence of missing values makes it suitable for immediate analysis without requiring extensive preprocessing.

In [ ]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
```

In [3]:

```
df = pd.read_csv("cb.csv")
```

In [4]:

```
df
```

Out[4]:

	homestatecode	salestatecode	month	year	txn_count	salestatename	homestate
0	5	1	5	2024	13	JAMMU AND KASHMIR	UTTARAKH
1	6	1	5	2024	43	JAMMU AND KASHMIR	HAR
2	7	1	5	2024	5	JAMMU AND KASHMIR	
3	8	1	5	2024	2	JAMMU AND KASHMIR	RAJAS
4	9	1	5	2024	2438	JAMMU AND KASHMIR	UTTAR PRA
...	...	...	...	...	...	...	
324	24	38	5	2024	813	Dadar & Nagar Haveli & Daman & Diu	GU
325	27	38	5	2024	504	Dadar & Nagar Haveli & Daman & Diu	MAHARAS
326	28	38	5	2024	1	Dadar & Nagar Haveli & Daman & Diu	AN PRA
327	29	38	5	2024	3	Dadar & Nagar Haveli & Daman & Diu	KARN
328	32	38	5	2024	1	Dadar & Nagar Haveli & Daman & Diu	KE

329 rows × 7 columns

In [5]:

```
df.head(5)
```

Out[5]:

	homestatecode	salestatecode	month	year	txn_count	salestatename	homestatena
0	5	1	5	2024	13	JAMMU AND KASHMIR	UTTARAKHA
1	6	1	5	2024	43	JAMMU AND KASHMIR	HARYA
2	7	1	5	2024	5	JAMMU AND KASHMIR	DE
3	8	1	5	2024	2	JAMMU AND KASHMIR	RAJASTH
4	9	1	5	2024	2438	JAMMU AND KASHMIR	UTTAR PRADE

Frist five data from dataset

In [6]:
```
print(df.columns)
```

```
Index(['homestatecode', 'salestatecode', 'month', 'year', 'txn_count',
       'salestatename', 'homestatename'],
      dtype='object')
```

No of columns in our dataset

In [7]:
```
print(df.sample(10))
```

```
     homestatecode  salestatecode  month  year  txn_count  \
71              21              7      5  2024         38   
255             18             30      5  2024          4   
170              9             22      5  2024         59   
159              7             20      5  2024         33   
84               7              8      5  2024        124   
82               5              8      5  2024          8   
315              7             38      5  2024          5   
327             29             38      5  2024          3   
299             19             36      5  2024          9   
101              8              9      5  2024          7   

                          salestatename  homestatename  
71                                DELHI         ODISHA  
255                                 GOA          ASSAM  
170                        CHHATTISGARH  UTTAR PRADESH  
159                           JHARKHAND          DELHI  
84                            RAJASTHAN          DELHI  
82                            RAJASTHAN    UTTARAKHAND  
315  Dadar & Nagar Haveli & Daman & Diu          DELHI  
327  Dadar & Nagar Haveli & Daman & Diu      KARNATAKA  
299                           TELANGANA    WEST BENGAL  
101                       UTTAR PRADESH      RAJASTHAN  
```

Frist 10 rows in our dataset

In [8]:

```
df.tail(5)
```

Out[8]:

	homestatecode	salestatecode	month	year	txn_count	salestatename	homestate
324	24	38	5	2024	813	Dadar & Nagar Haveli & Daman & Diu	GU
325	27	38	5	2024	504	Dadar & Nagar Haveli & Daman & Diu	MAHARAS
326	28	38	5	2024	1	Dadar & Nagar Haveli & Daman & Diu	AN PRA
327	29	38	5	2024	3	Dadar & Nagar Haveli & Daman & Diu	KARN
328	32	38	5	2024	1	Dadar & Nagar Haveli & Daman & Diu	KI

Last five data in our dataset

In [9]:

```
df.shape
```

Out[9]:

```
(329, 7)
```

This the shape of our dataset

In [10]:

```
df.dtypes
```

Out[10]:

```
homestatecode     int64
salestatecode     int64
month             int64
year              int64
txn_count         int64
salestatename    object
homestatename    object
dtype: object
```

The data types presented in the dataset

In [11]:

```
df.info
```

Out[11]:

```
<bound method DataFrame.info of      homestatecode  salestatecode  month  year
txn_count  \
0                5              1      5  2024         13
1                6              1      5  2024         43
2                7              1      5  2024          5
3                8              1      5  2024          2
4                9              1      5  2024       2438
..             ...            ...    ...   ...        ...
324             24             38      5  2024        813
325             27             38      5  2024        504
326             28             38      5  2024          1
327             29             38      5  2024          3
328             32             38      5  2024          1

                          salestatename   homestatename
0                     JAMMU AND KASHMIR     UTTARAKHAND
1                     JAMMU AND KASHMIR         HARYANA
2                     JAMMU AND KASHMIR           DELHI
3                     JAMMU AND KASHMIR       RAJASTHAN
4                     JAMMU AND KASHMIR   UTTAR PRADESH
..                                  ...             ...
324  Dadar & Nagar Haveli & Daman & Diu         GUJARAT
325  Dadar & Nagar Haveli & Daman & Diu     MAHARASHTRA
326  Dadar & Nagar Haveli & Daman & Diu  ANDHRA PRADESH
327  Dadar & Nagar Haveli & Daman & Diu       KARNATAKA
328  Dadar & Nagar Haveli & Daman & Diu          KERALA

[329 rows x 7 columns]>
```

This is the data info of dataset

In [12]:

```
df.head
```

```
Out[12]: <bound method NDFrame.head of      homestatecode  salestatecode  month  year  t
xn_count  \
0                5              1      5  2024         13
1                6              1      5  2024         43
2                7              1      5  2024          5
3                8              1      5  2024          2
4                9              1      5  2024       2438
..             ...            ...    ...   ...        ...
324             24             38      5  2024        813
325             27             38      5  2024        504
326             28             38      5  2024          1
327             29             38      5  2024          3
328             32             38      5  2024          1

                          salestatename   homestatename
0                     JAMMU AND KASHMIR     UTTARAKHAND
1                     JAMMU AND KASHMIR         HARYANA
2                     JAMMU AND KASHMIR           DELHI
3                     JAMMU AND KASHMIR       RAJASTHAN
4                     JAMMU AND KASHMIR   UTTAR PRADESH
..                                  ...             ...
324  Dadar & Nagar Haveli & Daman & Diu         GUJARAT
325  Dadar & Nagar Haveli & Daman & Diu     MAHARASHTRA
326  Dadar & Nagar Haveli & Daman & Diu  ANDHRA PRADESH
327  Dadar & Nagar Haveli & Daman & Diu       KARNATAKA
328  Dadar & Nagar Haveli & Daman & Diu          KERALA

[329 rows x 7 columns]>
```

show the frist 5 rows of the dataset

```
In [13]: df.tail
```

```
Out[13]: <bound method NDFrame.tail of      homestatecode  salestatecode  month  year  t
         xn_count  \
         0                5              1      5  2024         13
         1                6              1      5  2024         43
         2                7              1      5  2024          5
         3                8              1      5  2024          2
         4                9              1      5  2024       2438
         ..             ...            ...    ...   ...        ...
         324             24             38      5  2024        813
         325             27             38      5  2024        504
         326             28             38      5  2024          1
         327             29             38      5  2024          3
         328             32             38      5  2024          1

                                   salestatename   homestatename
         0                     JAMMU AND KASHMIR     UTTARAKHAND
         1                     JAMMU AND KASHMIR         HARYANA
         2                     JAMMU AND KASHMIR           DELHI
         3                     JAMMU AND KASHMIR       RAJASTHAN
         4                     JAMMU AND KASHMIR   UTTAR PRADESH
         ..                                  ...             ...
         324  Dadar & Nagar Haveli & Daman & Diu         GUJARAT
         325  Dadar & Nagar Haveli & Daman & Diu     MAHARASHTRA
         326  Dadar & Nagar Haveli & Daman & Diu  ANDHRA PRADESH
         327  Dadar & Nagar Haveli & Daman & Diu       KARNATAKA
         328  Dadar & Nagar Haveli & Daman & Diu          KERALA

         [329 rows x 7 columns]>
```

show the last 5 rows of our Dataset

```
In [14]: df.isnull
```

Out[14]:
```
<bound method DataFrame.isnull of      homestatecode  salestatecode  month  yea
r  txn_count  \
0                5              1      5  2024         13
1                6              1      5  2024         43
2                7              1      5  2024          5
3                8              1      5  2024          2
4                9              1      5  2024       2438
..             ...            ...    ...   ...        ...
324             24             38      5  2024        813
325             27             38      5  2024        504
326             28             38      5  2024          1
327             29             38      5  2024          3
328             32             38      5  2024          1

                          salestatename   homestatename
0                     JAMMU AND KASHMIR     UTTARAKHAND
1                     JAMMU AND KASHMIR         HARYANA
2                     JAMMU AND KASHMIR           DELHI
3                     JAMMU AND KASHMIR       RAJASTHAN
4                     JAMMU AND KASHMIR   UTTAR PRADESH
..                                  ...             ...
324  Dadar & Nagar Haveli & Daman & Diu         GUJARAT
325  Dadar & Nagar Haveli & Daman & Diu     MAHARASHTRA
326  Dadar & Nagar Haveli & Daman & Diu  ANDHRA PRADESH
327  Dadar & Nagar Haveli & Daman & Diu       KARNATAKA
328  Dadar & Nagar Haveli & Daman & Diu          KERALA

[329 rows x 7 columns]>
```

This checks for missing values in our dataset

In [15]:
```
null_values = df.isnull().sum()

null_values
```

Out[15]:
```
homestatecode    0
salestatecode    0
month            0
year             0
txn_count        0
salestatename    0
homestatename    0
dtype: int64
```

This will show the number of missing values in each column of the dataset

In [50]:
```
df.dropna()
```

Out[50]:

	homestatecode	salestatecode	month	year	txn_count	salestatename	homestate
0	5	1	5	2024	13	JAMMU AND KASHMIR	UTTARAK
1	6	1	5	2024	43	JAMMU AND KASHMIR	HAR
2	7	1	5	2024	5	JAMMU AND KASHMIR	
3	8	1	5	2024	2	JAMMU AND KASHMIR	RAJAS
4	9	1	5	2024	2438	JAMMU AND KASHMIR	UTTAR PRA
...	...	...	...	...	...	...	
324	24	38	5	2024	813	Dadar & Nagar Haveli & Daman & Diu	GU
325	27	38	5	2024	504	Dadar & Nagar Haveli & Daman & Diu	MAHARAS
326	28	38	5	2024	1	Dadar & Nagar Haveli & Daman & Diu	AN PRA
327	29	38	5	2024	3	Dadar & Nagar Haveli & Daman & Diu	KARN
328	32	38	5	2024	1	Dadar & Nagar Haveli & Daman & Diu	K

329 rows × 8 columns

drop rows or columns with missing values from dataset

In [16]:

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

Out[16]:

```
homestatecode    0
salestatecode    0
month            0
year             0
txn_count        0
salestatename    0
homestatename    0
dtype: int64
```

This shows the count of missing values in each column

In [17]:

```
c
```

```
Dataset size after removing duplicates: (329, 7)
```

Removes duplicate rows from the dataset and prints the updated shape

In [18]:

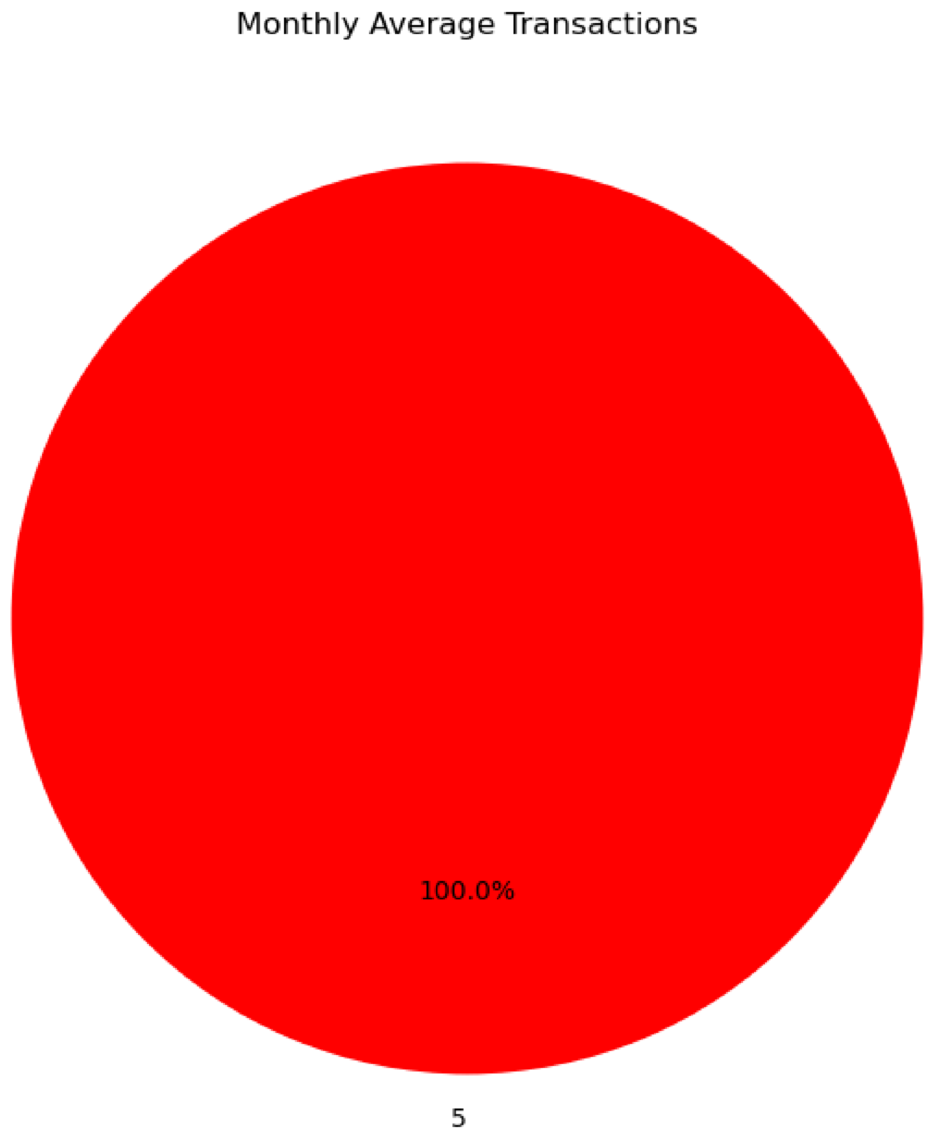
```
df_cleaned = df.dropna()
print(f"Original dataset size: {df.shape[0]} rows")
print(f"Cleaned dataset size: {df_cleaned.shape[0]} rows")
print("Remaining null values per column:")
print(df_cleaned.isnull().sum())
```

```
Original dataset size: 329 rows
Cleaned dataset size: 329 rows
Remaining null values per column:
homestatecode    0
salestatecode    0
month            0
year             0
txn_count        0
salestatename    0
homestatename    0
dtype: int64
```

Drop duplicates in our dataset

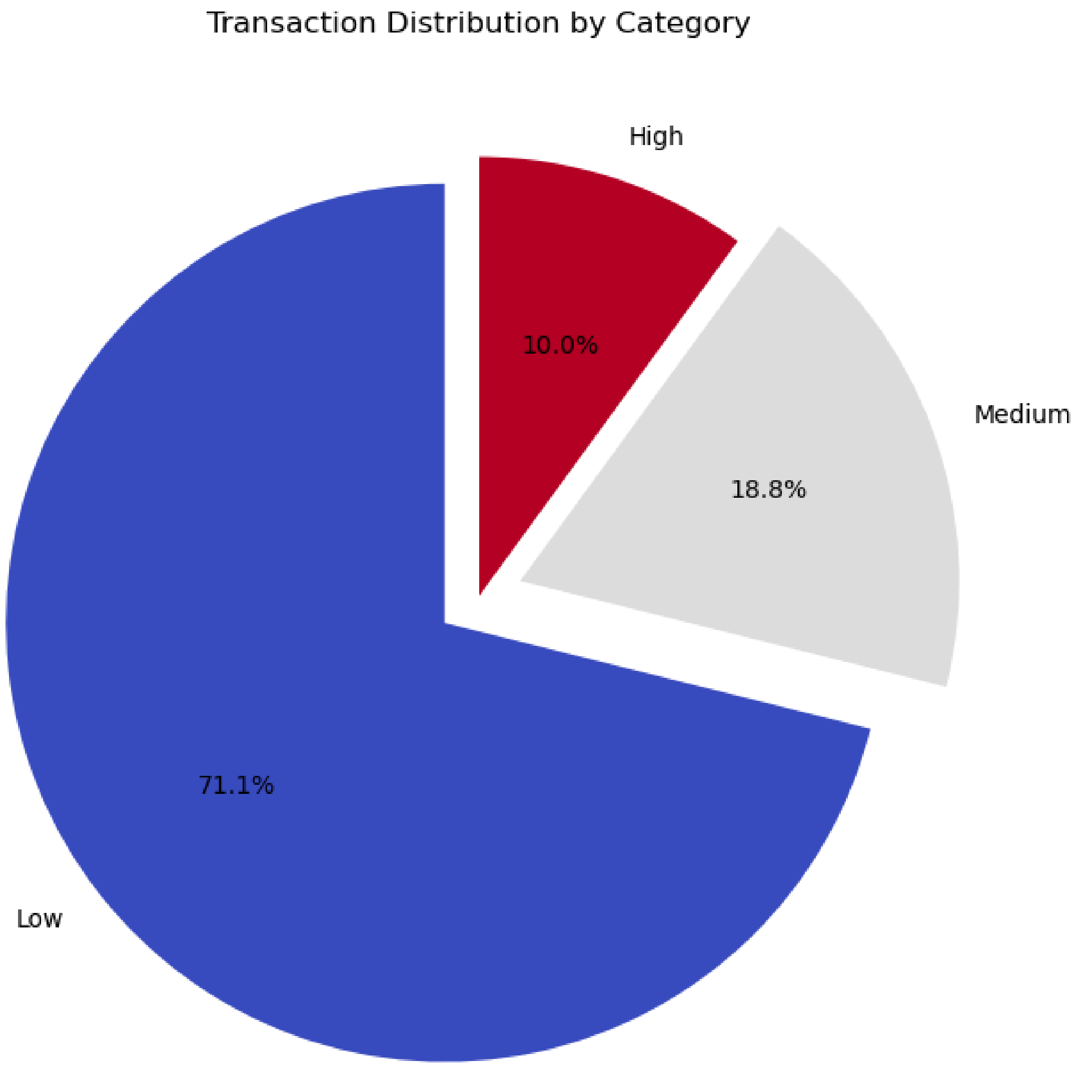
In [19]:

```
monthly_avg_txn = df.groupby('month')['txn_count'].mean()
plt.figure(figsize=(8, 8))
monthly_avg_txn.plot(kind='pie', autopct='%1.1f%%', startangle=90, cmap='autumn'
plt.show()
```



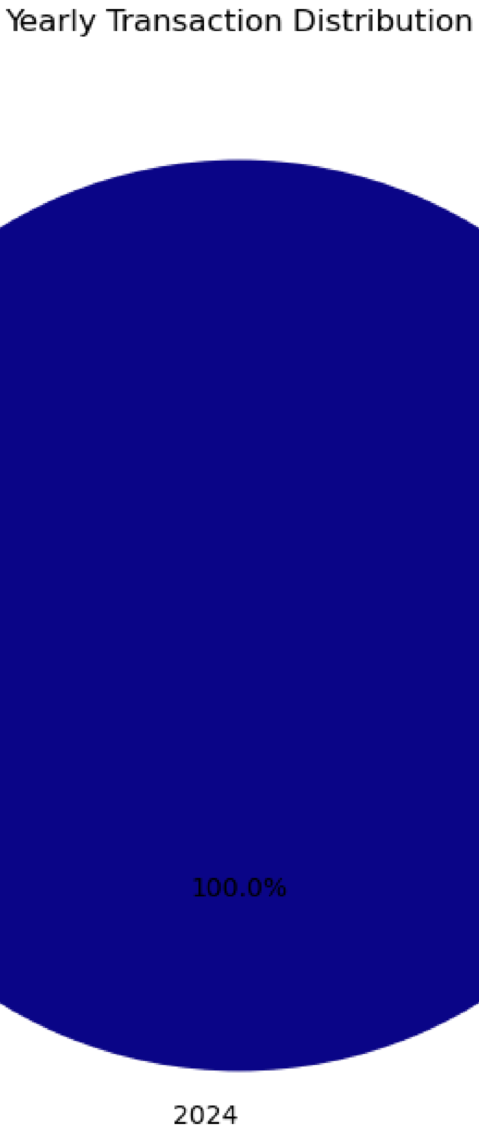
This pie chart show the Monthly Average Transactions of our dataset

In [47]:
```
plt.figure(figsize=(8, 8))
df["transaction_category"].value_counts().plot.pie(autopct='%1.1f%%', startangle
plt.title("Transaction Distribution by Category")
plt.ylabel("")
plt.show()
```



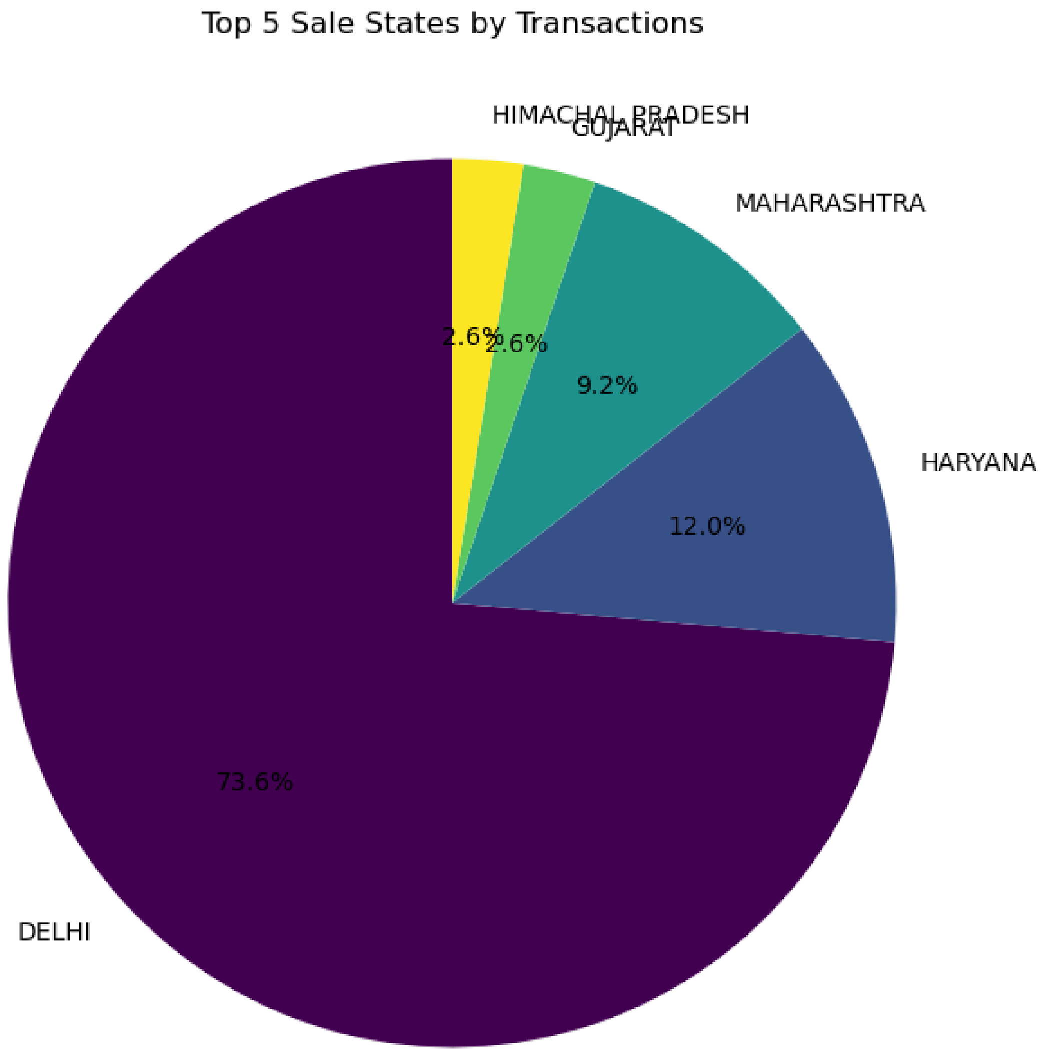
This pie chart show the Transaction Distribution by Category and low percentage is 71%

