#What is Data Analytics?

- **Definition**:
  Data Analytics is the process of collecting, processing, visualizing, and interpreting data to extract meaningful insights and support decision-making.

- **Core Steps in Data Analytics**:
  1. Data Collection → Loading data from files, APIs, databases.
  2. Data Cleaning → Handling missing values, duplicates, data types.
  3. Data Analysis → Aggregation, statistical summaries, grouping.
  4. Data Visualization → Charts, plots to reveal patterns.
  5. Interpretation → Derive insights, report findings.

#Important Libraries in Data Analytics:

Library	Purpose
pandas	Data manipulation, handling structured data (DataFrames).
numpy	Numerical computations (arrays, matrices, basic stats).
matplotlib	Basic plotting (line plots, bar charts, histograms).
seaborn	Statistical visualization built on top of matplotlib (correlation heatmaps, boxplots).
plotly	Interactive, web-based visualization.
openpyxl / xlrd	Excel file reading/writing.
SQLAlchemy / sqlite3	Database querying and connection.

#Sample Workflow + Code Snippets:

Step 1 – Load Data (CSV Example)

```
import pandas as pd

data = pd.read_csv('data.csv')
print(data.head())  # Show first 5 rows
```

Step 2 – Data Cleaning

```
# Check for missing values
print(data.isnull().sum())

# Fill missing numerical data with median
data['column_name'] = data['column_name'].fillna(data['column_name'].median())

# Remove duplicate rows
data = data.drop_duplicates()

# Convert column to datetime
data['date_column'] = pd.to_datetime(data['date_column'])
```

Step 3 – Data Analysis (Aggregation + Grouping)

```
# Statistical summary
print(data.describe())

# Group by example
avg_sales_per_region = data.groupby('region')['sales'].mean()
print(avg_sales_per_region)
```

Step 4 – Visualization with matplotlib

```
import matplotlib.pyplot as plt

# Line plot
plt.plot(data['date_column'], data['sales'])
plt.xlabel('Date')
plt.ylabel('Sales')
plt.title('Sales over Time')
plt.show()
```

Step 5 – Visualization with seaborn

```
import seaborn as sns

# Correlation heatmap
corr = data.corr()
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Feature Correlations')
plt.show()

# Boxplot
sns.boxplot(x='region', y='sales', data=data)
plt.title('Sales Distribution by Region')
plt.show()
```

Step 6 – Interactive Visualization (Optional)

```
import plotly.express as px
fig = px.scatter(data, x='sales', y='profit', color='region', size='transactions')
fig.show()
```

#Why These Libraries Matter:

Library	Why Important
pandas	Backbone of data manipulation → Handles large datasets, filtering, merging, grouping.
numpy	Fast numerical computations → Operates on large arrays efficiently.
matplotlib	Simple, foundational visualization → Good for static plots.
seaborn	High-level statistical plots → Great for correlation analysis, distributions.
plotly	Interactive dashboards → Useful for reporting insights in business.

#Critical Learning Outcome:

Goal	Outcome
Data Cleaning	Learn how to handle real-world messy data.
Aggregation + Grouping	Understand how to summarize data for insights.
Visualizations	Practice how to reveal patterns and relationships visually.
Reporting	Learn to communicate insights effectively.

Here are additional important libraries with simple code snippets, purely focused on code without detailed explanations:

#NumPy (Numerical Computation)

```
import numpy as np

# Create array
arr = np.array([1, 2, 3, 4, 5])

# Basic stats
print(np.mean(arr))
print(np.median(arr))
print(np.std(arr))
```

#SciPy (Advanced Statistical Functions)

```
from scipy import stats

data = [1, 2, 3, 4, 5, 5, 5, 6, 7]

# Z-score normalization
z_scores = stats.zscore(data)
print(z_scores)
```

#OpenPyXL (Excel File Handling)

```
from openpyxl import load_workbook

wb = load_workbook('data.xlsx')
sheet = wb.active

# Read cell value
print(sheet['A1'].value)
```

#SQLAlchemy (Database Connection)

```
from sqlalchemy import create_engine
import pandas as pd

engine = create_engine('sqlite:///data.db')
df = pd.read_sql('SELECT * FROM table_name', engine)
```

```
print(df.head())


#Pandas Profiling (Automatic EDA Report)
from pandas_profiling import ProfileReport

profile = ProfileReport(data, title="Data Report", explorative=True)
profile.to_file("report.html")


#Missingno (Missing Data Visualization)
import missingno as msno

msno.matrix(data)


#Plotly Express (Interactive Plot)
import plotly.express as px

fig = px.bar(data, x='category', y='sales')
fig.show()


#Feature-engine (Feature Engineering)
from feature_engine.imputation import MeanMedianImputer

imputer = MeanMedianImputer(imputation_method='median', variables=['age', 'salary'])
data = imputer.fit_transform(data)
```