CL-I DMV 12

July 20, 2025

[43]:
```
''' NAME:Aher Swami Sandip
ROLL NO.01
COURSE: AI&DS
CLASS: BE
SUB:Computer Laboratory-I (DMV) '''
```

[3]:
```
# 12. Data Aggregation
# Problem Statement: Analyzing Sales Performance by Region in a Retail Company
# Dataset: "Retail_Sales_Data.csv"
# Description: The dataset contains information about sales transactions in a␣
 ↪retail company. It
# includes attributes such as transaction date, product category, quantity␣
 ↪sold, and sales
# amount. The goal is to perform data aggregation to analyze the sales␣
 ↪performance by region
# and identify the top-performing regions.
# Tasks to Perform:
```

[4]:
```
# 1. Import the "Retail_Sales_Data.csv" dataset.
```

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
df= pd.read_csv("Retail_Sales_Data.csv")
```

[3]:
```
df
```

[3]:

	Transaction ID	Date	Customer ID	Gender	Age	Region
0	1	24-11-2023	CUST001	Male	34	usa
1	2	27-02-2023	CUST002	Female	26	india
2	3	13-01-2023	CUST003	Male	50	pak
3	4	21-05-2023	CUST004	Male	37	usa
4	5	06-05-2023	CUST005	Male	30	india
..	…	…	…	…	…	
995	996	16-05-2023	CUST996	Male	62	pak
996	997	17-11-2023	CUST997	Male	52	usa
997	998	29-10-2023	CUST998	Female	23	india

```
998             999  05-12-2023     CUST999  Female   36    pak
999            1000  12-04-2023    CUST1000    Male   47    usa

    Product Category  Quantity  Price per Unit  Sales Amount
0             Beauty         3              50           150
1           Clothing         2             500          1000
2        Electronics         1              30            30
3           Clothing         1             500           500
4             Beauty         2              50           100
..               …       …             …           …
995         Clothing         1              50            50
996           Beauty         3              30            90
997           Beauty         4              25           100
998      Electronics         3              50           150
999      Electronics         4              30           120

[1000 rows x 10 columns]
```

[9]:
```
# 2. Explore the dataset to understand its structure and content.
```

[7]:
```
df.info
```

[7]:
```
<bound method DataFrame.info of      Transaction ID        Date Customer ID
Gender  Age Region  \
0                 1  24-11-2023     CUST001    Male   34    usa
1                 2  27-02-2023     CUST002  Female   26  india
2                 3  13-01-2023     CUST003    Male   50    pak
3                 4  21-05-2023     CUST004    Male   37    usa
4                 5  06-05-2023     CUST005    Male   30  india
..              …         …         …     …  …    …
995             996  16-05-2023     CUST996    Male   62    pak
996             997  17-11-2023     CUST997    Male   52    usa
997             998  29-10-2023     CUST998  Female   23  india
998             999  05-12-2023     CUST999  Female   36    pak
999            1000  12-04-2023    CUST1000    Male   47    usa

    Product Category  Quantity  Price per Unit  Sales Amount
0             Beauty         3              50           150
1           Clothing         2             500          1000
2        Electronics         1              30            30
3           Clothing         1             500           500
4             Beauty         2              50           100
..               …       …             …           …
995         Clothing         1              50            50
996           Beauty         3              30            90
997           Beauty         4              25           100
998      Electronics         3              50           150
```

