DATA VISUALIZATION

ASSIGNMENT

NAME:- Anisetti Sri Vaishnavi

VTU No:- VTU 24209

SLOT:- Dr. Kausalya. K (S2 L5)

Banking Transaction Dataset

* Sample Dataset

Customer_ID	Name	Age	Acc_type	T_ID	Date	Amo	Balanc
C001	Rahul Sharma	28	Savings	T001	25-09-01	2500	18000
C002	Priya Singh	35	Current	T002	25-9-2	10000	50000
C003	Aarav Mehta	30	Savings	T003	25-9-3	4500	25000
C004	Neha Patel	40	Current	T004	25-9-4	20000	100000
C005	Riya Gupta	25	Savings	T005	25-9-5	1500	12000
C006	Karan Verma	45	Savings	T006	25-9-6	5000	42000
C007	Meena Iyer	32	Current	T007	25-9-7	6000	35000

Q1) Show the following details using the given dataset

a) Highest Transaction amount, lowest transaction amount

We need to find -

Maximum value of Amo → Higst

Minimum Value of Amo → lowest

```
import pandas as pd
df = pd.read_csv("banking_dataset.csv")
highest = df['Amo'].max()
lowest = df['Amo'].min()
Print("Highest Transaction Amount:", highest)
Print("Lowest Transaction Amount:", lowest)
```

(b) Count of Customers who made more than 10 transaction in a month

We group the dataset by Customer_ID and month, the number of Transactions. Finally Filter those having more than 10 transactions.

```
df['month'] = pd.to_datetime(df['Date']).dt.month
transaction_count = df.groupby(['Customer_ID', 'Month'])
                    .size().reset_index(name='Transaction_
                    Count')
active_Customers = transaction_count[transaction_count['Tran-
                   Saction_Count']>10
Print("Customers with more than 10 transactions in a month:")
Print(active_customers)
```

(c) Display All Customers Names who Have savings Account type.

- We need to filter all records where Acc_type = 'Savings' and show Unique Customer Name

```
Savings_cust = df[df['Acc_Type']=='Savings']['Name'].unique
Print("Customer having Savings Account:")
for name in Savings_customers:
        Print(name).
```

OUTPUT:-

Highest Amount: 20000
lowest Amount: 1500

Customer with more than 10 transactions: Non in this Sample

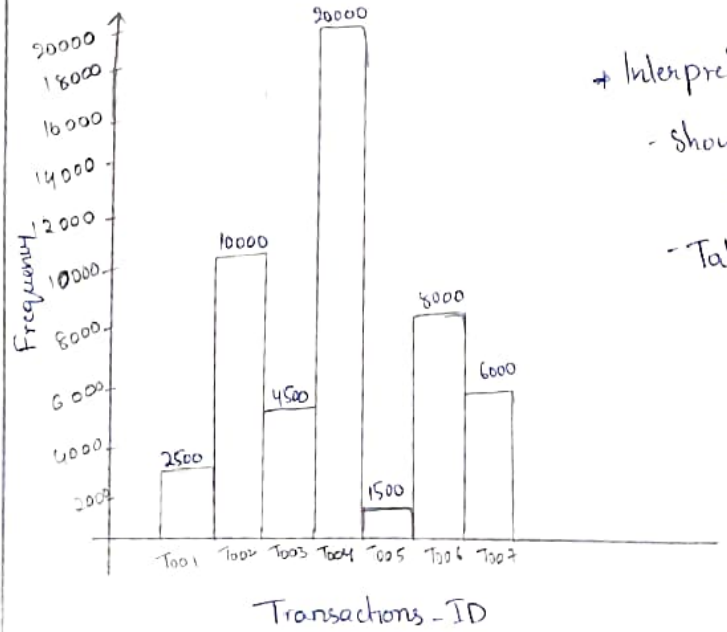
Customers having Savings:
Rahul Sharma
Naran Mehta
Riya Gupta
Karan Verma

Q2) Construct a Histogram for Univariate Analysis, Scatter plot for Bivariate (Transaction Vs Balance), and pairplot for Multivariate (Transaction, Balance, Age).

a) Histogram for Univariate Analysis (Transaction Amount)

