UNIVERSITY COLLEGE OF ENGINEERING

TINDIVANAM

(A Constituent College of Anna University, Chennai)

B.TECH SIXTH SEMESTER
RECORD FOR

DATA ANALYTICS (NM1069)

DEPARTMENT OF INFORMATION TECHNOLOGY

LABORATORY RECORD NOTE BOOK

2024-2025

UNIVERSITY COLLEGE OF ENGINEERING

TINDIVANAM

(A Constituent College of Anna University, Chennai)

DEPARTMENT OF INFORMATION TECHNOLOGY

LABORATORY RECORD NOTE BOOK

2024-2025

This is to certify that is a bonafide record of the work done by Mr./Ms. _____ Register Number _____ Of **3rd Year** B.Tech, Department of **Information Technology** in the **Data Analytics (NM1069)** in the VI Semester.

University **Examination held on .....**

Staff In-Charge **Head of the Department**

Internal Examiner **External Examiner**

INDEX

S.NO	DATE	TOPIC	SIGN
1.		EDA ON GLOBAL SUPERSTORE SALES DATASET	
2.		EDA ON COVID-19 GLOBAL DATASET	
3.		EDA ON YOUTUBE TRENDING VIDEOS DATASET	

EX.No:1	EDA ON GLOBAL SUPERSTORE SALES DATASET

EXPLORATORY DATA ANALYSIS (EDA):

Exploratory Data Analysis (EDA) is the process of examining and understanding a dataset before applying any modeling or predictive techniques. It involves summarizing the dataset's main characteristics using statistical measures and visualizations to uncover patterns, spot anomalies, test hypotheses, and check assumptions. EDA typically includes cleaning the data (handling missing values and duplicates), generating descriptive statistics (like mean, median, and standard deviation), and using plots such as histograms, bar charts, and line graphs to visualize trends and relationships. This step is crucial for gaining insights and making informed decisions about the direction of further analysis or modeling.

DATA SOURCE:

Dataset link: https://www.kaggle.com/datasets/fatihilhan/global-superstore-dataset

STEP 1: LOAD THE DATASET

PROGRAM**:**

```
import pandas as pd

file_path = "/content/GLOBAL DATASTORE.csv"

df = pd.read_csv(file_path)
```

OUTPUT:

```
         Category         City        Country Customer.ID     Customer.Name  \
0  Office Supplies  Los Angeles  United States    LS-172304  Lycoris Saunders
1  Office Supplies  Los Angeles  United States    MV-174854     Mark Van Huff
2  Office Supplies  Los Angeles  United States    CS-121304      Chad Sievert
3  Office Supplies  Los Angeles  United States    CS-121304      Chad Sievert
4  Office Supplies  Los Angeles  United States    AP-109154    Arthur Prichep

   Discount Market  记录数 Order.Date        Order.ID  ... Sales   Segment  \
0       0.0     US     1    00:00.0  CA-2011-130813  ...     19  Consumer
1       0.0     US     1    00:00.0  CA-2011-148614  ...     19  Consumer
2       0.0     US     1    00:00.0  CA-2011-118962  ...     21  Consumer
3       0.0     US     1    00:00.0  CA-2011-118962  ...    111  Consumer
4       0.0     US     1    00:00.0  CA-2011-146969  ...      6  Consumer

  Ship.Date       Ship.Mode  Shipping.Cost       State  Sub.Category  Year  \
0   00:00.0    Second Class           4.37  California         Paper  2011
1   00:00.0  Standard Class           0.94  California         Paper  2011
2   00:00.0  Standard Class           1.81  California         Paper  2011
3   00:00.0  Standard Class           4.59  California         Paper  2011
4   00:00.0  Standard Class           1.32  California         Paper  2011

         Market2 weeknum
0  North America        2
1  North America        4
2  North America       32
3  North America       32
4  North America       40

[5 rows x 27 columns]
```

STEP 2:DATA CLEANING

- Check and remove missing values
- Remove duplicates

PROGRAM:

```
df.dropna(inplace=True)

df.drop_duplicates(inplace=True)
```

STEP 3: SUMMARY STATISTICS

PROGRAM:

```
sales_summary = df["Sales"].describe()[["mean", "50%", "std"]]

profit_summary = df["Profit"].describe()[["mean", "50%", "std"]]

print("Sales Summary:\n", sales_summary)

print("Profit Summary:\n", profit_summary)
```