```
999      Electronics         4              30           120

[1000 rows x 10 columns]>
```

[9]:
```
df.describe()
```

[9]:
```
       Transaction ID         Age     Quantity  Price per Unit  Sales Amount
count     1000.000000  1000.00000  1000.000000     1000.000000   1000.000000
mean       500.500000    41.39200     2.514000      179.890000    456.000000
std        288.819436    13.68143     1.132734      189.681356    559.997632
min          1.000000    18.00000     1.000000       25.000000     25.000000
25%        250.750000    29.00000     1.000000       30.000000     60.000000
50%        500.500000    42.00000     3.000000       50.000000    135.000000
75%        750.250000    53.00000     4.000000      300.000000    900.000000
max       1000.000000    64.00000     4.000000      500.000000   2000.000000
```

[13]:
```
df.shape
```

[13]:
```
(1000, 10)
```

[15]:
```
df.columns
```

[15]:
```
Index(['Transaction ID', 'Date', 'Customer ID', 'Gender', 'Age', 'Region',
       'Product Category', 'Quantity', 'Price per Unit', 'Sales Amount'],
      dtype='object')
```

[17]:
```
df.head()
```

[17]:
```
   Transaction ID        Date Customer ID  Gender  Age Region  \
0               1  24-11-2023     CUST001    Male   34    usa
1               2  27-02-2023     CUST002  Female   26  india
2               3  13-01-2023     CUST003    Male   50    pak
3               4  21-05-2023     CUST004    Male   37    usa
4               5  06-05-2023     CUST005    Male   30  india

  Product Category  Quantity  Price per Unit  Sales Amount
0           Beauty         3              50           150
1         Clothing         2             500          1000
2      Electronics         1              30            30
3         Clothing         1             500           500
4           Beauty         2              50           100
```

[19]:
```
df.dtypes
```

[19]:
```
Transaction ID       int64
Date                object
Customer ID         object
Gender              object
```

```
Age                  int64
Region              object
Product Category    object
Quantity             int64
Price per Unit       int64
Sales Amount         int64
dtype: object
```

[21]:
```
df.isna().sum()
```

[21]:
```
Transaction ID      0
Date                0
Customer ID         0
Gender              0
Age                 0
Region              0
Product Category    0
Quantity            0
Price per Unit      0
Sales Amount        0
dtype: int64
```

[23]:
```
# 3. Identify the relevant variables for aggregating sales data, such as␣
↪region, sales amount, and product category.
```

[25]:
```
region_sales = df.groupby('Region')['Sales Amount'].sum()
region_sales
```

[25]:
```
Region
india    154360
pak      136975
usa      164665
Name: Sales Amount, dtype: int64
```

[27]:
```
category_sales = df.groupby('Product Category')['Sales Amount'].sum()
category_sales
```

[27]:
```
Product Category
Beauty         143515
Clothing       155580
Electronics    156905
Name: Sales Amount, dtype: int64
```

[29]:
```
region_category_sales = df.groupby(['Region', 'Product Category'])['Sales␣
↪Amount'].sum()
region_category_sales
```

[29]:
```
Region  Product Category
india   Beauty              47525
        Clothing            52855
        Electronics         53980
pak     Beauty              45935
        Clothing            45915
        Electronics         45125
usa     Beauty              50055
        Clothing            56810
        Electronics         57800
Name: Sales Amount, dtype: int64
```

[31]:
```
Product_category = df.groupby('Product Category')['Sales Amount'].sum()
```

[33]:
```
Product_category
```

[33]:
```
Product Category
Beauty          143515
Clothing        155580
Electronics     156905
Name: Sales Amount, dtype: int64
```

[35]:
```
# 4. Group the sales data by region and calculate the total sales amount for␣
 ↪each region.
```

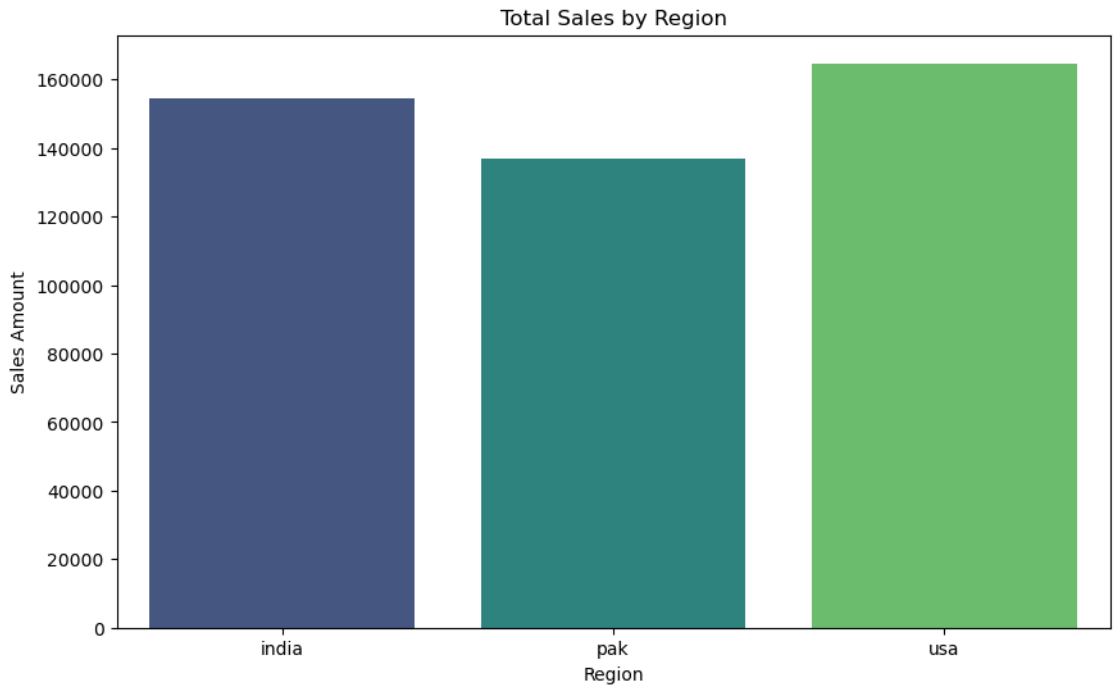
[37]:
```
# Group by region and calculate the total sales amount for each region
total_sales_by_region = df.groupby('Region')['Sales Amount'].sum().reset_index()


