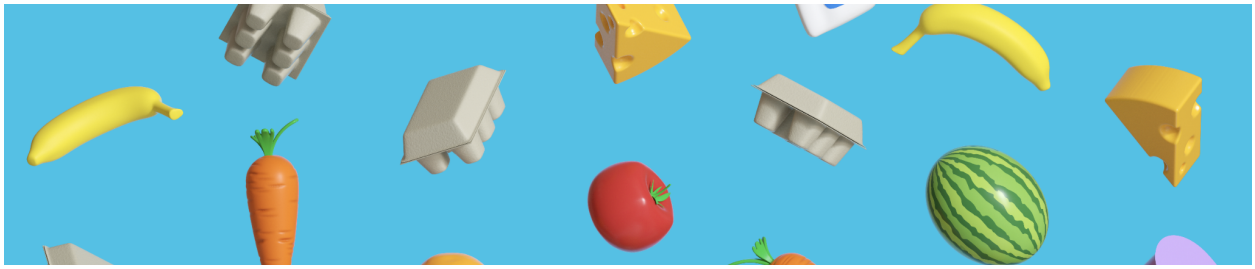

Radius Reduction Assignment

We are delighted to continue the recruitment process with you. Now is your time to shine. We've prepared a task for you so that you can show your skills and allow us to understand how you approach problems in general. Good luck!

Exercise

In the assignment you get to look at our delivery radius & purchase data and build a solution to understand how changes in a city's delivery radius affect purchases.

Delivery radius can be seen as the maximum delivery distance from the venue to the customer. For instance, a radius of 3500 meters would mean that a specific venue would only be available for delivery, if the customer's location is within 3500 meters from the venue. Delivery radius can be changed for many reasons. For example, in the short term we might reduce it temporarily to accommodate for large order volumes in the central area, but in the long term we could expand the delivery area to cater to more users. Changes to delivery radius are called radius reductions or expansions.

We've prepared the data to be almost ready for your needs, but as in real life, you will need to do some more transformations to it. You are free to use any tools you wish, as long as we see your SQL skills in action for both tasks. One way is to use Jupyter Notebook together with any SQL package, or you can opt for some other tool like DBeaver.

About the data

The data consists of two csv files:

- purchases.csv
- delivery_radius_log.csv

The data is an artificial dataset that could be generated by Wolt's applications. The first dataset contains purchases that happened in a few cities during 2022. The second dataset contains changes in the same cities' delivery radiuses around the same time. Changes in delivery radiuses can be understood as the changes in maximum delivery distance between a venue and a customer, which Wolt uses to optimize delivery operations in a city. For example, as mentioned earlier, during the hours when order volume soars, we might temporarily reduce the delivery radius to ensure swift deliveries to our customers.

In **purchases** dataset we have the following fields:

- Purchase ID: Unique ID of the purchase.
- Time Received: Timestamp of when the customer placed the order in UTC.
- Time Delivered: Timestamp of when the order was delivered to the customer in UTC.
- End Amount With VAT Eur: Total price of the purchase that the customer paid in euros, including VAT.
- Dropoff Distance Straight Line Meters: Total straight line distance in meters from the venue to the customer delivery address.
- Delivery Area ID: ID of the delivery area where the purchase took place.

In **delivery_radius_log** dataset we have the following fields:

- Delivery Area ID: Unique ID of the city's delivery area.
- Delivery Radius Meters: Maximum delivery radius of the city in meters.
- Event Started Timestamp: Timestamp when the delivery radius was changed in UTC.

Task 1

In the first task you'll work with the delivery radius log dataset. Given this delivery radius change log, we would like you to detect at any given time what is a temporary reduction (or increase) of the delivery radius and what is the "default" (more permanent) delivery radius. For this exercise, you can assume that the default radius at any given time is a radius that has lasted for at least 24 hours uninterrupted.

We would like you to produce a dataset(s) and answer the following:

- What are all the default delivery radiuses for the delivery areas during the timeframe provided? Keep in mind that each area can have multiple default radiuses in the given dataset.
- How many hours of radius reductions with respect to the the default radiuses have we had during the timeframe provided for each delivery area?

Please give answers in numerical values to the above questions.

Task 2

Now that we know the default delivery radiuses and times when the delivery radius was reduced, we would like you to create **a derived dataset aggregated to hourly level** that can be used to analyze delivery radius reductions and purchases in the areas for any hour in 2022. Build the dataset so that anyone could query the data without writing further joins or calculations and would be able to answer the following questions with a simple SELECT statement:

- How many purchases and how much revenue (End Amount With VAT Eur) do we produce during the hour?
- How long do the deviations (reductions) from default radius last during the hour? How many times have we modified the radius during the hour?
- How do these hourly values compare to the previous week for each area? This is just a simple week-over-week percentage difference for each of the above-mentioned four measures.

We want to emphasize that all three questions should be answered with the same aggregated dataset, meaning for instance that even the week-over-week differences are pre-calculated. Please note that for this task it is enough to only create the dataset and you are not expected to answer these questions. We only wish to see the code which creates this table and a sample of a few rows from the resulting dataset.

Try to solve both tasks in a programmatic way that would apply to all scenarios (e.g. if given a dataset of another delivery area or more purchases) and don't just provide the result tables, we also really want to see your code! Also note that there may not be one single correct answer to these tasks and one can approach them from many different angles. Thus, in addition to solving the tasks, please also answer the following questions to clarify your solution:

- What assumptions about the data have you made to produce the dataset?
- Why did you decide to go with this particular approach and what could be the pros and cons of applying it?
- How could the solution be improved if given more time and data?
- What strategy would you use for updating the dataset from task 2? Consider how often the default radius should be calculated, do we need to truncate the table before updating etc. Please assume that upstream data (i.e. purchases & delivery_radius_log) is streamed to the tables, so changes arrive near real time.

Please compile your results and explain the reasoning behind the solution in presentation format (e.g. PDF, Powerpoint or Jupyter Notebook) and the underlying code. The presentation should be sufficiently self-explanatory for us to follow your thought process when solving the assignment. Return the assignment in **English**. Good luck!