OUTPUT:

```
Sales Summary:
 mean    246.498440
50%      85.000000
std     487.567175
Name: Sales, dtype: float64
Profit Summary:
 mean     28.610982
50%       9.240000
std     174.340972
Name: Profit, dtype: float64
```

STEP 4: ANALYSIS

Total Sales per Region

PROGRAM:

```
sales_per_region = df.groupby("Region")["Sales"].sum()
print(sales_per_region)
```

OUTPUT:

```
Region
Africa           783776
Canada            66932
Caribbean        324281
Central         2822399
Central Asia     752839
EMEA             806184
East             678834
North           1248192
North Asia       848349
Oceania         1100207
South           1600960
Southeast Asia   884438
West             725514
Name: Sales, dtype: int64
```

Top 5 Most Profitable Product Categories

PROGRAM:

```
top_profitable_categories=df.groupby("Category")["Profit"].sum().nlargest(5)
print(top_profitable_categories)
```

OUTPUT:

Category

Technology 663778.73318

Office Supplies 518473.83430

Furniture 285204.72380

Name: Profit, dtype: float64

Year-wise Sales Trend

PROGRAM:

```
df["Order.Date"] = pd.to_datetime(df["Order.Date"])
df["Year"] = df["Order.Date"].dt.year
yearly_sales = df.groupby("Year")["Sales"].sum()
print(yearly_sales)
```

OUTPUT:

Year

2025 12642905

Name: Sales, dtype: int64

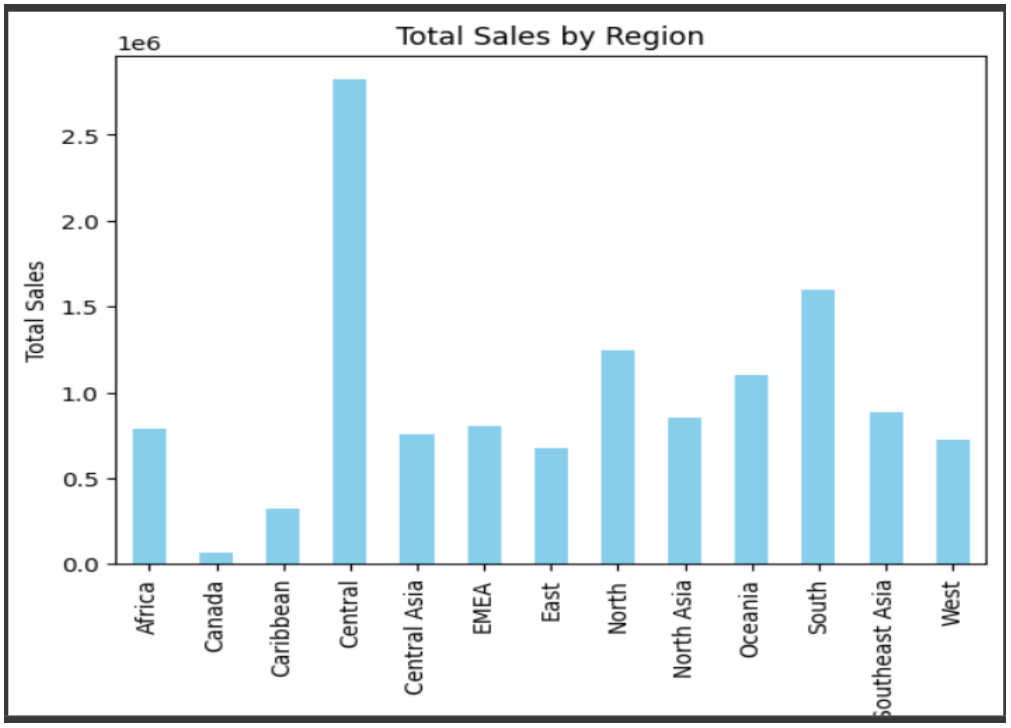
STEP 5: VISUALIZATIONS

Bar Chart: Sales by Region

PROGRAM:

```
import matplotlib.pyplot as plt
sales_per_region.plot(kind="bar", color="skyblue")
plt.title("Total Sales by Region")
plt.xlabel("Region")
plt.ylabel("Total Sales")
plt.show()
```

OUTPUT:

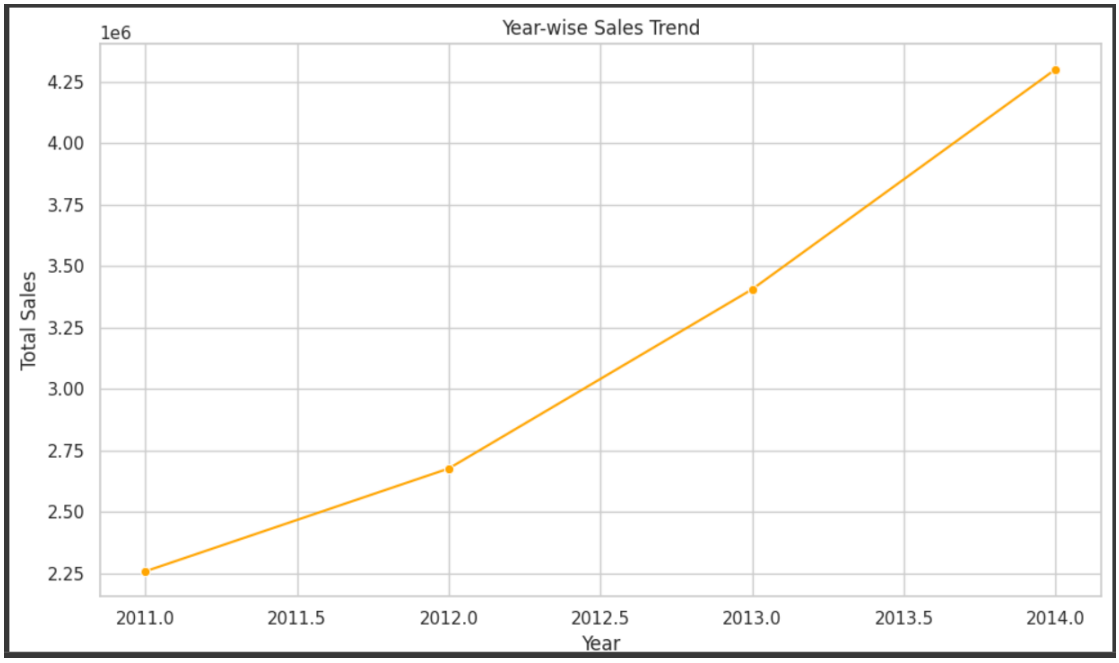


Line Chart: Year-wise Sales Trend

PROGRAM:

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid")
plt.figure(figsize=(10, 6))
yearly_sales = df.groupby('Year')['Sales'].sum()
sns.lineplot(x=yearly_sales.index, y=yearly_sales.values, marker='o', color='orange')
plt.title("Year-wise Sales Trend")
plt.xlabel("Year")
plt.ylabel("Total Sales")
plt.tight_layout()
plt.show()
```

OUTPUT:



GOOGLE COLAB LINK:

https://colab.research.google.com/drive/12ok_SXN84wnqSQL9AzV4OA4e7kDohCWQ?usp=sharing

STEP 6: INSIGHTS

Bar Chart – Sales by Region:

- The West region shows the highest total sales, followed by East and Central.
- South lags behind, indicating potential for growth or marketing focus.

Line Chart – Year-wise Sales Trend:

- Sales have shown a steady upward trend year over year.
- Indicates growing business or improved operations/logistics over time.

Ex.No:2	EDA ON COVID-19 GLOBAL DATASET

INTRODUCTION:

The COVID-19 pandemic, caused by the SARS-CoV-2 virus, has had a profound global impact since early 2020, affecting millions of lives and disrupting economies. To better understand the spread, trends, and regional impact of the virus, data-driven approaches such as Exploratory Data Analysis (EDA) are essential. By exploring confirmed cases, recoveries, and deaths, this analysis aims to uncover insights into the progression of the pandemic, identify the most affected states, and visualize daily trends in new infections.

Dataset link: https://www.kaggle.com/datasets/ COVID-19 in India

GOOGLE COLAB LINK:

https://colab.research.google.com/drive/1VEuFN6gRCyIMnIEkwccqlMqENFi11BRv?usp=sharing

STEP 1: LOAD AND INSPECT THE DATASET

PROGRAM:

```
import pandas as pd
df = pd.read_csv('path_to_covid_dataset.csv')
print(df.head())
```

OUTPUT:

```
   Sno       Date     Time State/UnionTerritory ConfirmedIndianNational  \
0    1  30-01-2020  6:00 PM               Kerala                       1
1    2  31-01-2020  6:00 PM               Kerala                       1
2    3  01-02-2020  6:00 PM               Kerala                       2
3    4  02-02-2020  6:00 PM               Kerala                       3
4    5  03-02-2020  6:00 PM               Kerala                       3

  ConfirmedForeignNational  Cured  Deaths  Confirmed
0                        0      0       0          1
1                        0      0       0          1
2                        0      0       0          2
3                        0      0       0          3
4                        0      0       0          3
```

PROGRAM:

```
print(df.columns)
print(df.info())
```

OUTPUT:

```
Index(['Sno', 'Date', 'Time', 'State/UnionTerritory',
       'ConfirmedIndianNational', 'ConfirmedForeignNational', 'Cured',
       'Deaths', 'Confirmed'],
      dtype='object')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18110 entries, 0 to 18109
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Sno                       18110 non-null  int64
 1   Date                      18110 non-null  object
 2   Time                      18110 non-null  object
 3   State/UnionTerritory      18110 non-null  object
 4   ConfirmedIndianNational   18110 non-null  object
 5   ConfirmedForeignNational  18110 non-null  object
 6   Cured                     18110 non-null  int64
 7   Deaths                    18110 non-null  int64
 8   Confirmed                 18110 non-null  int64
dtypes: int64(4), object(5)
memory usage: 1.2+ MB
None
```

STEP 2: HANDLE MISSING DATA AND CONVERT DATES

PROGRAM:

```
df.fillna(0, inplace=True)
df['Date'] = pd.to_datetime(df['Date'])
```

STEP 3: COMPUTE METRICS

a) Total confirmed, recovered, and death cases per state:

PROGRAM:

```
statewise_total=df.groupby('State/UnionTerritory')[['Confirmed','Cured',
'Deaths']].max().reset_index()
print(statewise_total)
```

OUTPUT:

	State/UnionTerritory	Confirmed	Cured	Deaths
0	Andaman and Nicobar Islands	7548	7412	129
1	Andhra Pradesh	1985182	1952736	13564
2	Arunachal Pradesh	50605	47821	248
3	Assam	576149	559684	5420
4	Bihar	725279	715352	9646
5	Bihar****	715730	701234	9452
6	Cases being reassigned to states	9265	0	0
7	Chandigarh	61992	61150	811
8	Chhattisgarh	1003356	988189	13544
9	Dadra and Nagar Haveli	10377	10261	4
10	Dadra and Nagar Haveli and Daman and Diu	10654	10646	4
11	Daman & Diu	2	0	0
12	Delhi	1436852	1411280	25068
13	Goa	172085	167978	3164
14	Gujarat	825085	814802	10077
15	Haryana	770114	759790	9652
16	Himachal Pradesh	208616	202761	3537
17	Himanchal Pradesh	204516	200040	3507
18	Jammu and Kashmir	322771	317081	4392
19	Jharkhand	347440	342102	5130
20	Karanataka	2885238	2821491	36197
21	Karnataka	2921049	2861499	36848
22	Kerala	3586693	3396184	18004
23	Ladakh	20411	20130	207
24	Lakshadweep	10263	10165	51
25	Madhya Pradesh	791980	781330	10514
26	Madhya Pradesh***	791656	780735	10506
27	Maharashtra	6363442	6159676	134201
28	Maharashtra***	6229596	6000911	130753
29	Manipur	105424	96776	1664
30	Meghalaya	69769	64157	1185
31	Mizoram	46320	33722	171
32	Nagaland	28811	26852	585
33	Odisha	988997	972710	6565

b) State with the highest number of confirmed cases:

PROGRAM:

```
top_state=statewise_total[statewise_total['Confirmed']==
statewise_total['Confirmed'].max()]

print("State with highest confirmed cases:\n", top_state)
```

OUTPUT:

State with highest confirmed cases:

	State/UnionTerritory	Confirmed	Cured	Deaths
27	Maharashtra	6363442	6159676	134201

c) Daily trend of new cases:

PROGRAM:

```
daily_cases = df.groupby('Date')['Confirmed'].sum().diff().fillna(0)
```

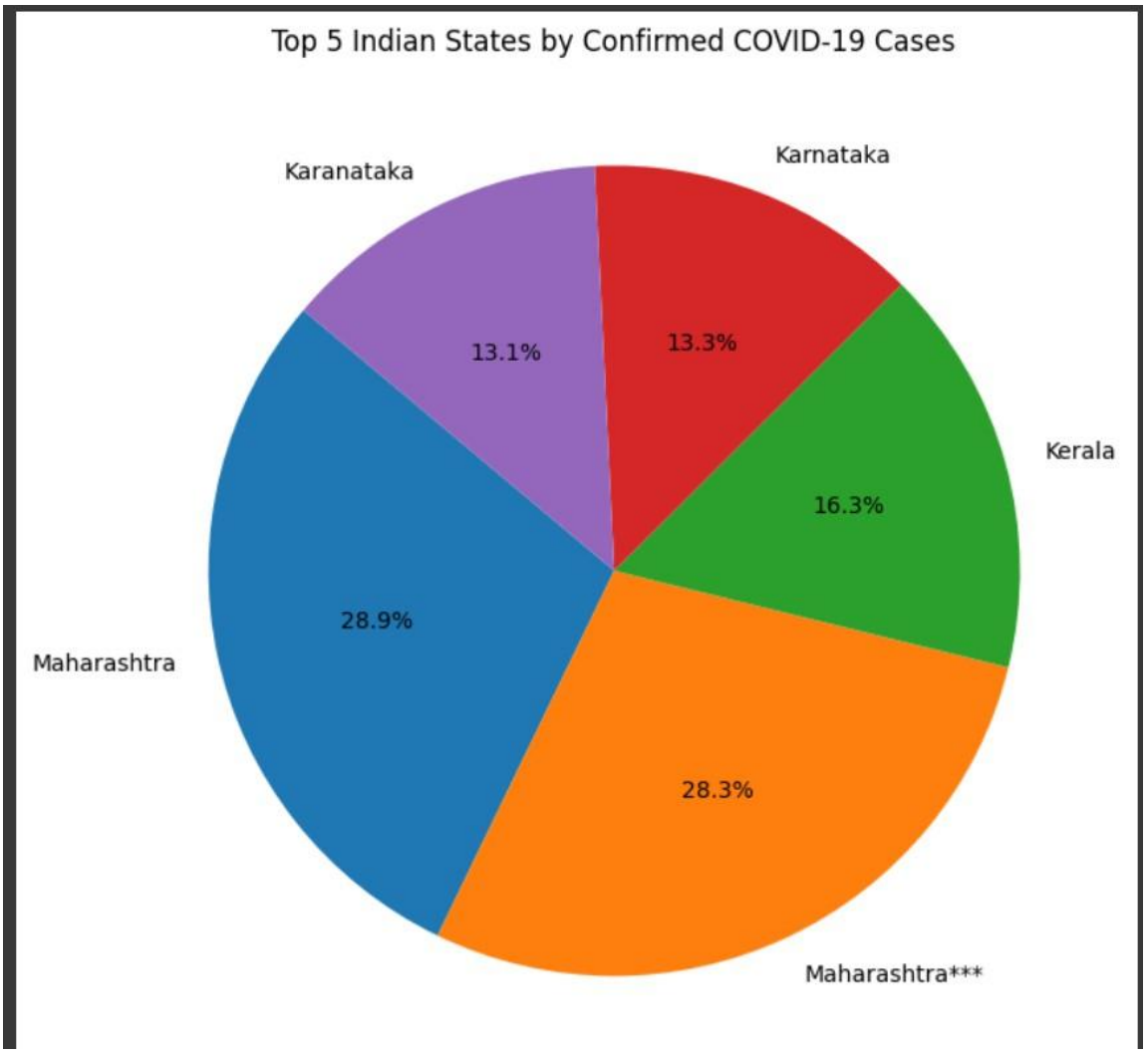
STEP 4: VISUALIZATIONS

a) Pie Chart: Top 5 States by Confirmed Cases

PROGRAM:

```
import matplotlib.pyplot as plt

top5_states = statewise_total.sort_values('Confirmed', ascending=False).head(5)

plt.figure(figsize=(8, 8))

plt.pie(top5_states['Confirmed'],labels=top5_states['State/UnionTerritory'],
autopct='%1.1f%%', startangle=140)

plt.title('Top 5 Indian States by Confirmed COVID-19 Cases')

plt.show()
```

OUTPUT:

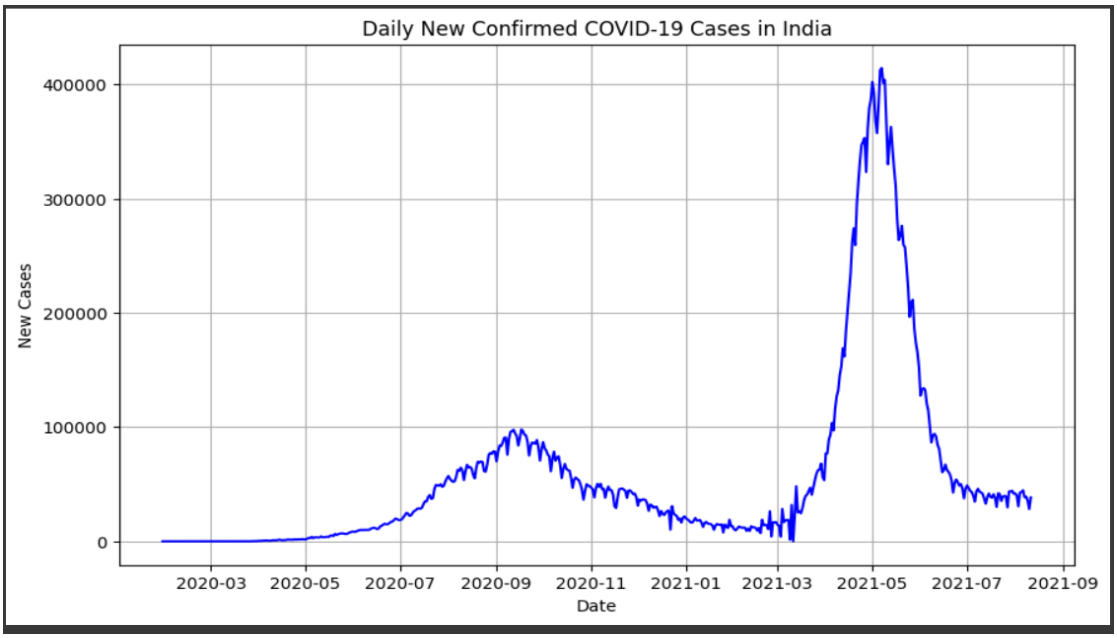


b) Line Graph: Daily Trend of Confirmed Cases

PROGRAM:

```
plt.figure(figsize=(10, 6))
plt.plot(daily_cases.index, daily_cases.values, color='blue')
plt.title('Daily New Confirmed COVID-19 Cases in India')
plt.xlabel('Date')
plt.ylabel('New Cases')
plt.grid(True)
plt.show()
```

OUTPUT:



STEP 5:OBSERVATION

- **Top affected states** (e.g., Maharashtra, Kerala, Karnataka) account for the majority of confirmed cases.
- **Trend graph** shows multiple waves—sharp increases followed by declines.
- **Lockdown periods** and **vaccination rollouts** align with noticeable trend changes.
- **Deaths and recovery rates** vary by region and wave, highlighting healthcare disparities.

Ex.No:3	EDA ON YOUTUBE TRENDING VIDEOS DATASET

INTRODUCTION:

YouTube has become a dominant platform for video sharing, content creation, and audience engagement worldwide. The YouTube Trending Videos **Dataset** provides a snapshot of videos that were trending in various regions over time, offering valuable insights into user preferences, content popularity, and engagement metrics.

This Exploratory Data Analysis (EDA) aims to uncover trends in video categories, the frequency of trending videos across different channels, and patterns in user interactions such as views, likes, and comments. By analyzing this data, we can better understand what makes a video trend, which content types perform best, and how users engage with trending content.

DATA SOURCE:

Dataset link: https://www.kaggle.com/datasets/anushabellam/Trending videos on Youtube

GOOGLE COLAB LINK:

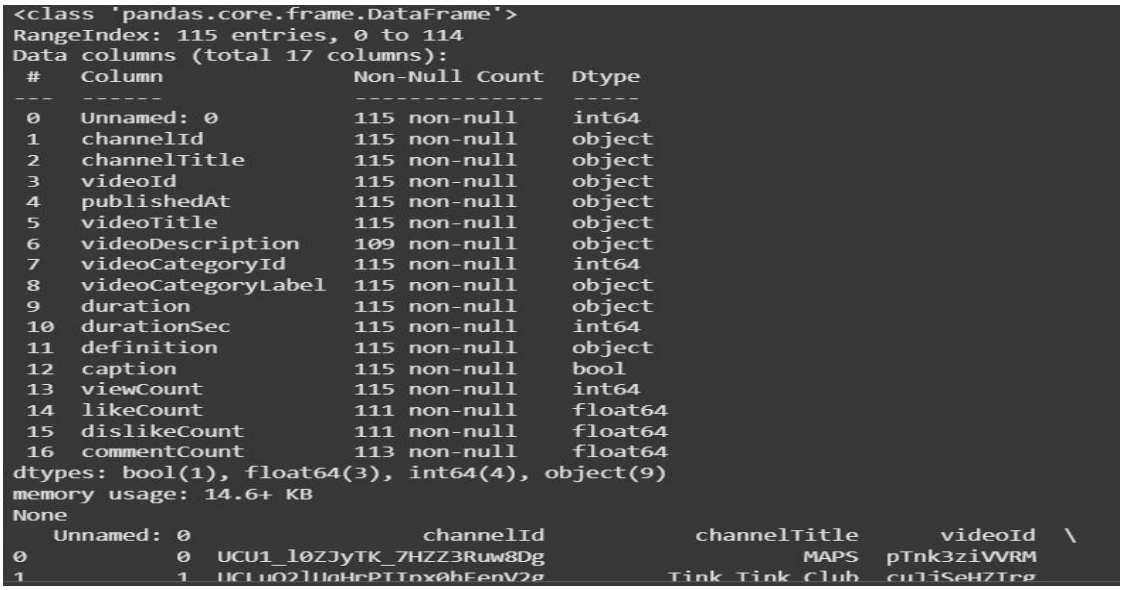
https://colab.research.google.com/drive/1xkMAoAhJsC8CxQH-ZaAoNUXe9g2HoFxs?usp=sharing

STEP 1: LOAD AND INSPECT THE DATASET

PROGRAM:

```
import pandas as pd

df = pd.read_csv('USvideos.csv')

print(df.info())

print(df.head())
```

OUTPUT:



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 115 entries, 0 to 114
Data columns (total 17 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Unnamed: 0           115 non-null    int64
 1   channelId            115 non-null    object
 2   channelTitle         115 non-null    object
 3   videoId              115 non-null    object
 4   publishedAt          115 non-null    object
 5   videoTitle           115 non-null    object
 6   videoDescription     109 non-null    object
 7   videoCategoryId      115 non-null    int64
 8   videoCategoryLabel   115 non-null    object
 9   duration             115 non-null    object
 10  durationSec          115 non-null    int64
 11  definition           115 non-null    object
 12  caption              115 non-null    bool
 13  viewCount            115 non-null    int64
 14  likeCount            111 non-null    float64
 15  dislikeCount         111 non-null    float64
 16  commentCount         113 non-null    float64
dtypes: bool(1), float64(3), int64(4), object(9)
memory usage: 14.6+ KB
None
   Unnamed: 0                channelId           channelTitle      videoId  \
0           0  UCU1_l0ZJyTK_7HZZ3Ruw8Dg                  MAPS  pTnk3ziVVRM
1           1  UCLuO2lUgHrPIInx0bEenV2g         Tink Tink Club  culiSeHZIrg
```

STEP 2: DATA CLEANING

PROGRAM**:**

```
df = df.drop_duplicates()

df = df.dropna()

df['publishedAt'] = pd.to_datetime(df['publishedAt'], errors='coerce')
```

STEP 3: KEY CALCULATIONS

PROGRAM:

```
most_common_categories = df['videoCategoryId'].value_counts().head(10)

top_channels = df['videoTitle'].value_counts().head(5)

avg_likes = df['likeCount'].mean()

avg_views = df['viewCount'].mean()

avg_comments = df['commentCount'].mean()

average_metrics = {

   'Average Likes': avg_likes,

   'Average Views': avg_views,

   'Average Comments': avg_comments

}

print(average_metrics)
```

OUTPUT:

{'Average Likes': np.float64(182.2095238095238), 'Average Views': np.float64(9999.657142857142), 'Average Comments': np.float64(82.97142857142858)}

STEP 4: VISUALIZATIONS

PROGRAM:

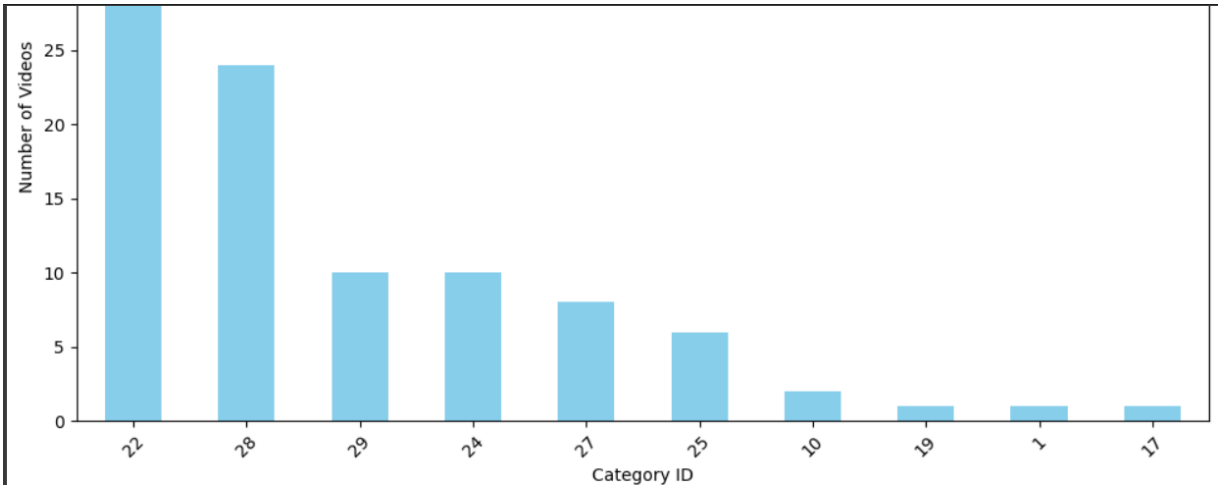
a) Bar chart: Video count by category

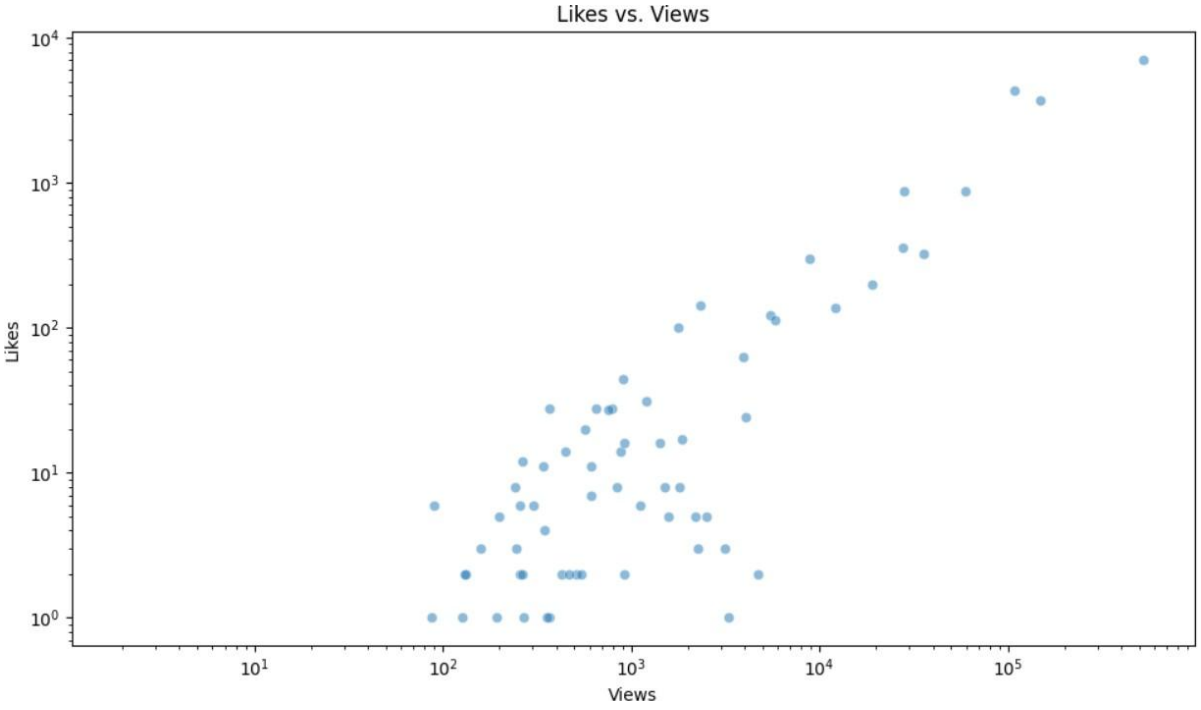
```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10, 6))
most_common_categories.plot(kind='bar', color='skyblue')
plt.title('Top 10 Video Categories by Count')
plt.xlabel('Category ID')
plt.ylabel('Number of Videos')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

b) Scatter plot: Likes vs Views

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='viewCount', y='likeCount', data=df, alpha=0.5)
plt.title('Likes vs. Views')
plt.xlabel('Views')
plt.ylabel('Likes')
plt.xscale('log')
plt.yscale('log')
plt.tight_layout()
plt.show()
```

OUTPUT:





STEP 5:OBSERVATION

- Top Categories: Certain categories like music, entertainment, and news dominate the trending list.
- Channel Popularity: A few channels consistently produce trending content.
- Engagement Patterns: There's a strong positive correlation between views and likes.
- Outliers: Some videos have extremely high views but relatively low likes/comments, suggesting passive viewing.