# Print the result
print(total_sales_by_region)
```

```
  Region  Sales Amount
0  india        154360
1    pak        136975
2    usa        164665
```

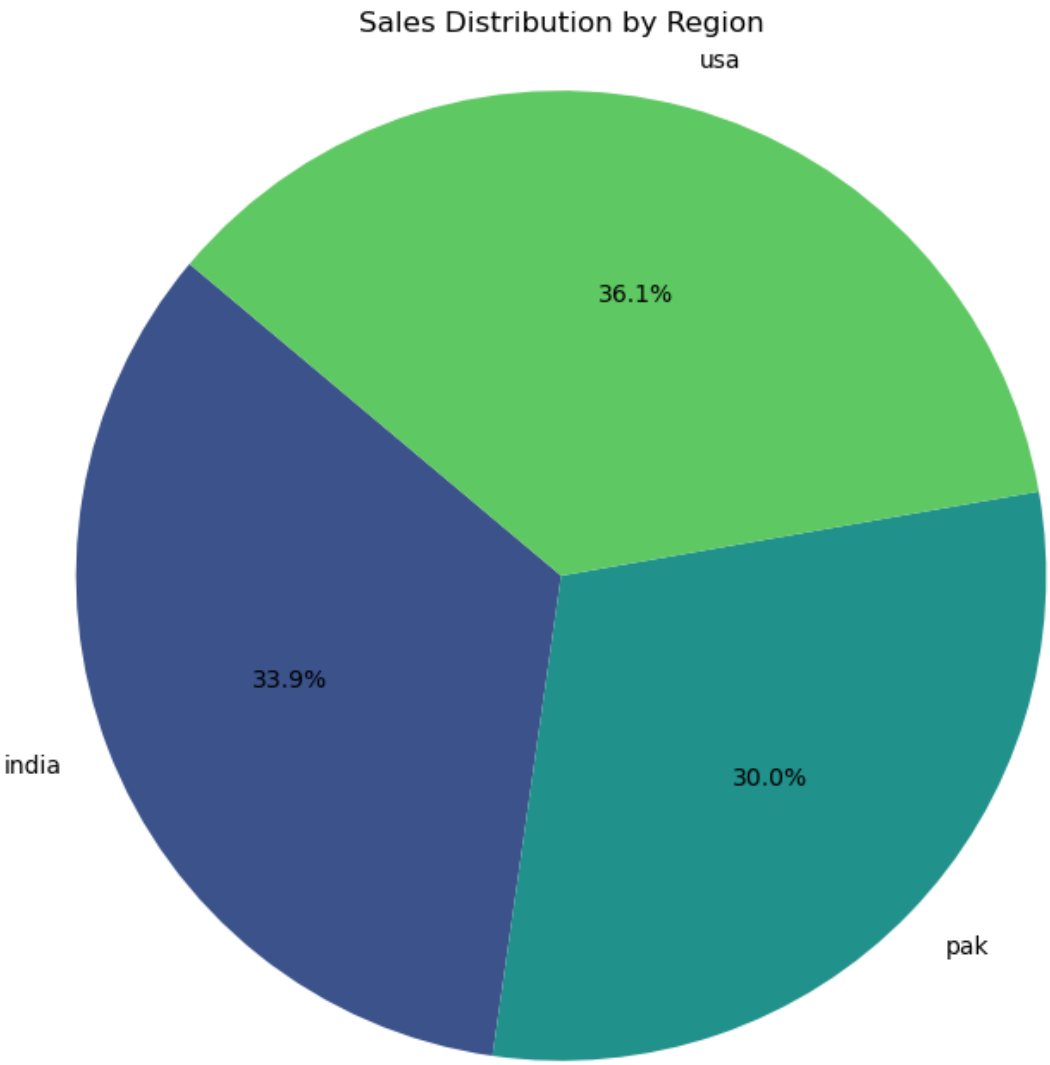
[39]:
```
# 5. Create bar plots or pie charts to visualize the sales distribution by␣
 ↪region.
```

[41]:
```
# Bar Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='Region', y='Sales Amount', data=total_sales_by_region,␣
 ↪palette='viridis')
plt.title('Total Sales by Region')
plt.xlabel('Region')
plt.ylabel('Sales Amount')
plt.show()
```

