Full step-by-step guide to Data Analytics using Python, with code examples using CSV data. We'll go from data loading to insights and visualization. I'll make it practical, so you can follow along with your own CSV file.

Let's structure it in **phases**:

Step 1: Setup Your Environment

You need Python installed. Commonly, people use **VS Code**. Also, install these essential packages:

```
pip install pandas numpy matplotlib seaborn
```

- **pandas** → data manipulation
- **numpy** → numerical computations
- **matplotlib & seaborn** → data visualization

Step 2: Load CSV Data

Assume we have a CSV file `sales_data.csv` with columns like `Date, Product, Sales, Quantity`.

```
import pandas as pd

# Load CSV
data = pd.read_csv('sales_data.csv')

# View first 5 rows
print(data.head())
```

Output Example:

```
         Date  Product  Sales  Quantity
0  2025-01-01  Laptop   1000         2
1  2025-01-02  Phone     500         3
2  2025-01-03  Tablet    700         1
```

Step 3: Explore the Data (Exploratory Data Analysis - EDA)

3.1 Check Data Info

```
print(data.info())
print(data.describe())
```

- `info()` → column types, missing values
- `describe()` → summary stats for numeric columns

3.2 Check Missing Values

```
print(data.isnull().sum())
```

3.3 Basic Statistics

```
# Mean sales
print("Average Sales:", data['Sales'].mean())

# Total quantity sold
print("Total Quantity:", data['Quantity'].sum())

# Max sale
print("Max Sale:", data['Sales'].max())
```

Step 4: Data Cleaning

Typical steps:

- Handle missing values
- Convert data types
- Remove duplicates

```
# Drop missing values
data = data.dropna()

# Convert 'Date' to datetime
data['Date'] = pd.to_datetime(data['Date'])

# Remove duplicates
data = data.drop_duplicates()
```

Step 5: Data Analysis

5.1 Grouping Data

```
# Total sales per product
sales_per_product = data.groupby('Product')['Sales'].sum()
print(sales_per_product)

# Average quantity sold per product
avg_quantity = data.groupby('Product')['Quantity'].mean()
print(avg_quantity)
```

5.2 Trend Analysis

```
# Daily sales trend
daily_sales = data.groupby('Date')['Sales'].sum()
print(daily_sales)
```

Step 6: Data Visualization

```
import matplotlib.pyplot as plt
import seaborn as sns

# 1. Sales per Product (Bar chart)
plt.figure(figsize=(8,5))
sns.barplot(x=sales_per_product.index, y=sales_per_product.values)
plt.title('Total Sales per Product')
plt.ylabel('Sales')
plt.show()

# 2. Daily Sales Trend (Line chart)
plt.figure(figsize=(10,5))
plt.plot(daily_sales.index, daily_sales.values, marker='o')
plt.title('Daily Sales Trend')
plt.xlabel('Date')
plt.ylabel('Sales')
plt.xticks(rotation=45)
plt.show()

# 3. Sales Distribution (Histogram)
plt.figure(figsize=(6,4))
sns.histplot(data['Sales'], bins=10, kde=True)
plt.title('Sales Distribution')
plt.show()

# 4. Correlation Heatmap
plt.figure(figsize=(5,4))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

Step 7: Advanced Analysis

7.1 Top Selling Product

```
top_product = sales_per_product.idxmax()
print("Top Selling Product:", top_product)
```

7.2 Monthly Sales

```
data['Month'] = data['Date'].dt.to_period('M')
monthly_sales = data.groupby('Month')['Sales'].sum()
print(monthly_sales)
```

7.3 Pivot Tables

```
# Pivot: Product vs Month Sales
pivot = data.pivot_table(values='Sales', index='Month', columns='Product',
aggfunc='sum')
print(pivot)
```

□ Step 8: Save Cleaned/Processed Data

```
data.to_csv('cleaned_sales_data.csv', index=False)
```

This workflow covers **everything from raw CSV data to analysis, visualization, and saving results**.