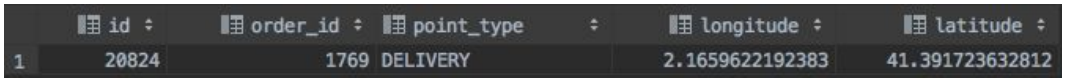
1. Let's say you have two tables: **orders** and **order_points**.
   Create an SQL query that shows the distance between the courier starting position and the pickup point, as well as the distance between the pickup point and the delivery point.

   The **orders** table has 1M+ rows; here's the first row:

	id	customer_id	courier_id	acceptance_latitude	acceptance_longitude
1	14019	18331	386343	39.46069070000001	-0.3649919

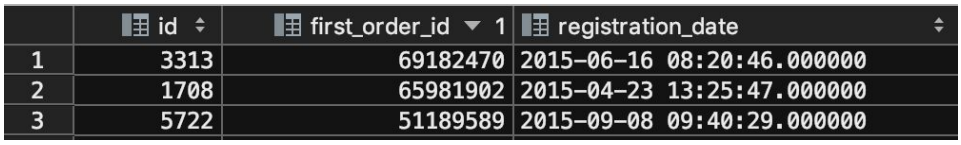
   The **order_points** table also has 2M+ rows. As FYI there are two types of point, **'DELIVERY'** and **'PICKUP'**. here's the first row:



	id	order_id	point_type	longitude	latitude
1	20824	1769	DELIVERY	2.1659622192383	41.391723632812

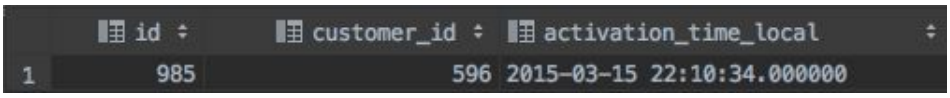
2. Build one SQL query to create a cohort of Signup to First Order and show the result. The objective of this cohort is to see, out of the users that signed up in Week N, how many did their first order in Week N+1, N+2, N+3...

   The **users** table has 1M+ rows; here's the first three rows:



	id	first_order_id ▼ 1	registration_date
1	3313	69182470	2015-06-16 08:20:46.000000
2	1708	65981902	2015-04-23 13:25:47.000000
3	5722	51189589	2015-09-08 09:40:29.000000

   The **orders** table has 1M+ rows; here's the first row:



	id	customer_id	activation_time_local
1	985	596	2015-03-15 22:10:34.000000

   The output does not require to be in a cohort format. The end user could potentially use the pivot function from Excel or Google sheets to do so.

   Make it scalable

3. Let's say you have one table: **events**. This table stores all the events from the app, from screen impressions to any kind of user activity. For instance, if we click in the **Restaurant** category, that will store a screen impression event in the table.

   This table has a field **attributes_json** with a json formatted string with information regarding the Restaurants shown to the user.

Table ***events:***

id (integer)
creation_time (timestamp)
attributes_json (text)

A key-value example of the attributes_json would be as follows:

“country_code”:”ES”
“city_code”:”BCN”
"storeIds":
"[31504,22479,49497,23431,41173,20684,31508,20683,40028,22478,52359,31431,39572,49496,48437,50763,53473,48814,53611,20666,22340,20681,46840,54062]"

Considering the available information above, build an SQL query that retrieves the average restaurant position per month, country and city. Consider we have only 100 available restaurants in the Food category and the json field stores all of the positions.