- Purpose is to observe the distribution of Transaction amount, detect skewness or outliers.

```
plt.figure(figsize=(7,5))
plt.hist(df['Amo'], bins=5, colour='skyblue', edgecolor='black')
plt.title('Histogram of Transaction Amounts')
plt.xlabel('Transaction Amount')
plt.ylabel('Frequency')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show().
```



→ Interpretation

- shows how amount Spread
- Taller bars, more Amount

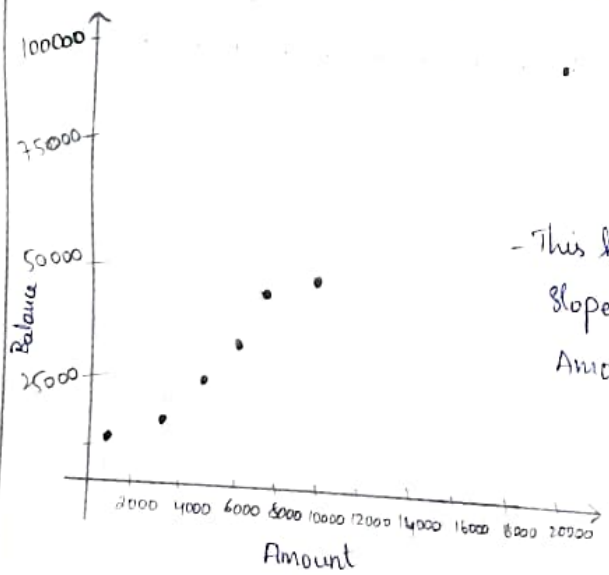
(b) Scatterplot for Bivariate Analysis (Transaction Vs Bal

- Purpose : to visualize the relationship between Amo Vs B
  ∴ Identify positive / Negative Correlation

- Code :-

```
plt.figure(figsize=(7,5))
sns.scatterplot(x='Amount', y='Balance',
                data=df, s=100, colour='teal')
plt.title("Scatterplot: Transaction Vs Balance")
plt.xlabel('Transaction Amount')
plt.ylabel('Balance Amount')
plt.grid(True, linestyle='--', alpha=0.6)
plt.show()
```

Scatter plot: Transaction Vs Balance



- This has a positive trend
  Slope upward, higher Amount, higher balance

(c) Pairplot for Multivariate Analysis (Transaction, Balance, Age)

-Purpose:- Analyze Multiple relationships simultaneously among numeric variables. Helps detect patterns or clusters

-code:-

```
Sns.pairplot(df[['Amo', 'Balance', 'Age']], diag_kind='kde',
             corner=True)
plt.Suptitle('Pairplot of Transaction, Amount, Balance and Age',
             y=1.02)
plt.show()
```

Q3) Utilize WordCloud package to generate a cloud of Transaction remarks / comments and visualize frequency of transaction types (deposit / withdrawal).

*Sample dataset

Trans_type:-	Deposit	Withdrawal	Deposit	Withdrawal	Deposit	Deposit	Withdrawal
Trans_Remark:-	Salary Credited	Bill Payment	Money transfer to frnd	Shopping Expense	Mobil recharge	Interest Credited	Online purchase

-Word cloud:- Visualize most frequent words from transaction type (Deposit / withdrawal)

Code:-

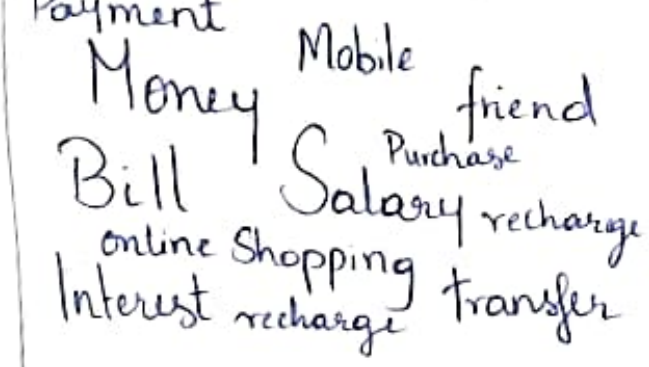
```
text = " ".join(remark for remark in df['Trans_Remark'])
WordCloud = WordCloud(width=800, height=400, background_
            color='white').generate(text)
plt.figure(figsize=(10,5))
plt.imshow(WordCloud, interpolation='bilinear')
```

