🔥 ADVANCED PANDAS

9️⃣ Date & Time Handling in Pandas

📒

Real-world data mein dates, time, timestamps hote hain (sales date, login time, attendance).
Pandas mein `datetime` data ko handle karne ke liye `pd.to_datetime()` use hota hai.

💻 CODE + EXAMPLE

```
import pandas as pd


df = pd.DataFrame({
    "Name": ["Aman", "Riya", "Neha"],
    "Date": ["2024-01-10", "2024-02-15", "2024-03-20"]
})


df["Date"] = pd.to_datetime(df["Date"])


df["Year"] = df["Date"].dt.year
df["Month"] = df["Date"].dt.month
df["Day"] = df["Date"].dt.day


print(df)
```

🔟 Pivot Table

📒 THEORY

Pivot Table ka use summary report **banane ke liye hota hai (jese: age ke hisab se average marks).**

💻 CODE + EXAMPLE

```
df = pd.DataFrame({
    "Name": ["Aman", "Riya", "Neha", "Aman"],
    "Age": [20, 21, 20, 21],
    "Marks": [85, 90, 78, 88]
})


pivot = pd.pivot_table(df, values="Marks", index="Age", aggfunc="mean")
print(pivot)
```

11. value_counts(), unique(), nunique()

Data analysis mein frequency & uniqueness **check karna bahut common hai.**

```
df = pd.DataFrame({
    "Result": ["Pass", "Fail", "Pass", "Pass"],
    "Age": [20, 21, 20, 22]
})
```

```
print(df["Result"].value_counts())
print(df["Age"].unique())
print(df["Age"].nunique())
```

12. Handling Large Files (Performance)

📘 **THEORY**

Badi CSV files ko ek baar mein load karna memory issue deta hai.
Isliye hum chunks mein file read karte hain.

💻 **CODE**

```
chunks = pd.read_csv("data.csv", chunksize=1000)
for chunk in chunks:
    print(chunk.head())
```

NOTE☞ **Har loop mein 1000 rows process hoti hain.**

13. String Operations (Text Data)

Names, emails, city, product sab string data **hota hai.**
Pandas .str methods deta hai.

💻 **CODE + EXAMPLE**

```
df = pd.DataFrame({
    "Name": ["aman", "riya", "neha"]
})

print(df["Name"].str.upper())
print(df["Name"].str.contains("a"))
print(df["Name"].str.replace("a", "@"))
```

14. Rename Columns & Reset Index

Clean & readable dataframe ke liye rename important hota hai.

💻 CODE

```
df = pd.DataFrame({
    "Marks": [85, 90, 78]
})

df.rename(columns={"Marks": "Score"}, inplace=True)
df.reset_index(drop=True, inplace=True)

print(df)
```

15 Export Data (CSV / Excel)

Project ke end mein data ko export karna padta hai.

💻 CODE

```
df.to_csv("output.csv", index=False)
df.to_excel("output.xlsx", index=False)
```

☞ index=False → unwanted index remove karta hai.

□ ADVANCED PANDAS – KEYWORDS EXPLAINED

9□Date & Time Handling – Keywords

🔑 Keywords & Meaning

Keyword	Kaam
pd.to_datetime()	Column ko date-time format mein convert karta hai
.dt	Date-time properties access karta hai
.dt.year	Sirf year nikalta hai
.dt.month	Month number deta hai
.dt.day	Day nikalta hai

💻 Code

```
df["Date"] = pd.to_datetime(df["Date"])

df["Year"] = df["Date"].dt.year
```

🔟 Pivot Table – Keywords

Keyword	Kaam
pd.pivot_table()	Summary table banata hai
values	Kis column ka calculation karna hai
index	Row-wise grouping
aggfunc	Aggregation function (mean, sum, count)

```
pd.pivot_table(df, values="Marks", index="Age",
aggfunc="mean")
```

1️⃣1️⃣Value Counts / Unique – Keywords

Keyword	Kaam
value_counts()	Frequency count batata hai
unique()	Unique values ki list deta hai
nunique()	Unique values ki count deta hai

```
df["Result"].value_counts()
```

1️⃣2️⃣ Handling Large Files – Keywords

Keyword	Kaam
pd.read_csv()	CSV file read karta hai
chunksize	File ko parts mein tod deta hai
for chunk in chunks	Ek-ek chunk process karta hai

```
pd.read_csv("data.csv", chunksize=1000)
```

1️⃣3️⃣ String Operations – Keywords

Keyword	Kaam
.str	String methods use karne deta hai
.upper()	Capital letters
.contains()	Check karta hai text present hai ya nahi
.replace()	Text replace karta hai

```
df["Name"].str.upper()
```

1️⃣4️⃣ Rename & Index – Keywords

Keyword	Kaam
rename()	Column ka naam change karta hai
columns	Column mapping deta hai
inplace=True	Original data change karta hai

Keyword	Kaam
reset_index()	Index ko normal number banata hai
drop=True	Old index delete karta hai

```
df.rename(columns={"Marks":"Score"}, inplace=True)
```

15□Export Data – Keywords

Keyword	Kaam
to_csv()	CSV file banata hai
to_excel()	Excel file banata hai
index=False	Index ko file mein save nahi karta

```
df.to_csv("output.csv", index=False)
```

□ BONUS – VERY IMPORTANT COMMON KEYWORDS

Keyword	Kaam
inplace=True	Original DataFrame modify
axis=0	Rows par operation
axis=1	Columns par operation
head()	Top 5 rows
tail()	Last 5 rows
info()	Data types & memory
describe()	Statistics summary