```
In [21]: yearly_txn = df.groupby('year')['txn_count'].sum()
         plt.figure(figsize=(8, 8))
         yearly_txn.plot(kind='pie', autopct='%1.1f%%', startangle=90, cmap='plasma', yla
         plt.show()
```



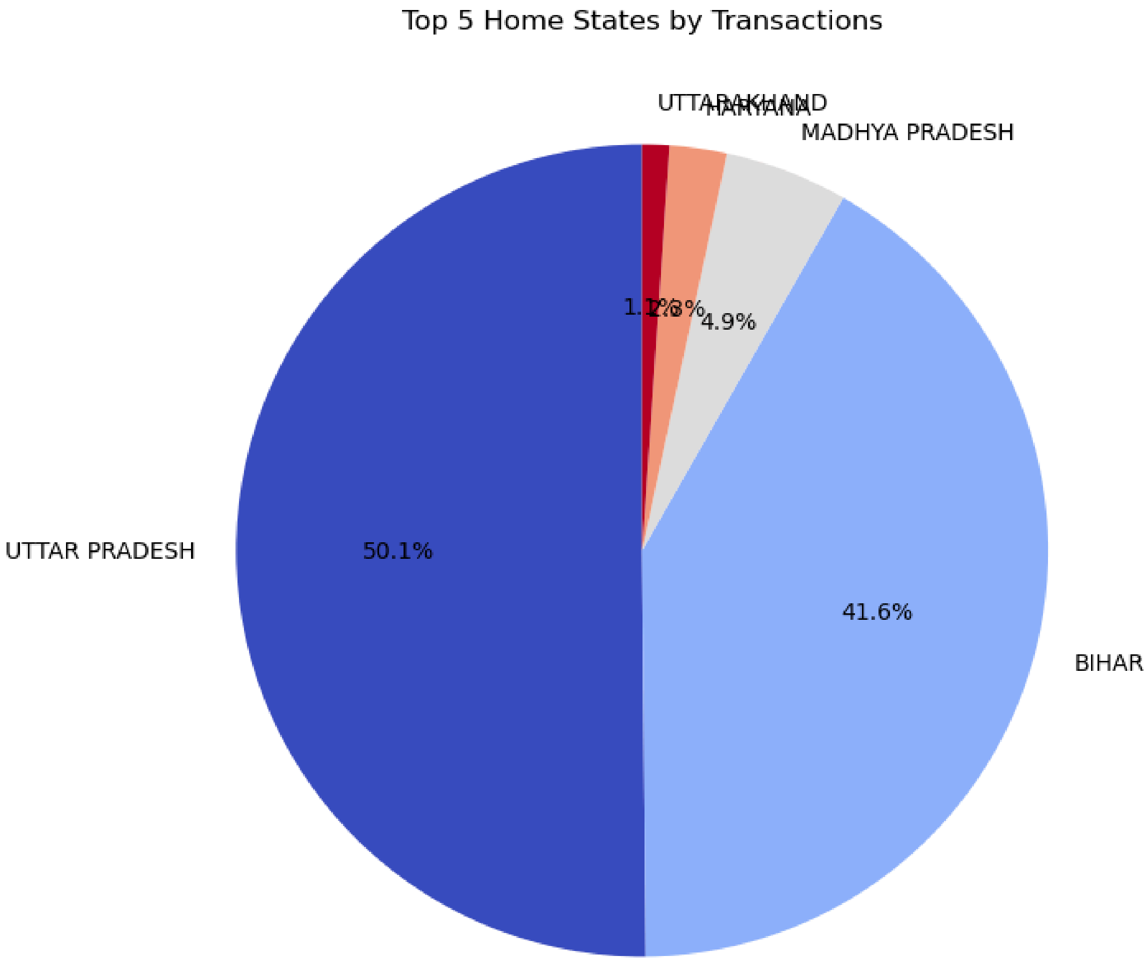
This pie chart show the Yearly Transaction Distribution

In [22]:
```
top_sale_states = df.groupby('salestatename')['txn_count'].sum().sort_values(asc

plt.figure(figsize=(8, 8))
top_sale_states.plot(kind='pie', autopct='%1.1f%%', startangle=90, cmap='viridis
plt.show()
```



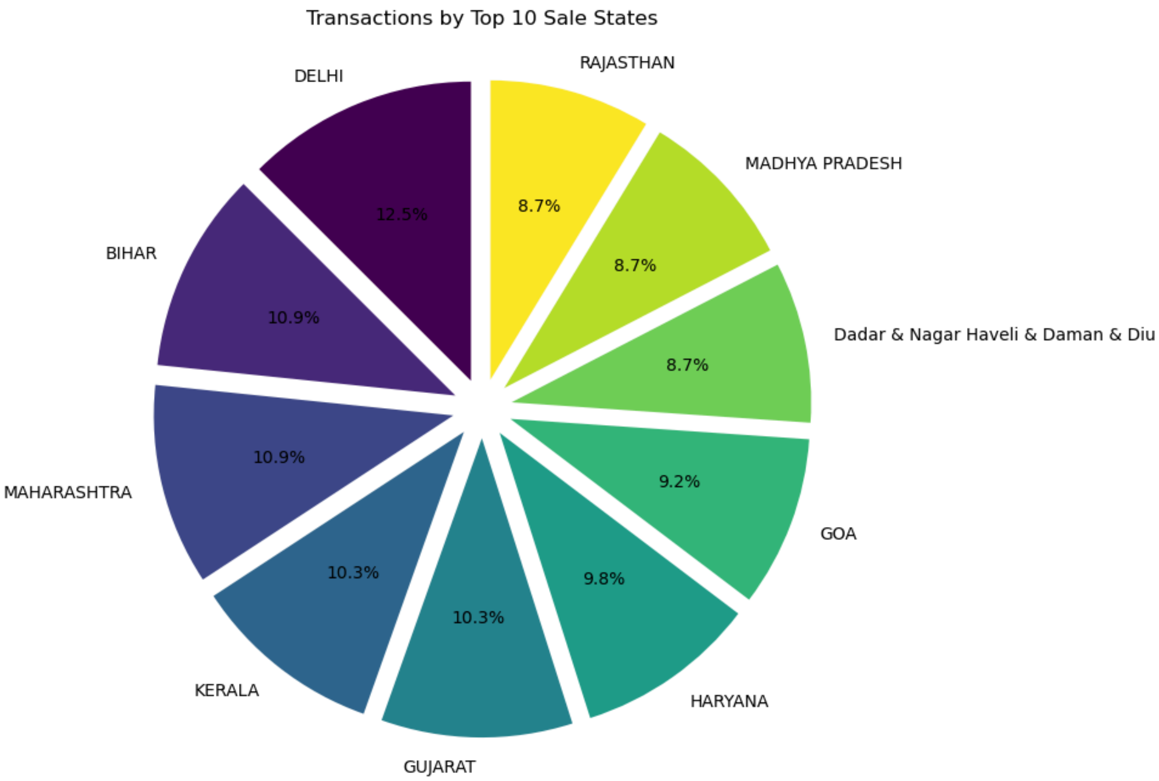
This pie chart show the top 5 sale states by transactions and delhi is the highest

```
In [23]: top_home_states = df.groupby('homestatename')['txn_count'].sum().sort_values(asc