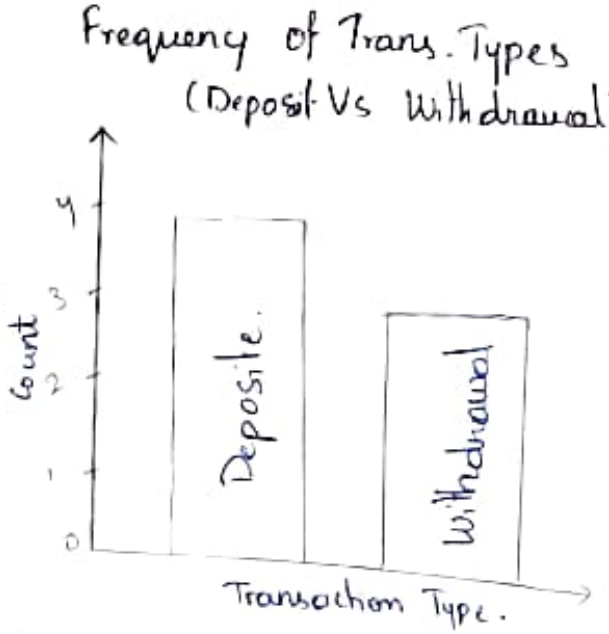
```
plt.axis("off")
plt.title('Word cloud of Transaction Remarks', fontsize=16)
plt.show()
```

- Countplot / Barchart :- to get the Frequency of Transaction

```
plt.figure(figsize=(6,4))
sns.countplot(x='Transaction_type', data=df, palette='Set2')
plt.title('Frequency of Transaction Types (Deposit Vs Withdrawal)')
plt.xlabel('Transaction Type')
plt.ylabel('count')
plt.show()
```

Word cloud





Q4) Build a Geospatial heatmap showing branch locations and transaction density across cities

- Geospatial Visualization allows us to represent data with a geographic components
- Heatmaps visually encode magnitude or density using color intensity.

* Extended Smaple Dataset

Branch_ID	Name	City	Lat	Long	Total_Trans	Total amount
B001	Central Branch	Mumbai	19.076	72.877	120	250000
B002	South Brach	Chennai	13.082	80.270	90	150000
B003	East Branch	Kolkata	22.572	88.363	110	200000
B004	North Branch	Delhi	28.613	77.209	150	300000
B005	West Branch	Bengaluru	12.971	77.594	100	220000

- Code :-

```
m = folium.Map(location = [22.5, 78.9], zoom_start = 5, tiles =
    'cartoDB positron')

for i, row in df.iterrows():
    folium.Marker(location = [row['latitude'], row['longitude']]
                  Popup = f"{row['Branch_Name']}({row['City']
                  }) In Transaction : {row['Total_Transactions']}
                  In Amount : ₹{row['Total_Amount']}",
    icon = folium.Icon(color = 'blue', icon = 'bank', prefix = 'fa')).add
    _to(m)

heat_data = [[row['latitude'], row['latitude'], row['Total_Amount']
             for index, row in df.iterrows()] HeatMap(heat_data
             radius = 25, blur = 15, max_zoom = 6), add_to(m)

m.save("bank_branch_heatmap.html")
```

Q5) Analyze and visualize customer transaction trends over time using line Graph (Monthly deposits Vs withdrawals)

- Time Series Analysis: Helps identify trends, patterns and seasonality in transaction data.
- Line Graph: Ideal for visualizing changes in numerial data over time
- Matplotlib / Seaborn: Used for creating clear, continuous plots of transactions vs time.

Month:	1	2	3	4	5	6	7	8	9	10	11
Deposits:	120000	135000	140000	160000	155000	170000	180000	175000	165000	160000	150000
withdrawals	90000	95000	100000	110000	120000	125000	130000	135000	128000	120000	115000

- Code:-

```
plt.figure(figsize=(10,6))
sns.lineplot(x='Month', y='Deposits', data=df, marker='o',)
sns.lineplot(x='Month', y='Withdrawals', data=df, marker='s',
             label='Withdrawals', color='red')
plt.title('Monthly Deposits Vs Withdrawals Trend', fontSize=14)
plt.xlabel('Month')
plt.ylabel('Transaction Amount (₹)')
plt.legend()
plt.grid(True, linestyle='--', alpha=0.6)
plt.tight_layout()
plt.show()
```