Welcome to the Product Analytics Technical Test

Being hands on is one of the key requirements for this role. You need to complete this test in a **maximum of 3 days** from the day you receive it. **The faster you deliver it, the better.** When you are done, please submit the results through Greenhouse.

If you have any technical questions feel free to reach Javier (javier.granda@glovoapp.com), and we will get back to you as soon as we can.

What we value:

- A clean coding style
- Efficient solutions to the problems given
- SQL readability and scalability
- Idiomatic use of Python
- Appropriate use of Python's built-in functions and standard libraries

1. KPIs (25 points)

In your opinion, what are the three main KPIs for Glovo? Ranked by decreasing importance. Explain your choice and try to make an educated guess of their value. Provide a step-by-step explanation of your guess. How would you improve them? ***Note:*** *Ignore pure financial KPIs that apply to every business.*

2. SQL (25 points)

You have the ***customer_courier_chat_messages*** table that stores data about individual messages exchanged between customers and couriers via the in-app chat. An example of the table is below:

Sender app type	Customer id	From id	To id	Chat started by message	Order id	Order stage	Courier id	Message sent time
Customer iOS	17071099	17071099	16293039	FALSE	59528555	PICKING_UP	16293039	2019-08-19 8:01:47
Courier iOS	17071099	16293039	17071099	FALSE	59528555	ARRIVING	16293039	2019-08-19 8:01:04
Customer iOS	17071099	17071099	16293039	FALSE	59528555	PICKING_UP	16293039	2019-08-19 8:00:04

Courier Android	12874122	18325287	12874122	TRUE	59528038	ADDRESS_DELIVERY	18325287	2019-08-19 7:59:33

You also have access to ***orders*** table where you have an order_id and city_code field.

Your task is to build a query that creates a table (***customer_courier_conversations***) that aggregates individual messages into conversations. Take into consideration that a conversation is unique per order. The required fields are the following:

- Order id
- City code
- Timestamp of the first courier message
- Timestamp of the first customer message
- Number of messages from courier
- Number of messages from customer
- The first message sender (courier or customer)
- Timestamp of the first message in the conversation
- Time (in secs) elapsed until the first message was responded
- Timestamp of the last message sent
- The stage of the order when the last message was sent

Make your query scalable and readable!

3. Experiment (25 points)

We would like to measure the impact of increasing the order delivery fee in a given city (not considering products value), from €1.9 to €2.1, on company KPIs.

You are asked to design the experiment from the conceptual stage, plan empirical analysis, and present the recommendations depending on the results of the experiment.

Explain in detail how you would approach this task, focusing on the following:

- What kind of test would this require?
- Would you test this in new users only or all active users? Why?
- What assumptions you would make, and how would you test whether or not these assumptions hold?
- What approach would you use to determine the duration of the experiment?

- What KPIs/metrics would you choose to evaluate the success of the test?
- What steps would you take to analyze the results of the test?
- What would your recommendations be depending on the results of the test?

4 Exploratory Data Analysis (Statistical Programing Exercise) (25 points)

You've been given a **data set** about orders coming from Fake partners in the app. Fake partners are the stores that are not integrated with Glovo directly so our content team manages their product catalog and prices for them. Fake partner orders are charged to the customer upon delivery and in many cases there is a mismatch between the total amount at checkout in the app (*products_total*) and what the courier pays at the store (*purchase_total_price*) causing many problems. When the *products_total* is lower than *purchase_total_price* we call them ***under-authorized orders*****,** otherwise is a correctly authorized order. We want to move away from *charge-on-delivery* to an *authorize-and-capture* model but we first need to understand the price fluctuation of past orders to know the risk of doing so.

Dataset description:

- order_id
- activation_time_local: local time when the order was activated
- country_code
- store_address
- final_status
- payment_status
- products: number of products in the order
- products_total: total amount at checkout (€)
- purchase_total_price: amount the courier paid at the store (€)

Your task is to perform an EDA process (R/Python) with this data to answer the following questions:

1. What percent of orders are under-authorized?
2. What percent of orders would be correctly authorized w/ incremental authorisation (+20%) on the amount at checkout?
3. Are there differences when split by country?

4. For the remainder of orders that would be outside of incremental auth what values would be necessary to capture the remaining amount?
5. Which stores are the most problematic in terms of orders and monetary value?
6. For under-auth orders is there a correlation between the difference in the prices and the cancellation of the order? In other words: Is an order more likely to be cancelled as the price difference increases?