A Simple Analysis Project

Put the whole pipeline together on one small project. A student-marks dataset, cleaned, used to answer three questions with code, and written up as five lines of findings.

Introduction

You have learned the pieces: types and cleaning, tables, arrays, charts, and summary numbers. Now it is time to use them together. A real analysis project is not one big trick — it is the same small steps, done in order, on one dataset, answering questions someone actually asked.

This lesson gives you a complete mini-project: a small table of student marks with real problems baked in — a missing mark, a duplicate row, a subject written two ways. You will clean it, answer three questions with code, and write the findings in five plain lines. That short write-up is the most important part. Code that nobody can explain is not analysis; a five-line story anyone can understand is.

One working habit separates tidy projects from messy ones: **never edit your original data.** Keep the raw table exactly as you received it, and do all cleaning on a copy. That way, if a cleaning step goes wrong — you drop the wrong row, or fix a spelling the wrong way — you can start again from the original instead of guessing what it used to say. In code this is simple: load the raw table into one variable, and build the cleaned table as a new one, step by step, printing the row count after each step so a surprise drop is caught at once.

(Roman Urdu: Ab tamam tukray jor kar ek mukammal project karte hain — ek chhoti table, us ki safai, teen sawalon ke jawab, aur aakhir mein paanch saaf lines mein natija. Code se zyada zaroori woh paanch lines hain jo har koi samajh sake.)

Key points

- **Every project starts with questions, not code.** Write the questions first, then let them choose what you compute.
- **Clean before you analyse.** One duplicate row or one missing mark quietly changes every answer after it.
- **Answer each question with one clear piece of code,** and print the result where you can see it before moving on.
- **A chart is evidence, not decoration.** In this project the bar chart proves which subject is weakest — it is part of the answer.
- **Findings are written for a non-technical reader.** No jargon, no code words, just what was found and what it suggests.
- **Say what you dropped and why.** "One row dropped for a missing mark" keeps the findings honest.
- **Keep the project small on purpose.** Ten clean rows analysed well teach more than a thousand messy rows analysed badly.

- **Keep the raw data untouched.** Clean a copy, never the original, so any mistake can be undone by starting again.
- **Count rows after every step.** "8 rows in, 7 rows out" after dropping one missing mark is expected; "8 rows in, 4 rows out" means a step went wrong and you must stop and check.

Examples

Example 1 — The dataset and its cleaning

This example needs `pip install pandas` first:

```
import pandas as pd

df = pd.DataFrame([
    {"name": "Ali", "subject": "Maths", "marks": 85},
    {"name": "Sara", "subject": "Maths", "marks": 92},
    {"name": "Bilal", "subject": "English", "marks": 74},
    {"name": "Ayesha", "subject": "Maths", "marks": 88},
    {"name": "Usman", "subject": "English", "marks": 61},
    {"name": "Hina", "subject": "maths", "marks": 79},
    {"name": "Ali", "subject": "Maths", "marks": 85},
    {"name": "Faraz", "subject": "English", "marks": None},
])

df = df.drop_duplicates()
df["subject"] = df["subject"].str.capitalize()
print("Missing marks:", df["marks"].isna().sum())
df = df.dropna()
print(df)
print("Rows left:", len(df))
```

What this means: the duplicate Ali row is gone, "maths" is fixed to "Maths" so the subject counts are correct, and Faraz's row is dropped because his mark is missing — 7 rows are left. Every later answer now rests on clean data, and the cleaning steps are written down so anyone can repeat them.

Example 2 — Answering the three questions with code

This example needs `pip install pandas` first, and uses the cleaned table from Example 1:

```
# Question 1: What is the average mark overall?
print("Overall average:", df["marks"].mean())

# Question 2: Which subject has the higher average?
print(df.groupby("subject")["marks"].mean())

# Question 3: Which students scored below 70?
print(df[df["marks"] < 70][["name", "subject", "marks"]])
```

What this means: the overall average is 79.9. Maths averages 86.0 and English averages 67.5, so Maths is clearly stronger. Only Usman, at 61 in English, falls below 70. Three questions, three short pieces of code, three answers you can check by hand from the table — that checkability is what makes an analysis trustworthy.

Example 3 — The findings in five lines

No code here — this is the part a teacher, a parent or a shop owner would actually read:

```
Findings — Class marks analysis (7 students, after cleaning):
1. The class average is 79.9 marks.
2. Maths (average 86.0) is much stronger than English (average 67.5).
3. Only one student, Usman, scored below 70 — 61 in English.
4. The gap suggests English needs extra practice time, not Maths.
5. Limit: this is only 7 students, so one new result could change the averages.
```

What this means: five lines, no jargon, each line a fact or a careful suggestion. Line 5 is the honesty line — small data means shaky conclusions, and a good analyst says so instead of hiding it. Lesson DS.8 is all about writing findings like these.