```
C:\Users\Dell\AppData\Local\Temp\ipykernel_19464\342875029.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x='Region', y='Sales Amount', data=total_sales_by_region,
palette='viridis')
```



[43]:
```
# Pie Chart
plt.figure(figsize=(8, 8))
plt.pie(total_sales_by_region['Sales Amount'],␣
  ↪labels=total_sales_by_region['Region'], autopct='%1.1f%%', startangle=140,␣
  ↪colors=sns.color_palette('viridis', len(total_sales_by_region)))
plt.title('Sales Distribution by Region')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```



[45]:
```
# 6. Identify the top-performing regions based on the highest sales amount.
```

[47]:
```
# Group by region and calculate the total sales amount for each region
total_sales_by_region = df.groupby('Region')['Sales Amount'].sum().reset_index()

# Rename columns for clarity
total_sales_by_region.columns = ['Region', 'Total Sales']

# Sort the DataFrame by Total Sales in descending order
sorted_sales_by_region = total_sales_by_region.sort_values(by='Total Sales',␣
↪ascending=False)
```

```
# Print the sorted DataFrame
print(sorted_sales_by_region)
```

```
  Region  Total Sales
2    usa       164665
0  india       154360
1    pak       136975
```

[49]:
```
# 7. Group the sales data by region and product category to calculate the total␣
 ↪sales amount for each combination.
```

[51]:
```
# Group by region and product category and calculate the total sales amount for␣
 ↪each combination
total_sales_by_region_category = df.groupby(['Region', 'Product␣
 ↪Category'])['Sales Amount'].sum().reset_index()

# Rename columns for clarity
total_sales_by_region_category.columns = ['Region', 'Product Category', 'Total␣
 ↪Sales']

# Print the result
print(total_sales_by_region_category)
```

```
  Region Product Category  Total Sales
0  india           Beauty        47525
1  india         Clothing        52855
2  india      Electronics        53980
3    pak           Beauty        45935
4    pak         Clothing        45915
5    pak      Electronics        45125
6    usa           Beauty        50055
7    usa         Clothing        56810
8    usa      Electronics        57800
```

