Data Engineering Challenge

Werkspot is a marketplace that connects homeowners with professionals to arrange home services.

The typical journey of a homeowner and a professional is as follows:

- A homeowner posts a job
- Interested professionals get in touch with the homeowner by writing a “Proposal”
- Contact details are shared between homeowner and professional if both parties are interested.

This assignment will focus on the activity of the **professionals**.
For the purpose of this assignment, let’s assume that in order to become an active member on the platform, the professional needs to create an account, choose the services they specialise in (e.g. gardening, painting, etc), and agree with the terms and conditions.
Once these steps are met, the professional can start getting in touch (=“Proposing”) with homeowners. best

<u>You are provided with the following dataset:</u>

Events dataset 'event_log.csv'

The dataset contains selected events from the professional’s journey on Werkspot.

Column	**Description**
event_id	A unique event identifier.
event_type	Describes the professional’s activity on the platform. This column stores one of the following types of activity: [created_account, became_able_to_propose, became_unable_to_propose, proposed] ● **created_account** event is triggered once, when the professional registers on the platform. ● **became_able_to_propose** event is triggered for the first time when the professional completes all the registration steps (selects services, agrees with terms and conditions). After this moment, the professional can contact homeowners and propose. Please note that this event can be triggered more than once (more on this in the next section).

	• **became_unable_to_propose** event is triggered if a professional didn't pay their invoice. After this moment, the professional can no longer contact homeowners. To become active again, the professional has to pay their invoice. Once paid, a **became_able_to_propose** event is triggered**,** indicating that the professional is back to being active on the platform. • **proposed** event is triggered every time a professional contacts a homeowner with an offer.
professional_id	A unique professional identifier.
created_at	Timestamp recorded for the event.
meta_data	Additional information associated with the **proposed** event type, with the following pattern: {service_id}_{service_name_nl]_{service_name_en}_{lead_fee} For example: **127_binnenhuis-ontwerp_interior-design_3.4** is for service_id=127, Interior Design (Binnenhuis Ontwerp in Dutch), with a (fictional) lead fee of €3.4 that will be paid by the professional to Werkspot for getting the homeowner's contact details.

The Challenge

The preferred languages to complete this challenge are **Python** and **SQL**, but you can choose any other language.
Please submit your code in a Github/Gitlab repository and provide all the files that are needed to execute the script in a Docker container.
The challenge should take around 5 hours. We encourage you to submit your code as is if you exceed that time (you can attach a short description of how you would finish the challenge instead).

Question 1

Choose a **relational DB** and **create a data pipeline** to load and transform the dataset. The tables should be optimised for readability and ease of querying by product analysts.

Bonus: Follow Dimensional Modeling principles to design your schema (i.e. define fact and dimension(s) tables).

Question 2

Create an **availability_snapshot** table that would store the amount of active professionals per day (reminder: an active professional is a professional who is “able to propose”).

Example (based on the event_log.csv dataset):

Timestamp	**Professional id**	**Event type**
2020-03-02 07:30	1	created_account
2020-03-02 08:00	1	became_able_to_propose
2020-03-03 07:30	1	became_not_able_to_propose
2020-03-04 07:30	1	became_able_to_propose
2020-03-03 09:00	2	created_account
2020-03-05 10:00	2	became_able_to_propose
2020-04-07 07:30	2	became_not_able_to_propose

The 1st professional was able to propose (=active) on 2020-03-02 and 2020-03-04 onward (including).
The 2nd professional was able to propose on 2020-03-05 and 2020-03-06.

Based on the above, the **availability_snapshot** would be:

Date	**Active professionals count**
2020-03-02	1
2020-03-03	0
2020-03-04	1
2020-03-05	2
2020-03-06	2
2020-03-07	1
2020-03-08	1

2020-03-09	1
2020-03-10	1

Notes:

- The date range for the table should be between **min(timestamp)** and **2020-03-10**.
- If a professional became able (or unable) to propose on a specific day, we consider that they were able (or not able) to propose throughout **the whole day**, regardless of the exact time the event was triggered.

Good luck!
Werkspot Dangineering Team