Example 4 — A fourth question, and a chart as evidence

This example needs `pip install pandas` and `pip install matplotlib` first, and uses the cleaned table from Example 1:

```
import matplotlib.pyplot as plt

# Question 4: in which subject do more students score below 80?
below_80 = df[df["marks"] < 80]
print(below_80[["name", "subject", "marks"]])
print(below_80.groupby("subject").size())

averages = df.groupby("subject")["marks"].mean()
averages.plot(kind="bar")
plt.ylabel("Average marks")
plt.title("Maths averages 86, English averages 67.5")
plt.show()
```

What this means: below 80 sit Hina (79, Maths) and both English students (74 and 61) — so English has 2 students below 80 against Maths' 1, which matches the averages story from a different angle. The bar chart then proves the main finding visually: the Maths bar towers over the English bar. Two angles on the same finding — counts and averages — agreeing with each other is what makes an analysis convincing.

Example 5 — Checking your answers by changing one mark

Still using the cleaned table from Example 1, change one value and watch what moves:

```
df_check = df.copy()
df_check.loc[df_check["name"] == "Usman", "marks"] = 75

print("New overall average:", df_check["marks"].mean())  # 82.2
print(df_check.groupby("subject")["marks"].mean())
print("Below 70 now:", df_check[df_check["marks"] < 70].shape[0])  # 0
```

What this means: raising Usman's 61 to 75 lifts the overall average from 79.9 to 82.2, lifts English from 67.5 to 74.5, and leaves nobody below 70. This sensitivity check teaches two things: small data moves fast when one value changes (which is why lesson DS.7's findings carry a limit line), and a good analyst re-runs the answers after any change instead of trusting old numbers.

Example 6 — Real-world case: a shop's month-end analysis

A small shop owner ends the month with a notebook of daily sales: date, item, amount. Her questions are the same shape as this lesson's: which item sold most, which week was weakest, and which days fell below a target. She cleans first (two days entered twice, one amount written as text), answers each question with grouped averages and filters, and writes five lines: total sales, best item, weakest week, days below target, and one limit ("one month only, festival week included"). Then she acts: she orders more of the best item and less of the weakest week's stock. The dataset is a notebook, not a database — but the method is the full pipeline, and the five lines are what she actually uses.

(Roman Urdu: Dukaan ki month-end analysis bhi wohi pipeline hai — safai, teen sawal, paanch lines, phir seedha faisla.)

Quick recap

- Start with questions, then clean, then analyse, then write findings.
- Clean first: duplicates out, formats fixed, missing values handled and stated.
- One question, one clear piece of code, one visible answer.
- Findings are five plain lines for a non-technical reader.
- Always include the limit: what the small data cannot prove.

Common mistakes

1. **Analysing before cleaning** — the duplicate row and the "maths" spelling would have skewed both subject averages. Fix: clean, print the table, and only then compute answers.
2. **Writing findings full of code words** — "the groupby mean of df" means nothing to a teacher. Fix: translate every result into a plain sentence before you call it a finding.
3. **Hiding the small-data problem** — 7 students cannot prove a whole school's weakness. Fix: end with a limit line, like line 5 above, every single time.
4. **Editing the raw data by hand and leaving no record** — fixing a value in the original file means nobody, including future you, can tell what changed or why. Fix: keep the raw file untouched and put every fix in code, where it is visible and repeatable.
5. **Answering a question nobody asked** — a beautiful chart about ages does not help a teacher who asked about subjects. Fix: before computing anything, write the questions down and check every result against them at the end.

Practice

1. Copy Examples 1 and 2 and run them. Change one mark and watch which answers change.
2. Add two students of your own to the table, re-run the cleaning and the three questions, and write your own five-line findings.
3. Rewrite your five findings for a different reader — a student instead of a teacher. What changes in the wording?

1. Run Example 4's fourth question on the cleaned table. In your own words, write one sentence that the below-80 counts add to the story from Examples 2 and 3.
2. Do the sensitivity check from Example 5 with a different change — lower Sara's 92 to 72 — and write down which answers move and which stay the same. Why does the English average not move at all?
3. Take a small dataset of your own (your weekly expenses, or a week's screen time) and run the full project on it: questions first, cleaning on a copy, answers, and a five-line findings note with a limit line.

Practice quiz

1. What is the first step of a small analysis project?

- Draw charts
- Clean the data
- Write findings
- Share the result

Answer: Clean the data

2. How do you answer questions from a cleaned table?

- Guess
- Run code on the table — filters, counts and averages — and read the results
- Ask the table
- Copy answers from the internet

Answer: Run code on the table — filters, counts and averages — and read the results

3. What should the findings of a small project look like?

- A 20-page report
- A few short, plain lines that answer the questions asked
- Only charts, no words
- The raw data again

Answer: A few short, plain lines that answer the questions asked

---

Watch the video lesson: https://youtu.be/G78YFKUmIJM

Read online: https://scopetreasure.com/notes/ds-analysis-project · More free tools and notes at https://scopetreasure.com