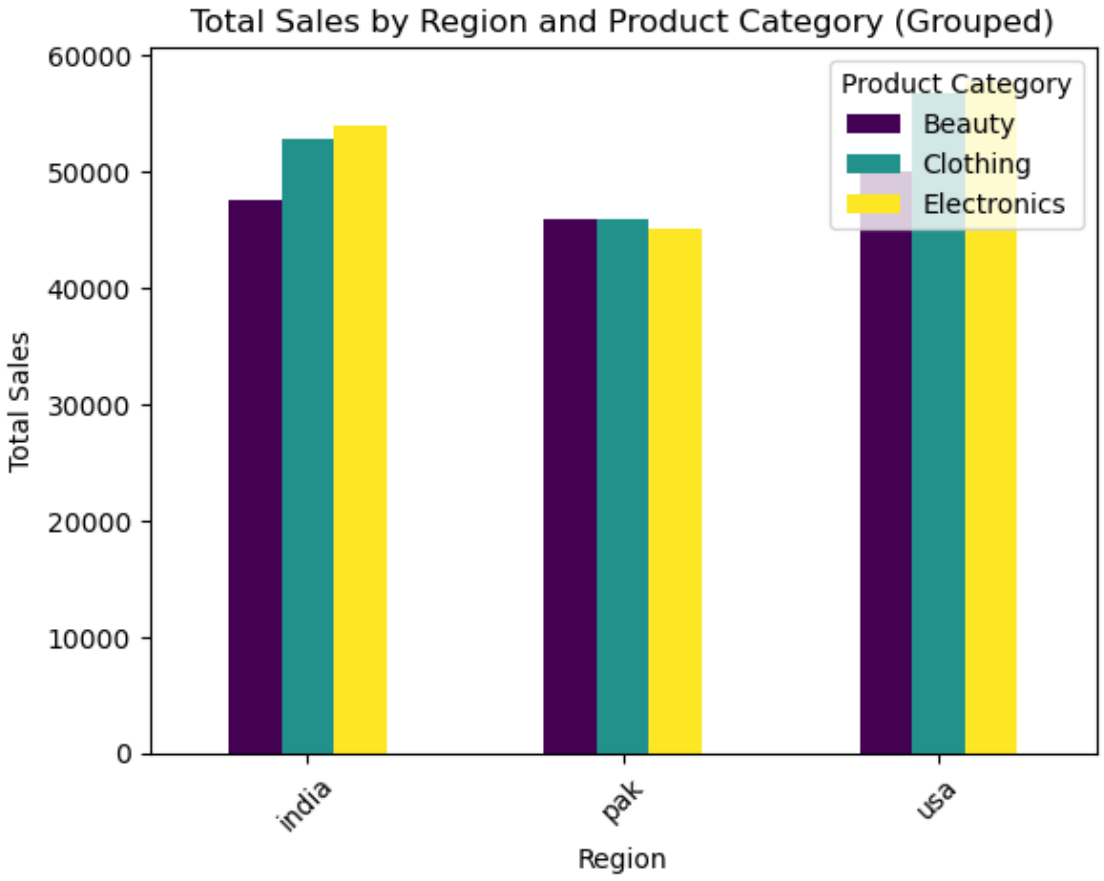
[53]:
```
# 8. Create stacked bar plots or grouped bar plots to compare the sales amounts␣
 ↪across  different regions and product categories.
```

[55]:
```
# Group by region and product category and calculate the total sales amount for␣
 ↪each combination
total_sales_by_region_category = df.groupby(['Region', 'Product␣
 ↪Category'])['Sales Amount'].sum().unstack().fillna(0)

# Print the result
print(total_sales_by_region_category)

# Create a grouped bar plot
plt.figure(figsize=(14, 8))
```

```
total_sales_by_region_category.plot(kind='bar', stacked=False,␣
 ↪colormap='viridis')
plt.title('Total Sales by Region and Product Category (Grouped)')
plt.xlabel('Region')
plt.ylabel('Total Sales')
plt.legend(title='Product Category')
plt.xticks(rotation=45)
plt.show()
```

```
Product Category  Beauty  Clothing  Electronics
Region
india              47525     52855        53980
pak                45935     45915        45125
usa                50055     56810        57800
```