         plt.figure(figsize=(8, 8))
         top_home_states.plot(kind='pie', autopct='%1.1f%%', startangle=90, cmap='coolwar
         plt.show()
```



This pie chart show the top 5 home states by transactions and uttar pradesh is hghest

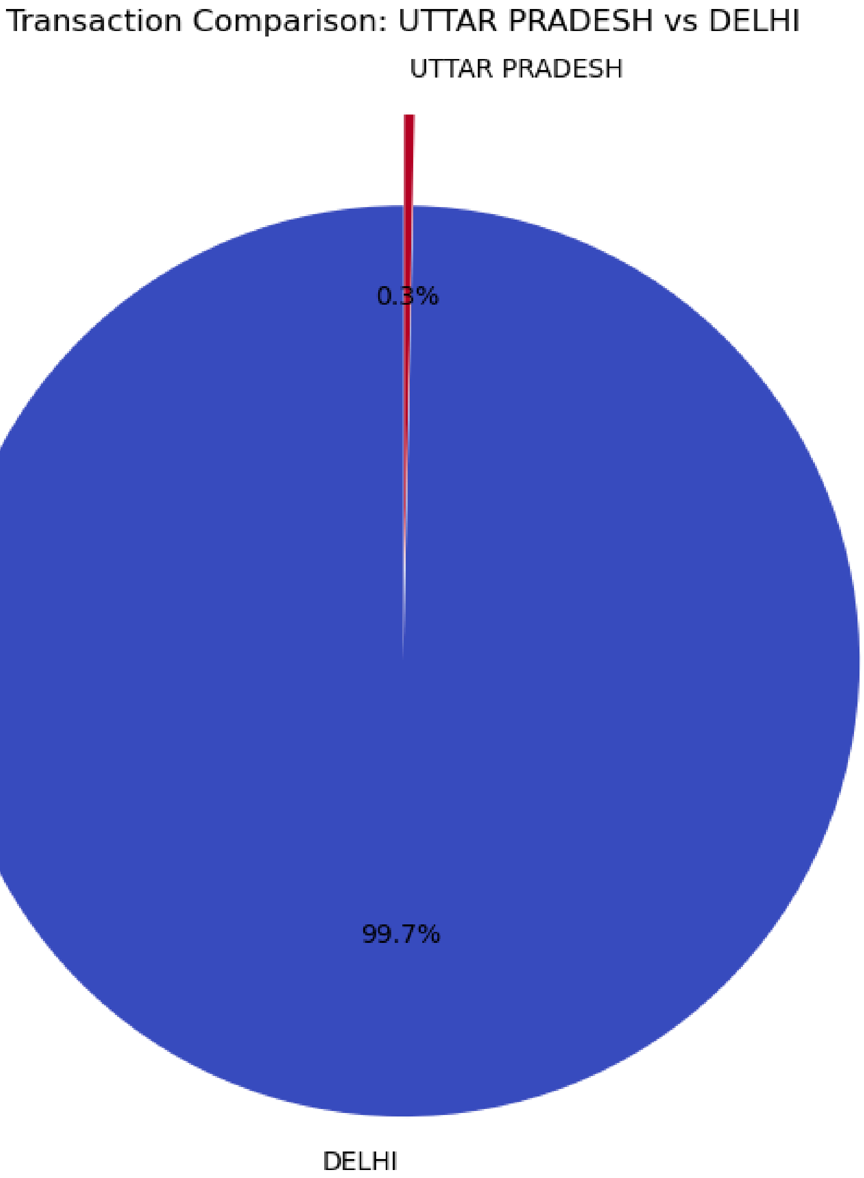
```
In [24]: top_10_states = df['salestatename'].value_counts().nlargest(10)
         explode_values = [0.1] * 10
         plt.figure(figsize=(8, 8))
         top_10_states.plot.pie(autopct='%1.1f%%', startangle=90, cmap='viridis', explode
         plt.title("Transactions by Top 10 Sale States")
         plt.ylabel("")
         plt.show()
```



This pie chart show the Transactions by top 10 Sale State and Delhi is highest and rajasthan, madhya pradesh are the lowest

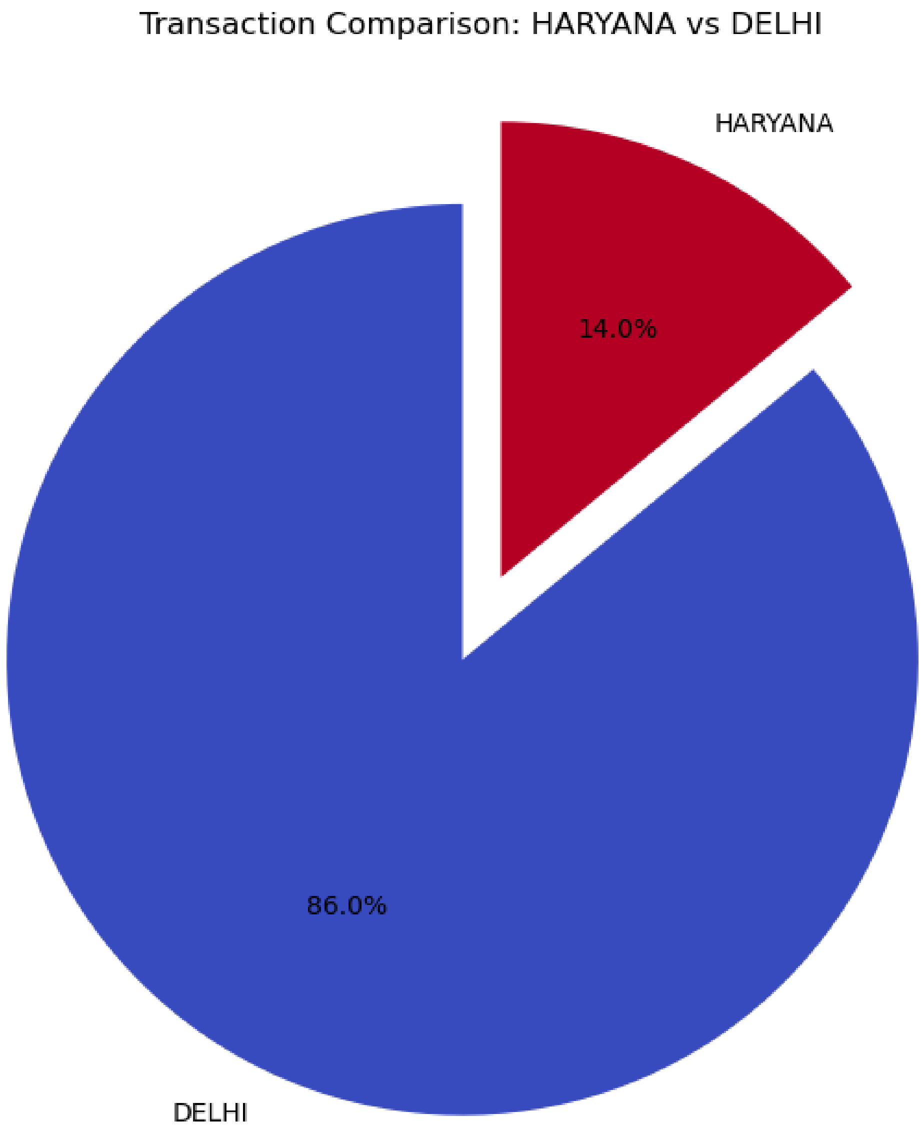
In [25]:

```
state_1 = "UTTAR PRADESH"
state_2 = "DELHI"
states_data = df[df['salestatename'].isin([state_1, state_2])].groupby("salestat
plt.figure(figsize=(8, 8))
states_data.plot.pie(autopct='%1.1f%%', startangle=90, cmap='coolwarm', explode=
plt.title(f"Transaction Comparison: {state_1} vs {state_2}")
plt.ylabel("")
plt.show()
```



Comparing Transactions between Two States (DELHI and UTTAR PRADESH) and DELHI is highest

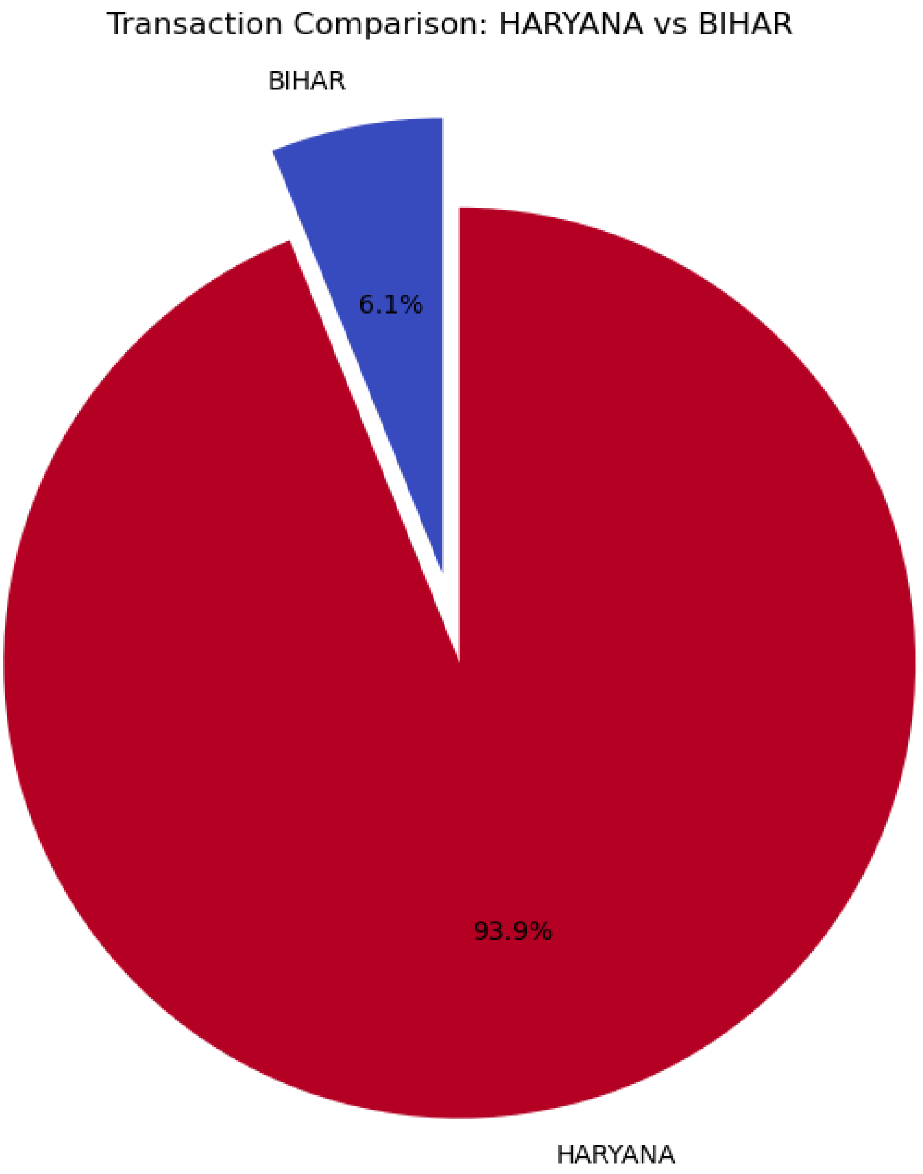
```
In [26]: state_1 = "HARYANA"
         state_2 = "DELHI"
         states_data = df[df['salestatename'].isin([state_1, state_2])].groupby("salestat
         plt.figure(figsize=(8, 8))
         states_data.plot.pie(autopct='%1.1f%%', startangle=90, cmap='coolwarm', explode=
         plt.title(f"Transaction Comparison: {state_1} vs {state_2}")
         plt.ylabel("")
         plt.show()
```



Comparing Transactions between Two States (DELHI and HARYANA) and DELHI is the highest

In [27]:

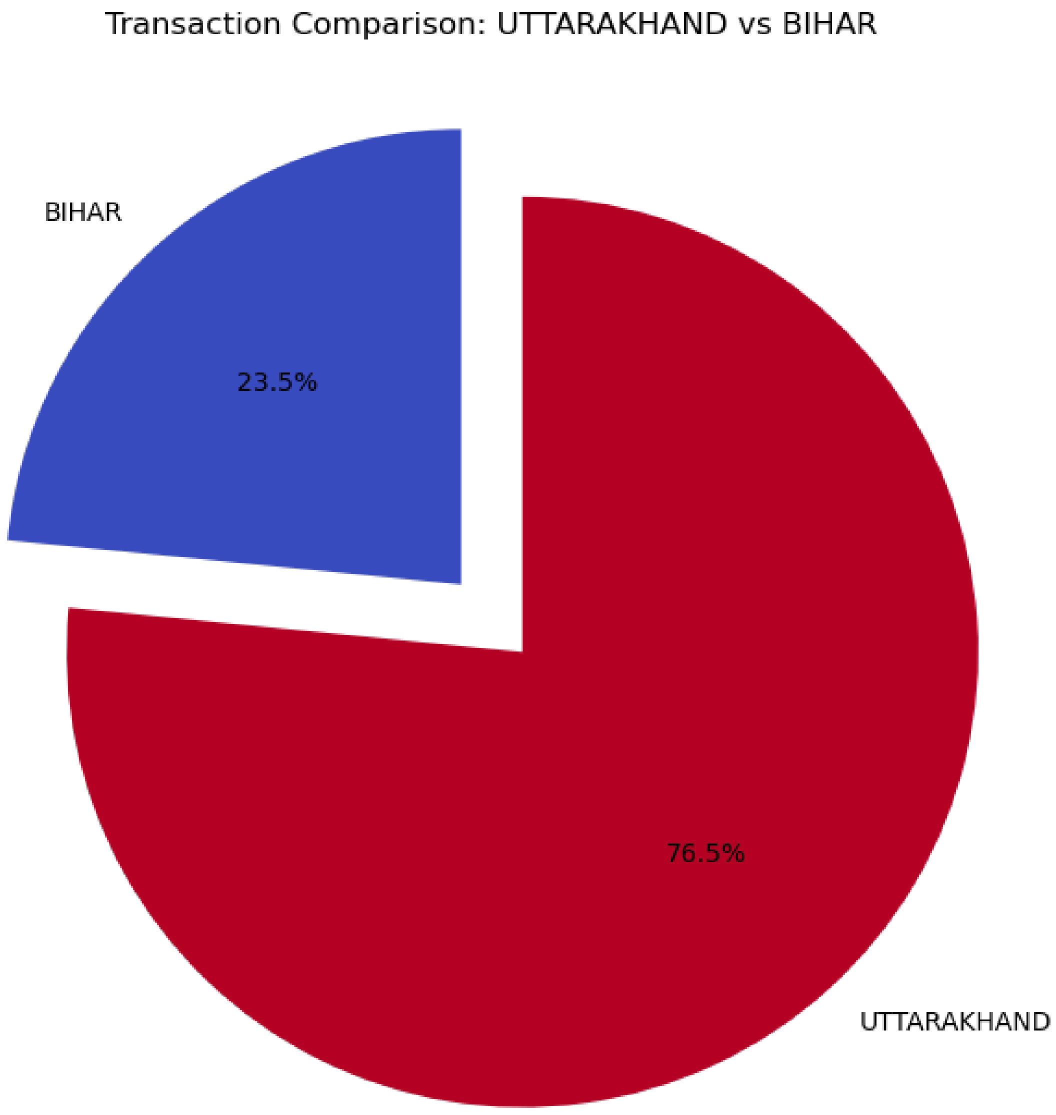
```
state_1 = "HARYANA"
state_2 = "BIHAR"
states_data = df[df['salestatename'].isin([state_1, state_2])].groupby("salestat
plt.figure(figsize=(8, 8))
states_data.plot.pie(autopct='%1.1f%%', startangle=90, cmap='coolwarm', explode=
plt.title(f"Transaction Comparison: {state_1} vs {state_2}")
plt.ylabel("")
plt.show()
```



Comparing Transactions between Two States (BIHAR and HARYANA) and HARYANA is the highest with 93.9%

In [46]:

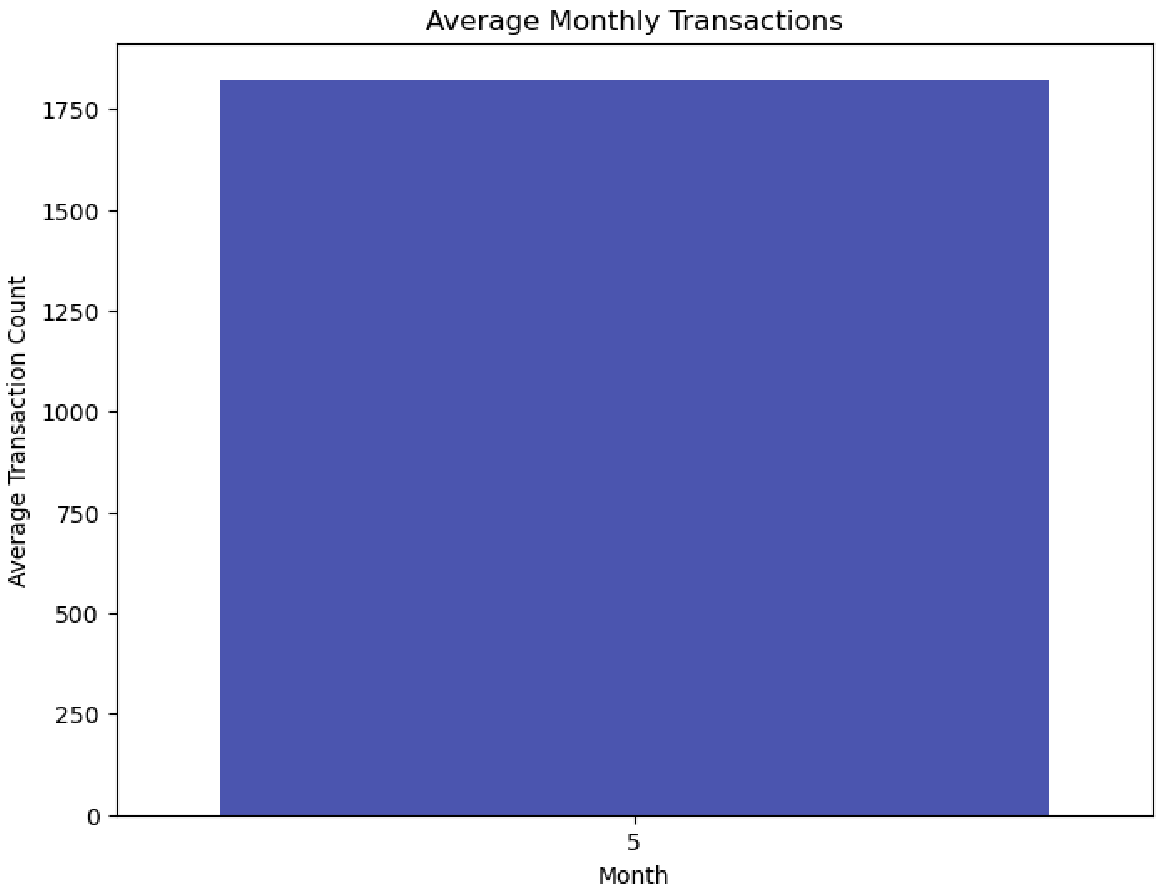
```
state_1 = "UTTARAKHAND"
state_2 = "BIHAR"
states_data = df[df['salestatename'].isin([state_1, state_2])].groupby("salestat
plt.figure(figsize=(8, 8))
states_data.plot.pie(autopct='%1.1f%%', startangle=90, cmap='coolwarm', explode=
plt.title(f"Transaction Comparison: {state_1} vs {state_2}")
plt.ylabel("")
plt.show()
```



Comparing Transactions between Two States (BIHAR and UTTARAKHAND) and UTTARAKHAND is the highest with 76.5%

In [28]:

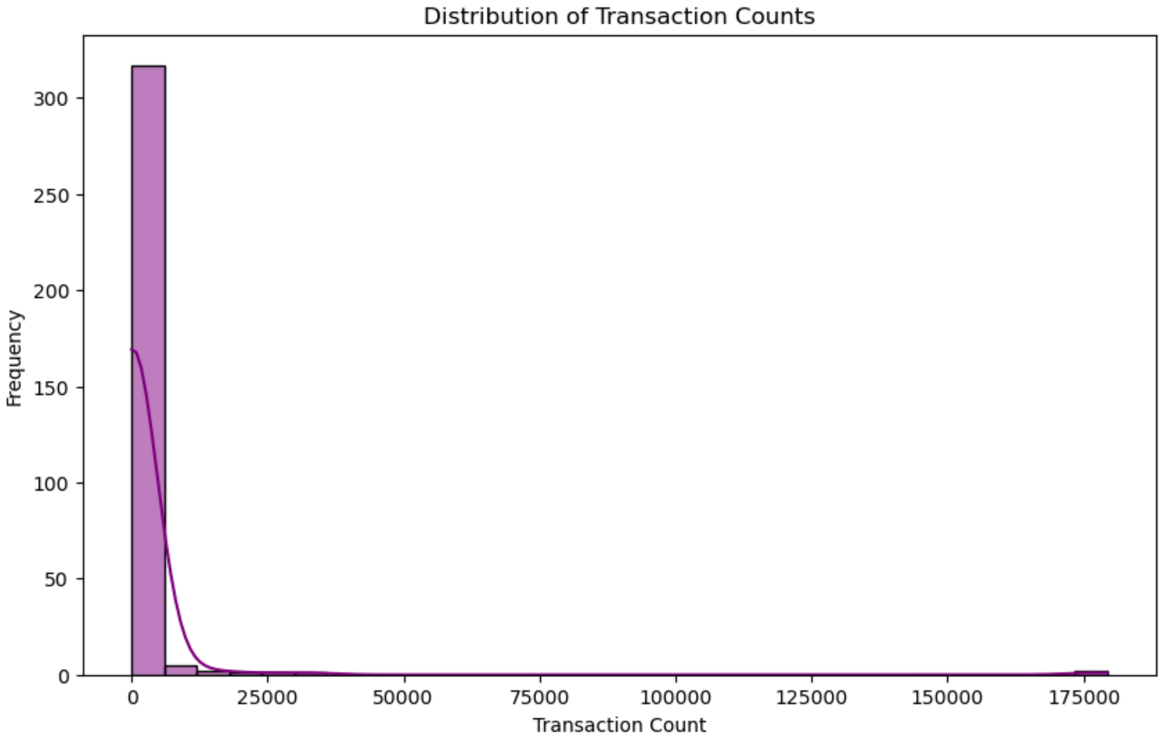
```
monthly_txn = df.groupby('month')['txn_count'].mean().reset_index()
plt.figure(figsize=(8, 6))
sns.barplot(data=monthly_txn, x='month', y='txn_count', hue='month', palette='cc
plt.title("Average Monthly Transactions")
plt.xlabel("Month")
plt.ylabel("Average Transaction Count")
plt.legend([], [], frameon=False)
plt.show()
```



This is the average transaction count of our dataset and 1750 is the highest average monthly transaction

In [29]:

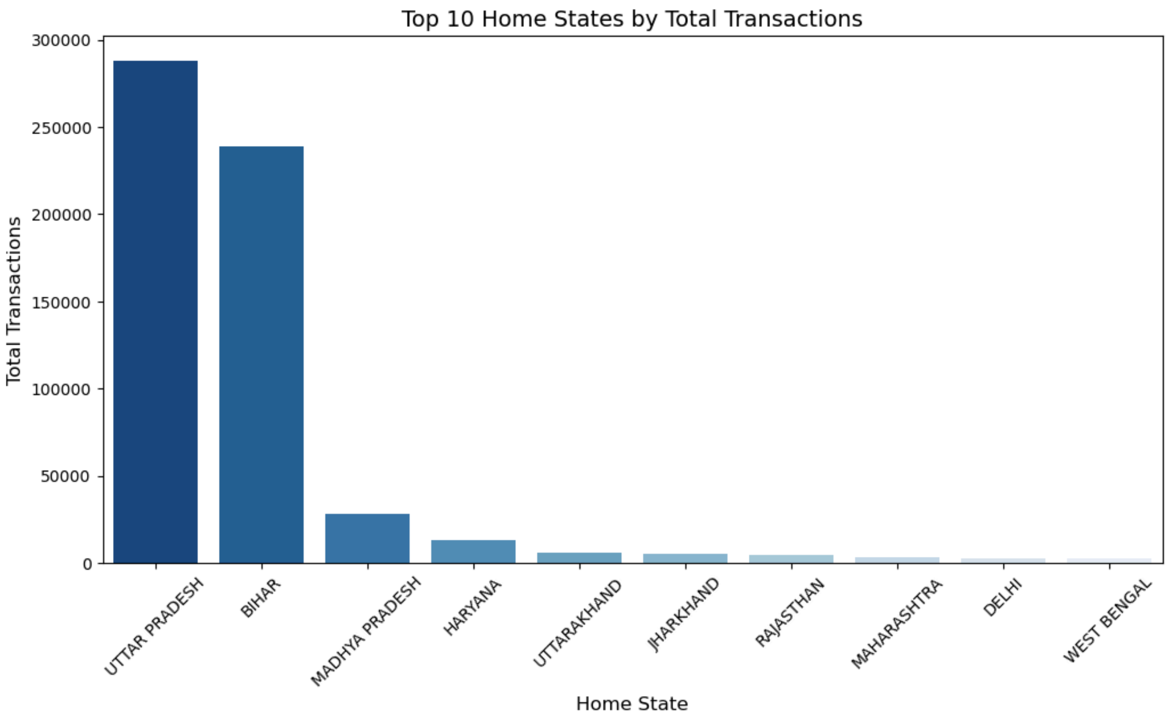
```
plt.figure(figsize=(10, 6))
sns.histplot(df['txn_count'], bins=30, kde=True, color='purple')
plt.title("Distribution of Transaction Counts")
plt.xlabel("Transaction Count")
plt.ylabel("Frequency")
plt.show()
```



This shows Distribution of Transaction Counts of the dataset

In [49]:
```
top_home_states = df.groupby('homestatename')['txn_count'].sum().sort_values(asc
plt.figure(figsize=(12, 6))
sns.barplot(x=top_home_states.index, y=top_home_states.values, palette='Blues_r'
plt.title("Top 10 Home States by Total Transactions", fontsize=14)
plt.xlabel("Home State", fontsize=12)
plt.ylabel("Total Transactions", fontsize=12)
plt.xticks(rotation=45)
plt.show()
```

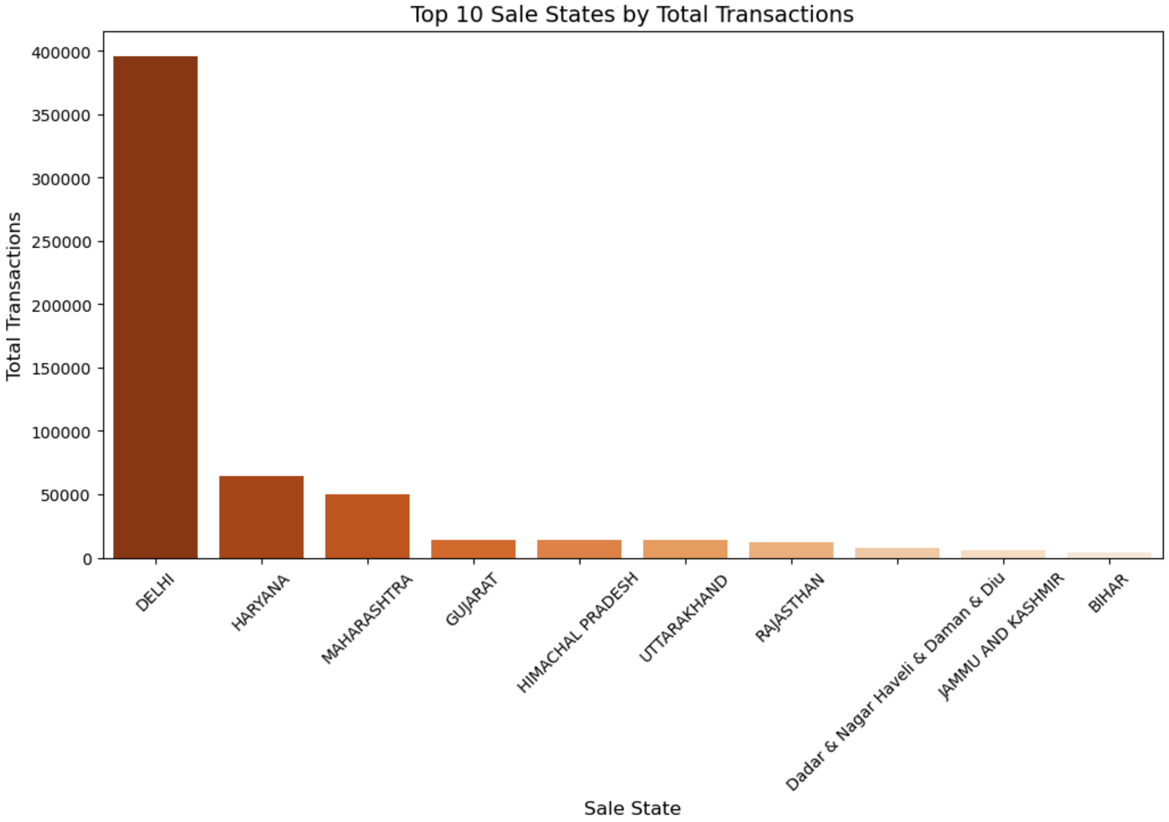
```
C:\Users\91998\AppData\Local\Temp\ipykernel_3608\1682953177.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.barplot(x=top_home_states.index, y=top_home_states.values, palette='Blues_
r')
```



Bar Chart shows the Top 10 Home States by Total Transactions of our dataset

In [31]:
```
top_sale_states = df.groupby('salestatename')['txn_count'].sum().sort_values(asc

plt.figure(figsize=(12, 6))
sns.barplot(x=top_sale_states.index, y=top_sale_states.values, palette='Oranges_
plt.title("Top 10 Sale States by Total Transactions", fontsize=14)
plt.xlabel("Sale State", fontsize=12)
plt.ylabel("Total Transactions", fontsize=12)
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\91998\AppData\Local\Temp\ipykernel_3608\1850974802.py:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.barplot(x=top_sale_states.index, y=top_sale_states.values, palette='Oranges
_r')
```

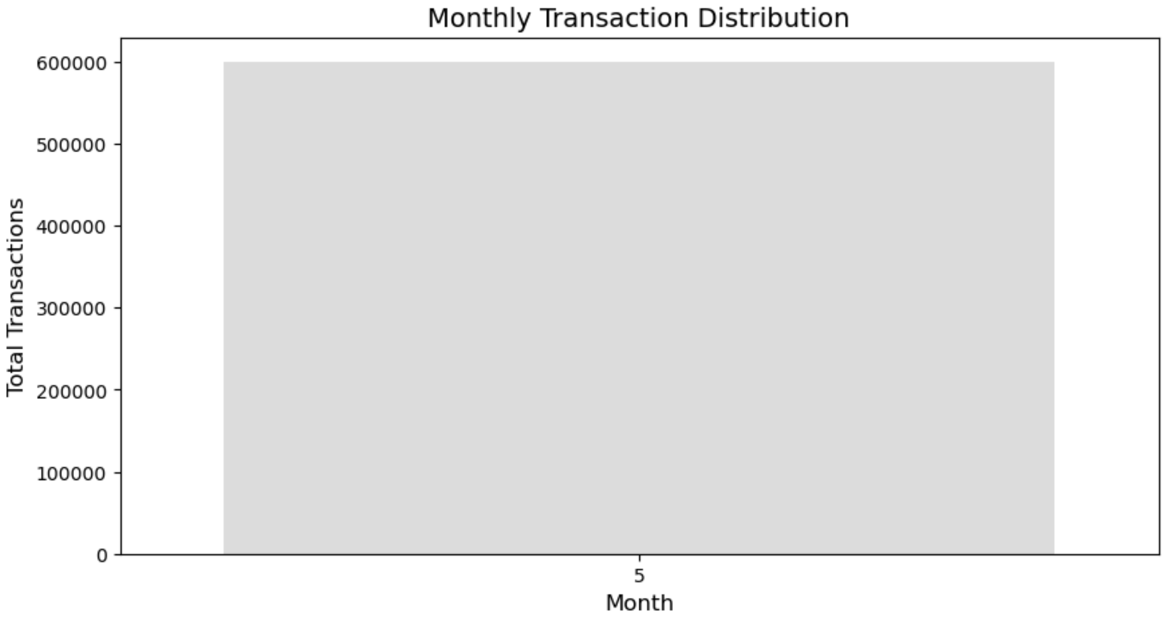


This Bar Chart shows the Top 10 Sale States by Total Transactions in the dataset

In [32]:

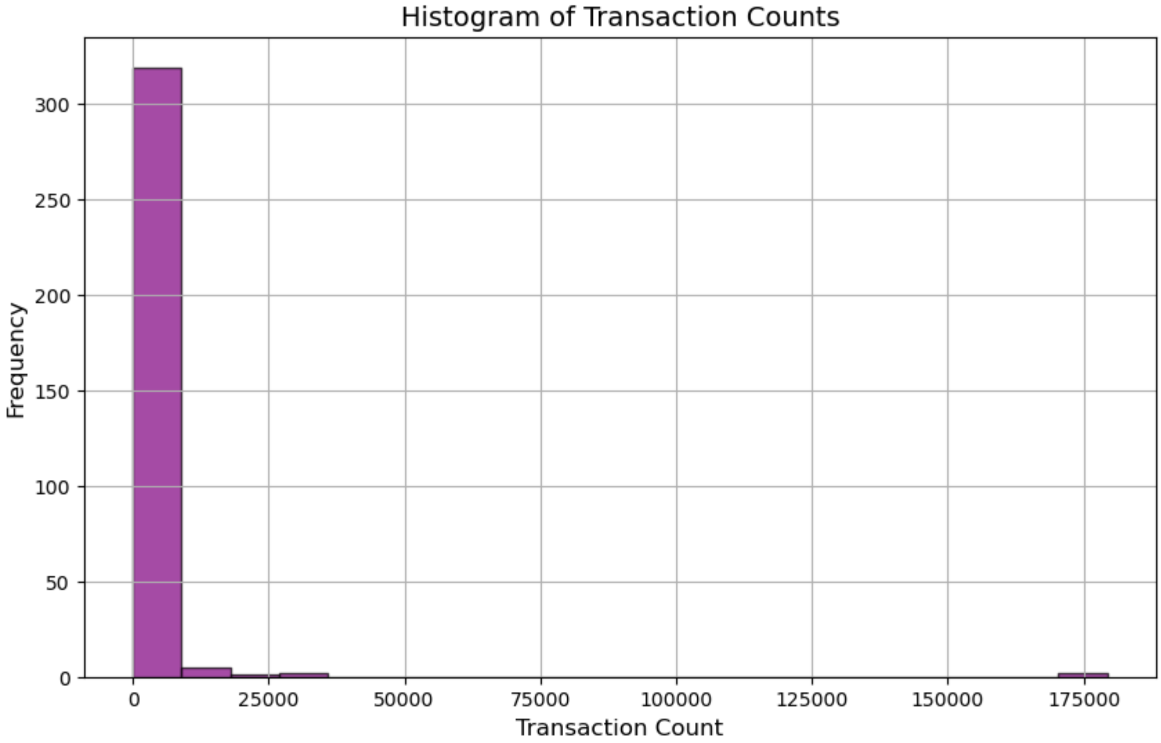
```python
monthly_txn = df.groupby('month')['txn_count'].sum()
plt.figure(figsize=(10, 5))
sns.barplot(x=monthly_txn.index, y=monthly_txn.values, palette='coolwarm')
plt.title("Monthly Transaction Distribution", fontsize=14)
plt.xlabel("Month", fontsize=12)
plt.ylabel("Total Transactions", fontsize=12)
plt.show()
```

```
C:\Users\91998\AppData\Local\Temp\ipykernel_3608\536352585.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.barplot(x=monthly_txn.index, y=monthly_txn.values, palette='coolwarm')
```



This Bar chart show the Monthly Transaction Distribution in our dataset

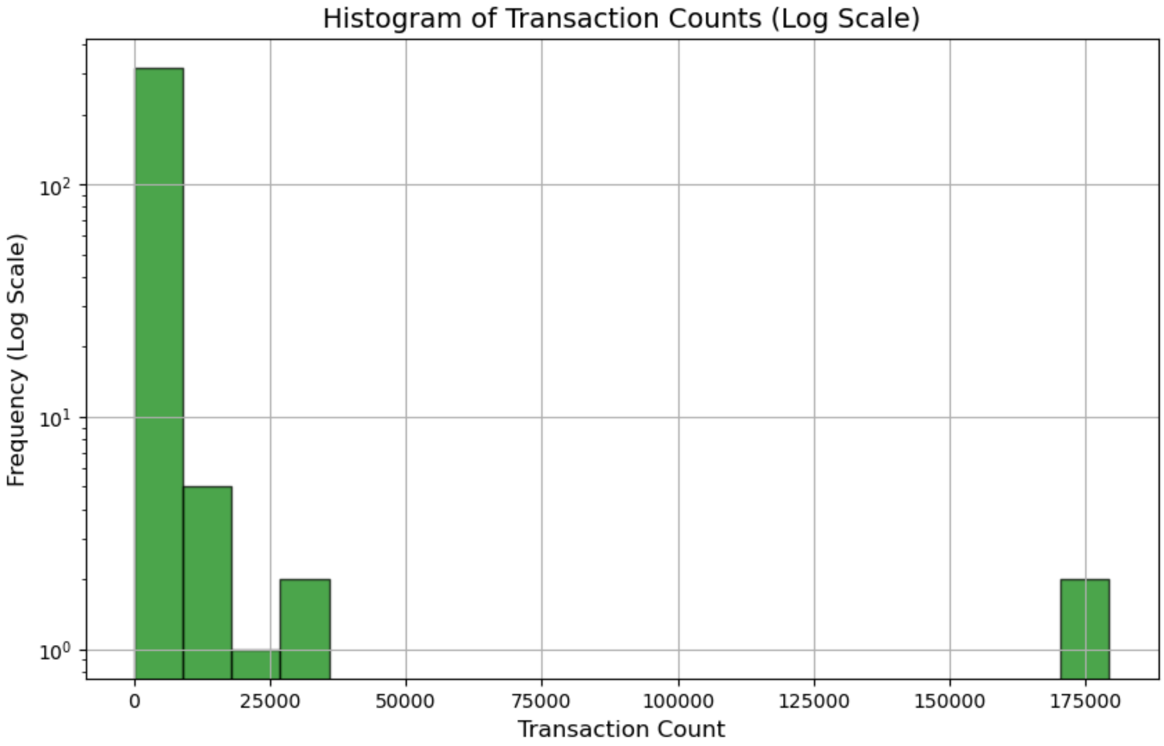
```
In [33]: plt.figure(figsize=(10, 6))
plt.hist(df['txn_count'], bins=20, color='purple', edgecolor='black', alpha=0.7)
plt.title("Histogram of Transaction Counts", fontsize=14)
plt.xlabel("Transaction Count", fontsize=12)
plt.ylabel("Frequency", fontsize=12)
plt.grid(True)
plt.show()
```



This Histogram shows the Transaction Counts in the dataset

In [34]:

```
plt.figure(figsize=(10, 6))
plt.hist(df['txn_count'], bins=20, color='green', edgecolor='black', alpha=0.7,
plt.title("Histogram of Transaction Counts (Log Scale)", fontsize=14)
plt.xlabel("Transaction Count", fontsize=12)
plt.ylabel("Frequency (Log Scale)", fontsize=12)
plt.grid(True)
plt.show()
```



This Histogram show the Transaction Counts (Log Scale) in the dataset

In [35]:

```
state_1 = df[df['salestatename'] == 'UTTAR PRADESH']['txn_count']
state_2 = df[df['salestatename'] == 'DELHI']['txn_count']
t_stat, p_value = stats.ttest_ind(state_1, state_2, equal_var=False)
print(f"T-Test Results: t-statistic={t_stat}, p-value={p_value}")
```

```
T-Test Results: t-statistic=-1.6243367752719553, p-value=0.118546012769709842
```

T-Test of Comparing uttar pradesh and delhi states for transaction counts

In [36]:

```
state_1 = df[df['salestatename'] == 'JAMMU AND KASHMIR']['txn_count']
state_2 = df[df['salestatename'] == 'DELHI']['txn_count']
t_stat, p_value = stats.ttest_ind(state_1, state_2, equal_var=False)
print(f"T-Test Results: t-statistic={t_stat}, p-value={p_value}")
```

```
T-Test Results: t-statistic=-1.5946813783338056, p-value=0.1250397316816096
```

T-Test of Comparing jammu & kashimar and delhi states for transaction counts

In [37]:

```python
contingency_table = pd.crosstab(df['homestatename'], df['salestatename'])
chi2_stat, p_value, dof, expected = stats.chi2_contingency(contingency_table)
print(f"Chi-Square Test Results: chi2-statistic={chi2_stat}, p-value={p_value}")
```

```
Chi-Square Test Results: chi2-statistic=296.2119182699737, p-value=1.0
```

This the P-Test Chi-Square test for the dataset

In [38]:

```python
def categorize_transactions(count):
    if count > 1000:
        return "High"
    elif count > 100:
        return "Medium"
    else:
        return "Low"

df["transaction_category"] = df["txn_count"].apply(categorize_transactions)
```

In [39]:

```python
df
```

Out[39]:

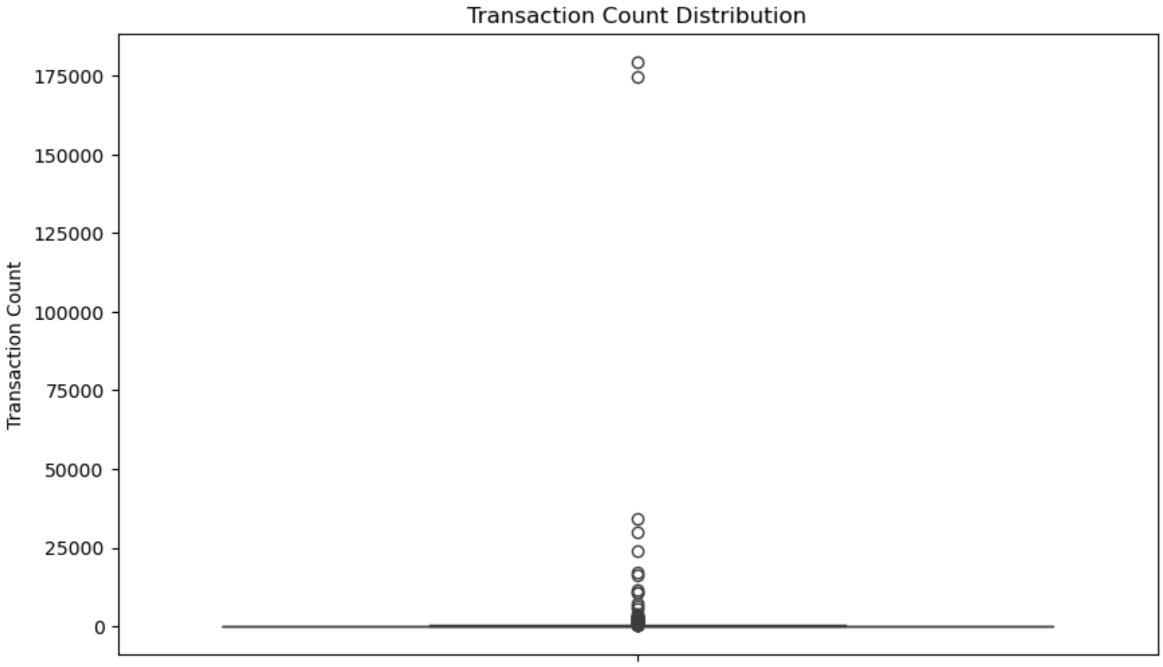
	homestatecode	salestatecode	month	year	txn_count	salestatename	homestate
0	5	1	5	2024	13	JAMMU AND KASHMIR	UTTARAKI
1	6	1	5	2024	43	JAMMU AND KASHMIR	HAR
2	7	1	5	2024	5	JAMMU AND KASHMIR	
3	8	1	5	2024	2	JAMMU AND KASHMIR	RAJAS
4	9	1	5	2024	2438	JAMMU AND KASHMIR	UTTAR PRA
...	...	...	...	...	...	...	
324	24	38	5	2024	813	Dadar & Nagar Haveli & Daman & Diu	GU
325	27	38	5	2024	504	Dadar & Nagar Haveli & Daman & Diu	MAHARAS
326	28	38	5	2024	1	Dadar & Nagar Haveli & Daman & Diu	AN PRA
327	29	38	5	2024	3	Dadar & Nagar Haveli & Daman & Diu	KARN
328	32	38	5	2024	1	Dadar & Nagar Haveli & Daman & Diu	KE

329 rows × 8 columns

Creating a new column transaction_category in the dataset

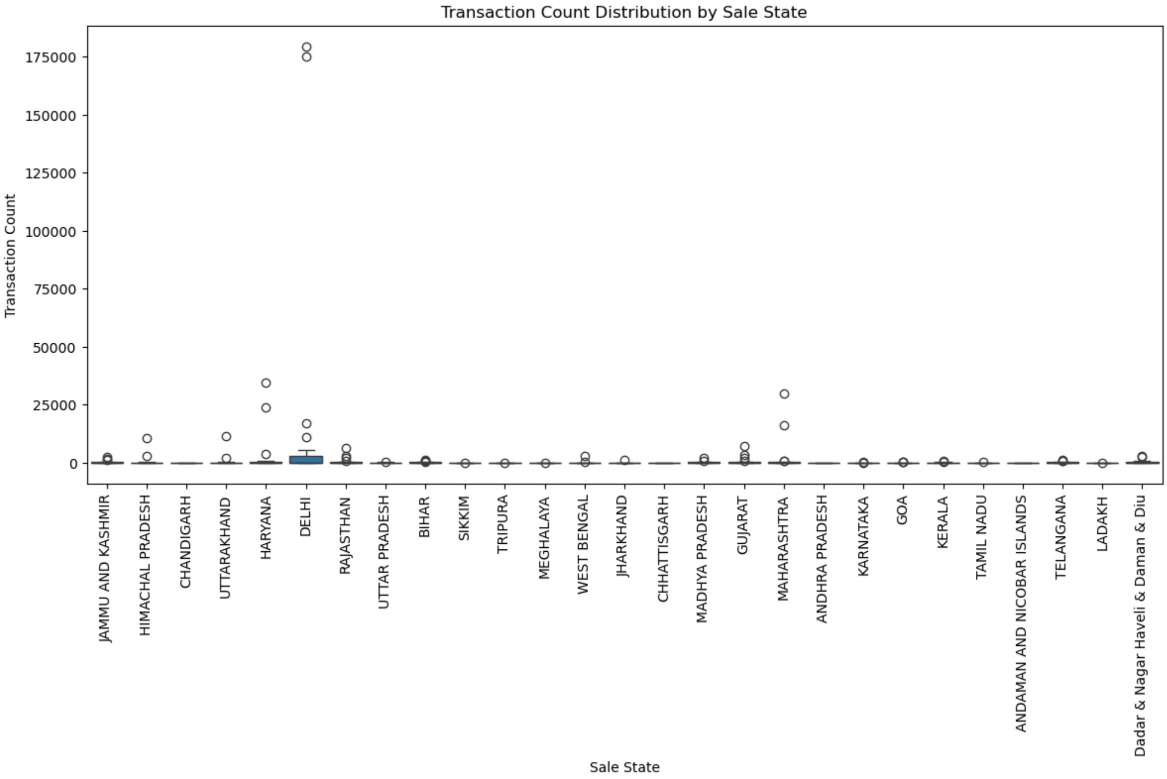
In [40]:

```
plt.figure(figsize=(10, 6))
sns.boxplot(y=df["txn_count"])
plt.title("Transaction Count Distribution")
plt.ylabel("Transaction Count")
plt.show()
```



This box plot show the Transaction Count Distribution in the dataset and highest transaction count is 175000

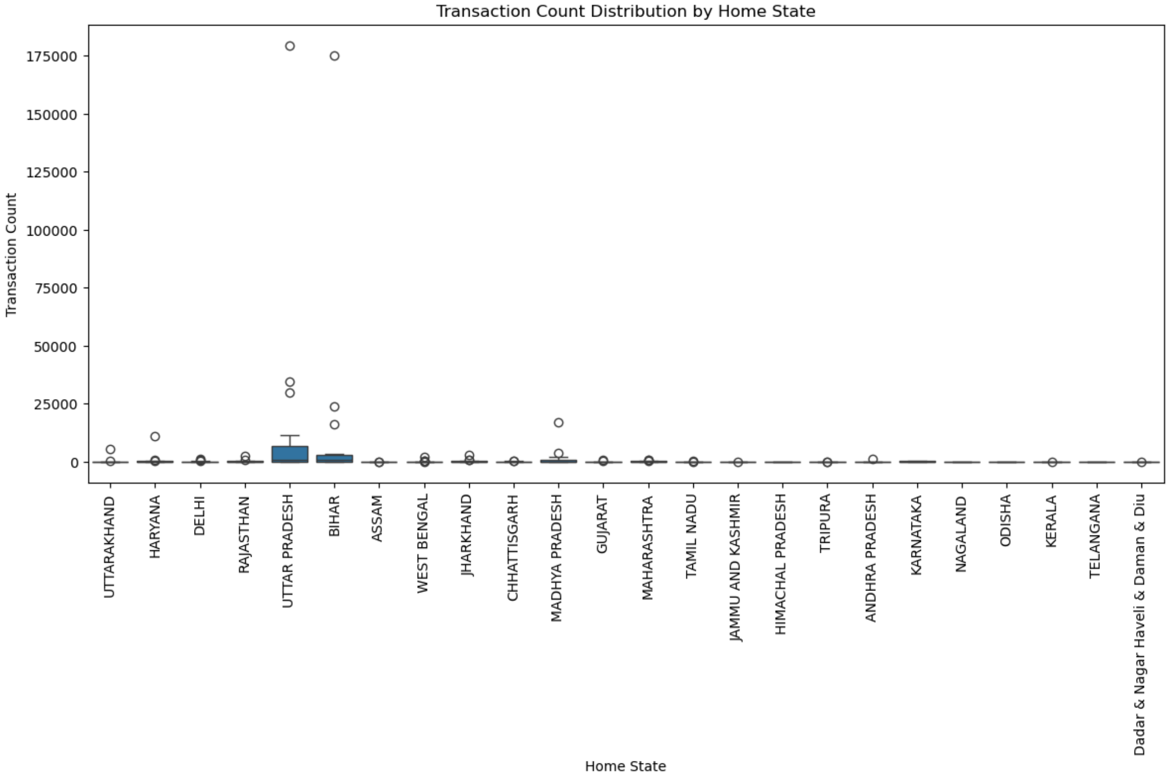
In [41]:
```
plt.figure(figsize=(14, 6))
sns.boxplot(x="salestatename", y="txn_count", data=df)
plt.xticks(rotation=90)
plt.title("Transaction Count Distribution by Sale State")
plt.xlabel("Sale State")
plt.ylabel("Transaction Count")
plt.show()
```



Box Plot by Sale State and delhi is the highest and gujarat is lowest

In [42]:

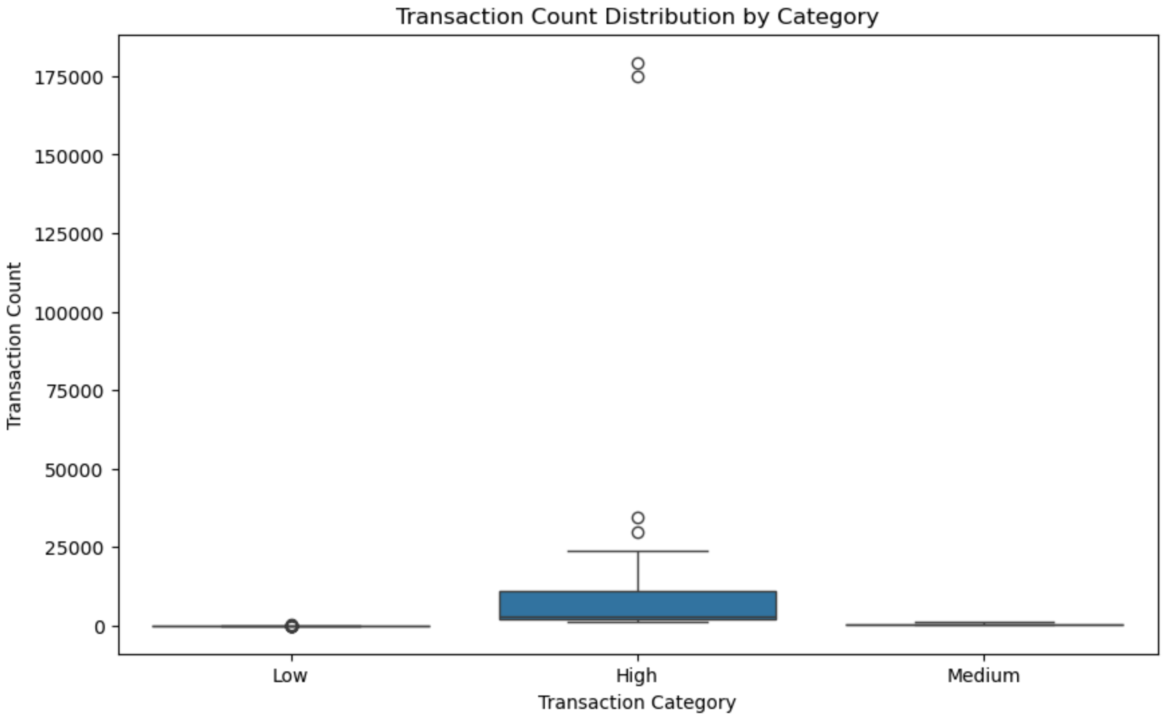
```
plt.figure(figsize=(14, 6))
sns.boxplot(x="homestatename", y="txn_count", data=df)
plt.xticks(rotation=90)
plt.title("Transaction Count Distribution by Home State")
plt.xlabel("Home State")
plt.ylabel("Transaction Count")
plt.show()
```



This Box Plot shows Transaction Count Distribution by Home State and uttar pradesh is highest state

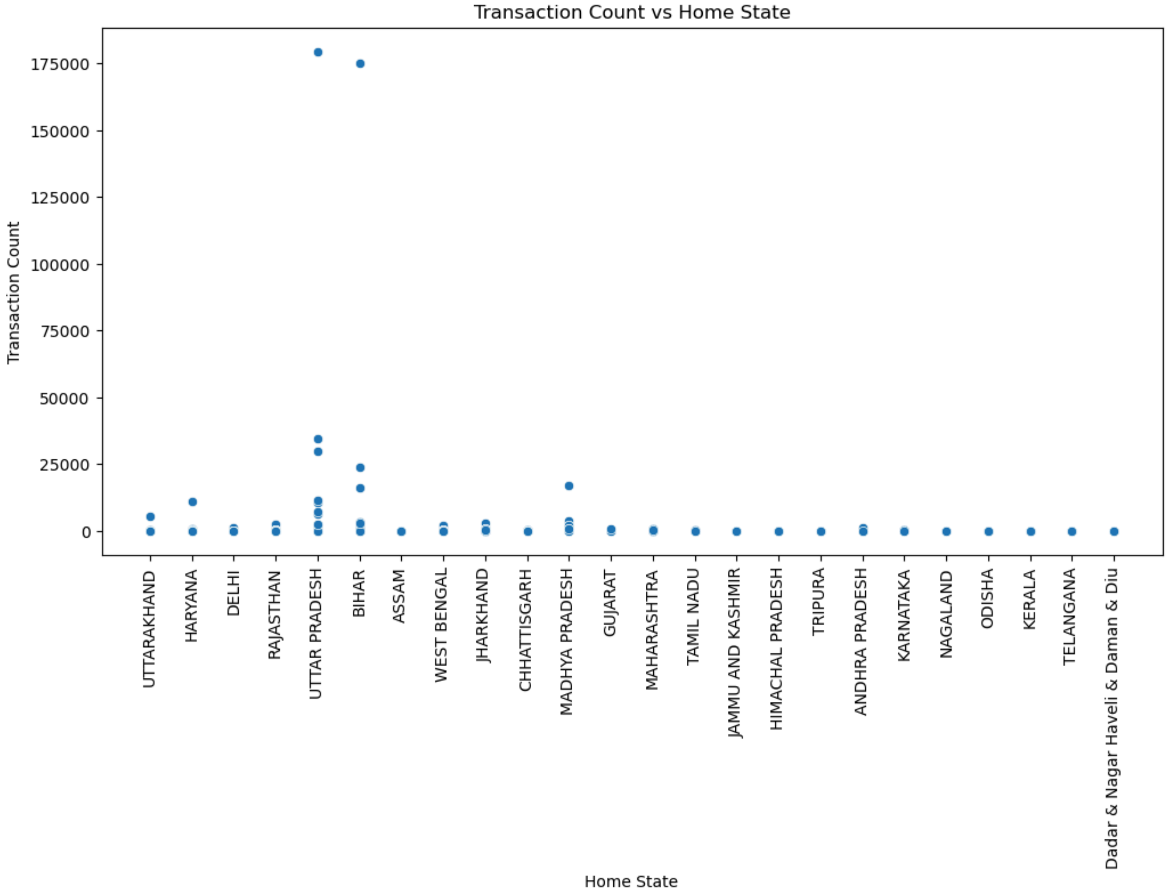
In [43]:

```
plt.figure(figsize=(10, 6))
sns.boxplot(x="transaction_category", y="txn_count", data=df)
plt.title("Transaction Count Distribution by Category")
plt.xlabel("Transaction Category")
plt.ylabel("Transaction Count")
plt.show()
```



Box Plot by Transaction Category and transaction count is highest

In [44]:
```
plt.figure(figsize=(12, 6))
sns.scatterplot(x=df["homestatename"], y=df["txn_count"])
plt.xticks(rotation=90)
plt.title("Transaction Count vs Home State")
plt.xlabel("Home State")
plt.ylabel("Transaction Count")
plt.show()
```



Scatter Plot: Transaction Count vs Home State and bihar is second state

In [45]:

```
state_1 = df[df['salestatename'] == 'UTTAR PRADESH']['txn_count']
state_2 = df[df['salestatename'] == 'DELHI']['txn_count']
t_stat, p_value = stats.ttest_ind(state_1, state_2, equal_var=False)
print(f"T-Test Results: t-statistic={t_stat}, p-value={p_value}")
```

```
T-Test Results: t-statistic=-1.6243367752719553, p-value=0.118546012769709 84
```

T-Test of Comparing uttar pradesh and delhi states for transaction counts

---

Data Analysis Report

1) Introduction:

This report analyzes transaction data between different Indian states. The dataset consists of 329 rows and 7 columns, including home state, sale state, transaction count, and timestamps (month, year).

2) Data Overview or About data:

Columns: The columns in our dataset is homestatecode, salestatecode, month, year, txn_count, salestatename, homestatename

No missing values: The dataset is complete with no null values and the dataset id very clean

No duplicate entries: Duplicates were removed, maintaining data integrity.

3)Top Transaction States:

Delhi recorded the highest number of transactions

After Delhi we having Uttar Pradesh and Maharashtra followed closely in transaction volume.

4)Transaction Distribution

Pie Chart Analysis:

Delhi accounts for the largest share of transactions.

Rajasthan and Madhya Pradesh have lower transaction counts.

Box Plot Analysis:

Delhi and Gujarat show high in transaction counts

Uttar Pradesh always leads in home state transactions

Histogram Analysis:

The histogram shows the distribution of transaction volumes across different states

Delhi, Uttar Pradesh, and Maharashtra show higher transaction volumes, while smaller states like Goa and Daman & Diu have much lower counts.

Bar Chart Analysis:

Delhi, Uttar Pradesh, and Maharashtra lead in total transactions

Smaller states like Goa and Daman & Diu have significantly lower transactions

5) Statistical Analysis:

T-Test of the transaction differences between states

Uttar Pradesh vs. Delhi:

t-statistic = -1.6243, p-value = 0.1185

Conclusion: No difference in transaction

Jammu & Kashmir vs. Delhi:

t-statistic = -1.5946, p-value = 0.1250

Conclusion: No difference

Chi-Square Test of the state-wise transaction homogeneity

Chi-square statistic = 296.21, p-value = 1.0

Conclusion: The transaction distribution among states is not random.

6) Visualizations Used in our data analysis:

Bar Charts: Top 10 home and sale states by transaction volume.

Pie Charts: Distribution of transactions by state.

Comparison between states like Delhi vs. Uttar Pradesh.

Box Plots: Transaction spread across different states.

Scatter Plot: Relationship between home states and transaction volume.

Histograms: Frequency distribution of transaction counts.

7) Conclusion:

Delhi and Uttar Pradesh dominate in terms of transactions.

Although there are variations by state, there was no statistically significant difference between the major states.

```
In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```