Assessment test: Data Analytics Engineering

This test is intended to verify basic skills in **Data Analytics Engineering** (data extraction, modeling, critical reasoning) for candidates to the role of Data Analytics Engineer in Shopfully.

The **deadline** for returning the output is specified to the candidate when sharing the test.

The results will then be used as a topic for discussion in the subsequent interview. The candidate is expected to complete the following using Python and SQL.

Project description

For the Shopfully team it is very important to understand the weather conditions for certain locations. Use the OpenWeatherMap API (use a free tier subscription) to get the current weather conditions for 3 cities where Shopfully has offices: Milano, Bologna, Cagliari.

The task has two main parts, one focuses on the modeling while the other one is centered on script writing.

Part 1 - Data Modeling

Look at the data structure provided by the API documentation:

● Decide which data could be considered important and bring value and discard the data which looks less relevant.

● The data granularity should be 1-hour (we want to have hourly temperature to be able to analyze historical data in the future).

● Create a logical and physical model for this data having the following questions in mind:

○ How many distinct weather conditions were observed (rain/snow/clear/...) in a certain period?

○ Can you rank the most common weather conditions in a certain period of time per city?

○ What are the temperature averages observed in a certain period per city? ○ What city had the highest absolute temperature in a certain period of time?

○ Which city had the highest daily temperature variation in a certain period of time? ○ What city had the strongest wind in a certain period of time?

Deliverable:

- Visualized logical schema;
- Complete DDL for physical database implementation;
- SQL queries.

Bonus Part

How would you handle incremental loading of hourly data?

Develop these two examples:

- **Delays in data availability**
Some weather stations may provide data with delays (e.g., the 10:00 data arrives at 12:00). What strategies would you implement to ensure reliable incremental loading despite these delays?

- **Corrections to already processed data**
Stations recalibrate their instruments every 3 days, so they may send corrected values that replace those already loaded (within a maximum time window of 3 days).

How would you handle corrections to already ingested data?
Which techniques would you use to keep the data consistent and up to date?

NB: Assume that both phenomena are predictable and part of the standard flow: no manual recalculation requests are required, but rather systematic system behaviors.

Design a robust and automatable strategy.

Part 2 - Script writing and KPI definition

- Automate the data download process.
- Store the raw (response) data in the format you find the most suitable.
- Identify the information you find useful and create a dataframe with it.
- Write the data into the table(s) you identified in the modeling process.
- Write the data to a relational database. You have the freedom to decide how to organize your data/relationships, data types, primary, foreign keys, indices, etc.

● Could you answer the questions from the previous section using aggregations in Python applied on the denormalized dataframe?

Notes

If you are not able to provide a script, you can concentrate on the modeling part of the data by looking at the expected API output/response in the documentation, but the bonus part is mandatory.

Organize the project taking into consideration that data needs could grow and involve:

● All Italian municipalities and beyond;
● Different ways to get the information and consequently different API calls (always using the same endpoint).