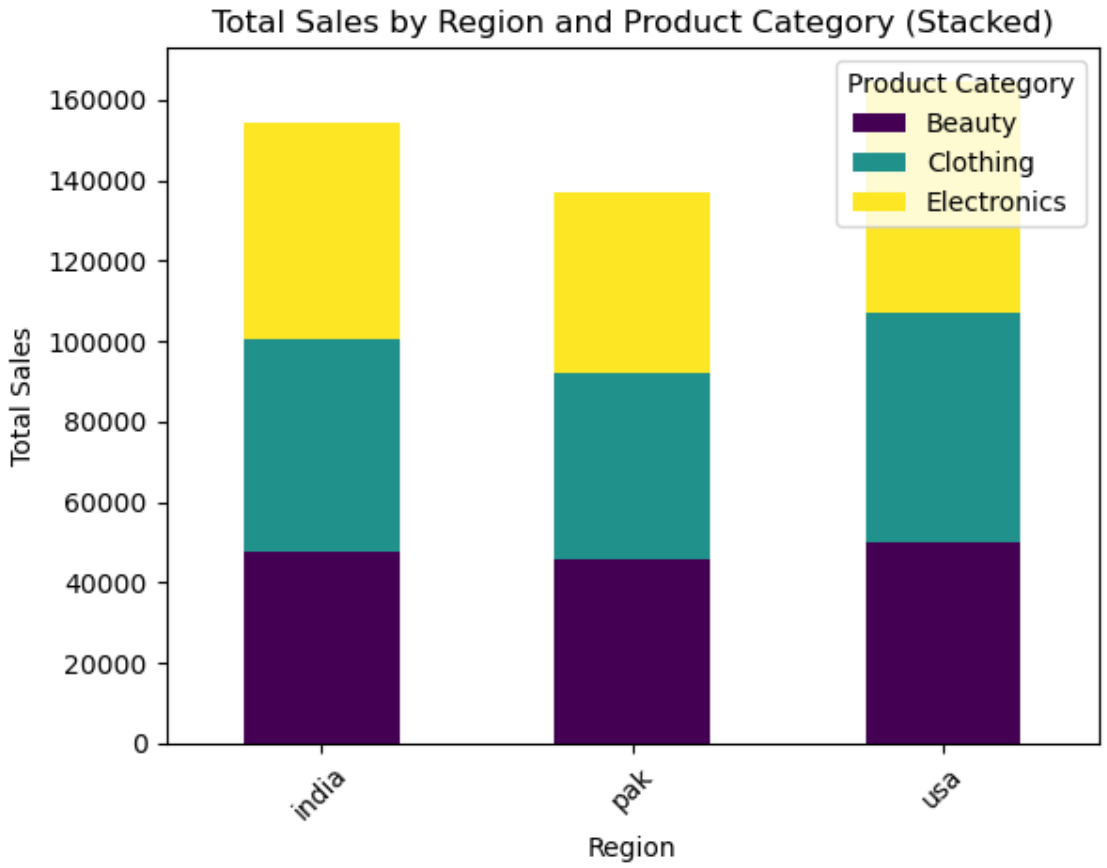
```
<Figure size 1400x800 with 0 Axes>
```



[42]:
```
# Create a stacked bar plot
plt.figure(figsize=(14, 8))
```

```
total_sales_by_region_category.plot(kind='bar', stacked=True,␣
 ↪colormap='viridis')
plt.title('Total Sales by Region and Product Category (Stacked)')
plt.xlabel('Region')
plt.ylabel('Total Sales')
plt.legend(title='Product Category')
plt.xticks(rotation=45)
plt.show()
```

```
<Figure size 1400x800 with 0 Axes>
```



[ ]: