[CSED261] Introduction to Data Analysis

Practice #1. Wide → Tidy, Types, Timezones, and CSV vs Parquet (Python-only)

Abstract

Practice core data-engineering habits: schema thinking, wide → tidy reshape, type enforcement, timezone normalization, and storage comparison (CSV vs Parquet).

Deadlines & Deliverables

Code submission	9/10 23:59	{student_id}.py
Report submission	9/14 23:59	{student_id}.pdf

Important Notes

- Any submission that does not run will receive a score of 0.
- No late submissions will be accepted.

Problem Statements

Requirements

- Environment
    - python ≥ 3.10
    - pyarrow
- Rules
    - **Do not use pandas, polars and numpy library!**

Data

- A tiny, realistic sample derived from a London bike-share scenario.
- File: `data/sample_bike_wide.csv` has 3 stations × 2 dates, hourly counts from 08:00–18:00 in **Europe/London** local time.
- Some cells are missing (`'', 'NA'`, `'N/A'`).
- The resulting tidy data will have a unique record for each station at each timestamp.
- Schema
    - Wide schema in .csv file

```
- `station_id`: str (keep leading zeros)
- `station_name`: str
- `date`: ISO date string (`YYYY-MM-DD`)
- `tz`: str (`Europe/London`)
- hourly columns: integers or `None` (`''`, `'NA'`, `'N/A'` → `None`)
```

Process

1. Load wide-format CSV from data/sample_bike_wide.csv
2. Transform to tidy format with timezone handling
    - Reshape into the following tidy schema
        - station_id, station_name: Same as given input data
        - timestamp_local
            - The values from the hourly columns (h08 through h18) represent the count for that hour.
            - Combine with `date` to form a **timezone-aware** local datetime using `zoneinfo.ZoneInfo("Europe/London")`.
            - e.g., "2025-03-29T08:00:00+00:00"
        - timestamp_utc
            - Convert timestamp_local to UTC with `.astimezone(ZoneInfo("UTC"))` and output ISO8601 (use `Z` for UTC).
            - e.g., "2025-03-29T08:00:00Z"
        - count: Number of rentals for the given station and hour
    - Schema
        - Tidy schema for output file

```
- `station_id`: str,
- `station_name`: str,
- `timestamp_local`: datetime,
- `timestamp_utc`: datetime,
- `count`: int
```

3. Export data to both CSV and Parquet formats

Output

- Generated files in out/: bike_tidy.csv, bike_tidy.parquet
    - Example for .csv

```
station_id,station_name,timestamp_local,timestamp_utc,count
001,Waterloo,2025-03-29T08:00:00+00:00,2025-03-29T08:00:00Z,12
001,Waterloo,2025-03-29T09:00:00+00:00,2025-03-29T09:00:00Z,15
001,Waterloo,2025-03-29T10:00:00+00:00,2025-03-29T10:00:00Z,18
```

- Note. You do not need to submit the generated output files.

Grade

- Code [50 pts]
    - TA will compare generated .csv and .parquet files with the reference solution (solution.py).
    - Submit as {student_id}.py
    - Check if the submitted {student_id}.py runs correctly and meets requirements. Code that does not run or produces largely incorrect output receives 0 points.
- Report [50 pts]
    - After the code submission deadline, compare and analyze your code with the released reference solution
    - Include differences in implementation, data handling, and lessons learned
    - Submit as {student_id}.pdf