Data Validation Challenge

Note : The code will later be tested with a different, more challenging file with the same data schema but with much less data quality on the same or more fields, so design your models and validators to be reasonably robust.

You're given a newline-delimited JSON file (orders_sample.ndjson) where each line is one order. Imagine it's coming from a streaming API.

Your task is to:

1. Model the data with Pydantic

- Create Pydantic models that describe an "order" and its nested structures (e.g. shipping info).
- Use appropriate data types
- Use custom validators where needed.

2. Infer and enforce business rules from the data

- Inspect the sample file and infer reasonable rules.
- Define additional rules you consider appropriate based on the data patterns.

3. Compute and validate the file line by line

- Separate each data row into lists that contain only valid data or invalid data.

4. Normalization vs rejection

As part of your validation logic, determine which types of "dirty" input values should be automatically normalized and which should be rejected outright.

5. Data Quality Analysis

After you've implemented your ingestion and validation logic, perform a small data-quality analysis over the dataset and your results. Use any approach you like (pure Python, Pandas, etc.), but the analysis should involve aggregations, filters, minimum/maximum values, and simple groupings.

Answer the following questions:

1. Overall Acceptance Rate

What fraction of records in the file are accepted by your validation logic versus rejected?

2. Per-Field Basic Profiles

For a few key fields of your choice (e.g. numeric, boolean, categorical, or text-like fields), compute simple summaries such as:

- Counts
- Distinct values
- Minimum / maximum (where applicable)

3. Missing / Unusable Values

For several columns of your choice, estimate how many records have values

that you consider “missing” or otherwise unusable, report these counts or

percentages.

4. Outliers and Extreme Values

For at least one numeric-like field, identify:

- The minimum and maximum values observed
- How many records fall into what you consider “extreme” ranges, based on simple thresholds you define.

5. Quality by Grouping

Pick a categorical-like field and compute, per category:

- How many records fall into each group
- For each group, what proportion of records are accepted vs rejected

by your validation

6. Output

Provide:

- Your exported Pydantic models.
- The actual script.
- A short summary (in comments or README) of:
- The rules you inferred.
- Which issues you correct automatically.
- Which issues cause hard failures.
- The answer to the analysis part