MOP for Exercise 2 – Python

1) Environment setup (once)

1.1 Install tools

- Windows/Mac: Install Anaconda (includes Python + Jupyter): google “Anaconda Individual Edition” → Download → Install (default options).

1.2 Create a working folder

- Make a folder on Desktop called training_p.
- You’ll save all files (CSV, DB, notebooks) here.

1.3 Launch Jupyter

- Open Anaconda Navigator → Jupyter
- In the browser tab, navigate into training_p/.
- Click New → Python 3 (ipykernel) to create a notebook. Name it training.ipynb.

2) Prepare the sample dataset
2.1 Create telecom_usage.csv
In a new notebook cell, paste and run:

```
import pandas as pd

raw = """customer_id,data_used_gb,calls_made,revenue_inr,region,date
1001,5.2,25,180,Delhi,2025/09/25
1002,,40,280,Mumbai,2025-09-25
1003,7.8,32,210,chennai,25-09-2025
1004,15.6,55,,DELHI,2025-09-25
1005,3.4,18,120,Kolkata,2025-09-25
1005,3.4,18,120,Kolkata,2025-09-25
1006,8.7,52,150,Jabalpur,2025-09-25
1007,5.8,22,135,Bhopal,2025-09-25
1008,6.5,40,180,Gwailor,2025-09-25
1009,11.7,100,350,Indore,2025-09-25
1010,2.5,23,105,Chandigarh,2025-09-25
1011,9.5,50,190,Ambala,2025-09-25
1012,12.1,80,270,Bangalore,2025-09-25
1013,,37,205,Gurgaon,2025-09-25
1014,4.8,43,132,Hyderbad,2025-09-25
"""

open("telecom_usage.csv","w").write(raw)
pd.read_csv("telecom_usage.csv").head()
```

3) Exercise 2 — Python cleaning

3.1 Load the CSV

```
import pandas as pd
df = pd.read_csv("telecom_usage.csv")
print("Original shape:", df.shape)   # (rows, columns)
df.head()
```

Why: Confirms the file loaded and shows row/column count.

3.2 Handle missing values
Tip: If you only want to require data_used_in_GB and date to be present:

```
before = len(df)
df = df.dropna()                    # removes rows with any blank
after = len(df)
print(f"Removed {before - after} rows with missing values.")
# df = df.dropna(subset=["data_used_in_GB","date"])

df.to_csv("missing_telecom_usage.csv", index=False)
print("Saved missing_telecom_usage.csv with", len(df), "rows.")
```

3.3 Standardise the date format

```
df["date"] = pd.to_datetime(df["date"], errors="coerce")  # coerce turns bad dates into NaT
bad_dates = df["date"].isna().sum()
print("Unparseable date rows:", bad_dates)
df = df.dropna(subset=["date"])  # simplest for class
```

If you see unparseable dates, decide: drop them or fix manually.

3.4 Remove duplicates

Why: Prevents double counting in metrics.

```
before = len(df)

# Keep the first occurrence of each duplicate and drop the rest
df = df.drop_duplicates(keep='first')

after = len(df)
print(f"Removed {before - after} duplicate rows (kept the first occurrence of each).")
```

3.5 Save the cleaned dataset

```
df.to_csv("cleaned_telecom_usage.csv", index=False)
print("Saved cleaned_telecom_usage.csv with", len(df), "rows.")
```

3.6 Verify the save

```
import pandas as pd
pd.set_option('display.max_rows', None)  # show all rows in output
cleaned_df = pd.read_csv("cleaned_telecom_usage.csv")
display(cleaned_df)
```

Checkpoint (students record in notes):

- Original rows: ___
- After missing-value removal: ___
- After duplicate removal: ___
- After date parsing/removal: ___
- Final rows saved: ___

****