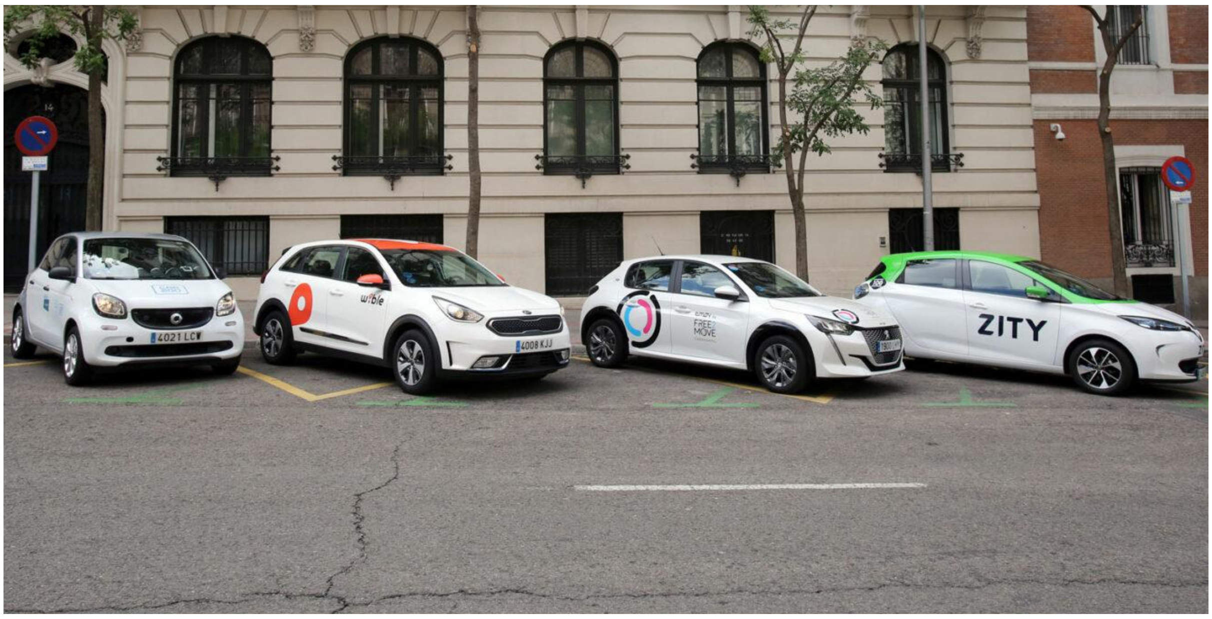
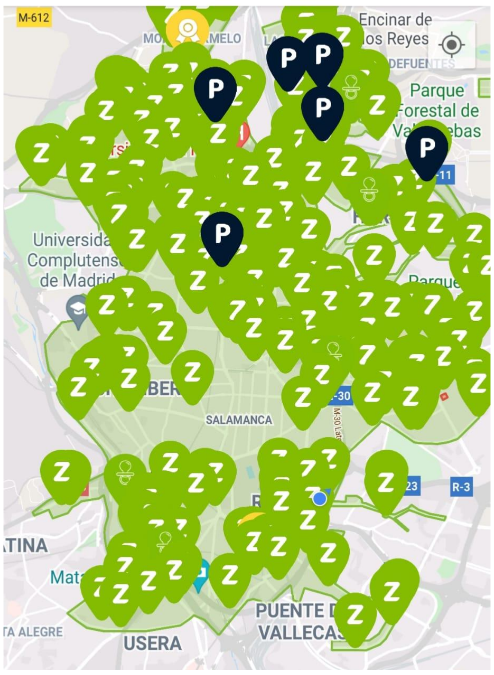
# Mobility services demand prediction



Alberto González Gómez
Septiembre - 2022

# 1. Introduction:


M-612
Encinar de los Reyes
Parque Forestal de Valdebebas
Universidad Complutense de Madrid
SALAMANCA
USERA
PUENTE DE VALLECAS
R-3

In the last few years the number of car sharing/moto sharing/… companies has increased so fast.

Now, in the city of this study, Madrid, there are 5 companies of carsharing (gotoglobal, free2move, wible, zity and share-now ), 4 companies of motosharing (gotoglobal, ecooltra, Acciona, Movo) and a few more of scooter sharing.

Between all these options it is curious that some places with a high demand like the Salamanca neighborhood usually do not have enough vehicles for all the customers.

Also it is so important to know that there is no recolocation of the vehicles. The vehicles are available for the next customer in the place where the last customer leaves it so it could be so important to analyze because some positions can be a sink without demand.

For all of this I think that it could be interesting to analyze the demand of the service, analyzing when the people use the service, in which parts of the city…

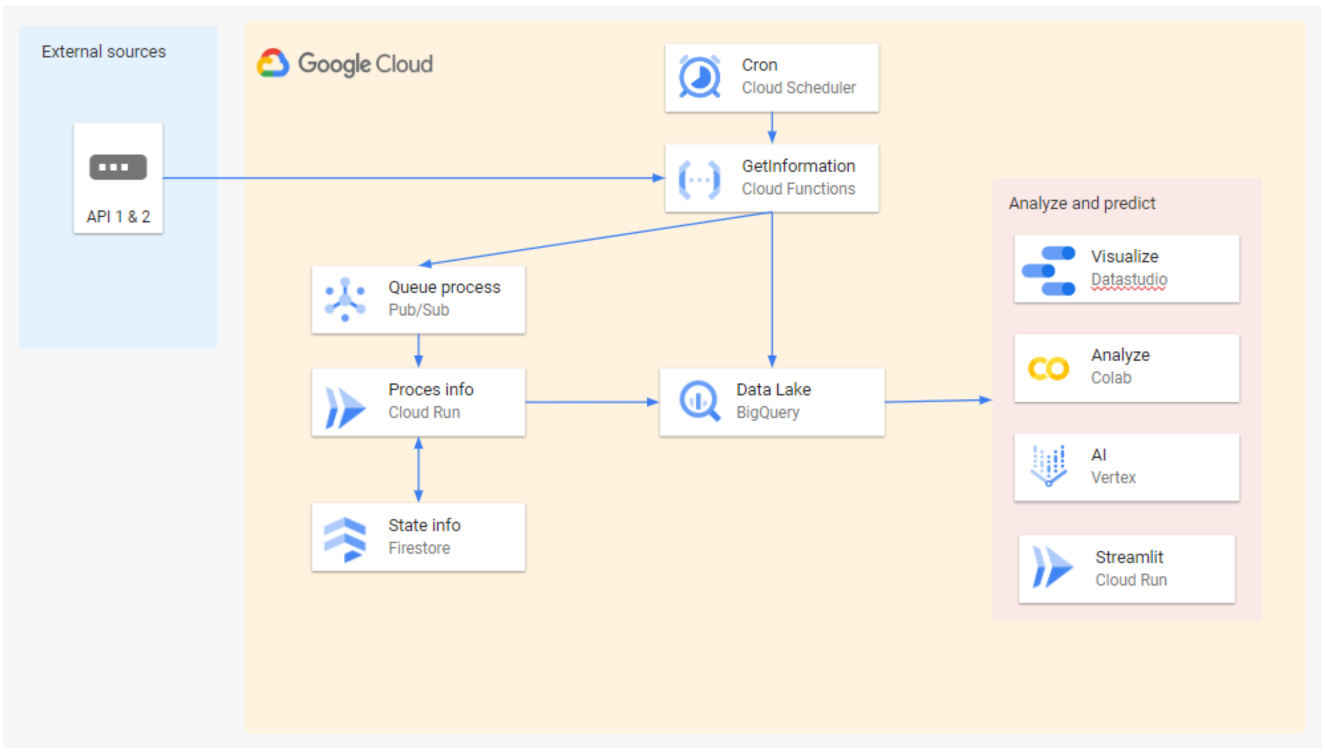
# 2. Architecture

## ¿Why in the cloud?

All the data and workloads are hosted in Google Cloud. Use a cloud provider is a key point for several reasons:

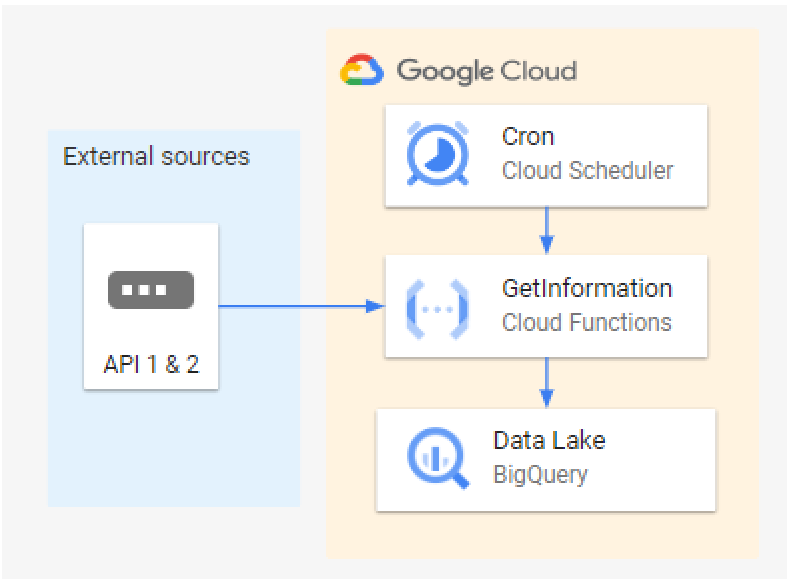
- **Availability**:The data acquisition is 24/7.This involves having a server always on. Using a personal computer or a simple server would be so expensive and not failure tolerant so a Cloud Function was chosen. (More info about cloud function will be explained in data acquisition part)
- **No infrastructure**: With cloud it is not needed to manage the infrastructure so the time to build and maintain the infrastructure is reduced enabling you to put all the efforts in development.
- **Big data**: In this project, we manage a lot of data. The table of points has 70.000.000 rows, a total of 20GB of data. With BigQuery complex queries were solved in seconds and you do not need to have a big system because they could give you a massive compute capacity but you only pay for that the amount of time that you are using it.

## Architecture diagram:



In order to be more clear the architecture is explained piecemeal in detail during the part of data acquisition of this document.

# 3. Data acquisition:



There are two origins of data. One is a aggregation os sharing service  (source A or 1) and the other is a car sharing service (only one an identified as source B or 2)

## Retrieve data:

The process starts with a new call of Cloud Scheduler every 15 minutes / 1 hour.
The process launches a Cloud function and this cloud function calls the API of data (n times depending on how this API is defined) and formats the data to save in BigQuery as a facts table.

The data retrieved of the APIs is similar: A list of the location of vehicles at the time of the request.

A example of data received is:

- service: brand of service
- idVehicle: id provided by company
- plate: plate of vehicle
- energy: energy of vehicle
- latitude and longitude
- And many other business variables like: price, type, category,...

Is so important to define a common structure for both data sources to make it easier to build the process and analysis after without major changes.

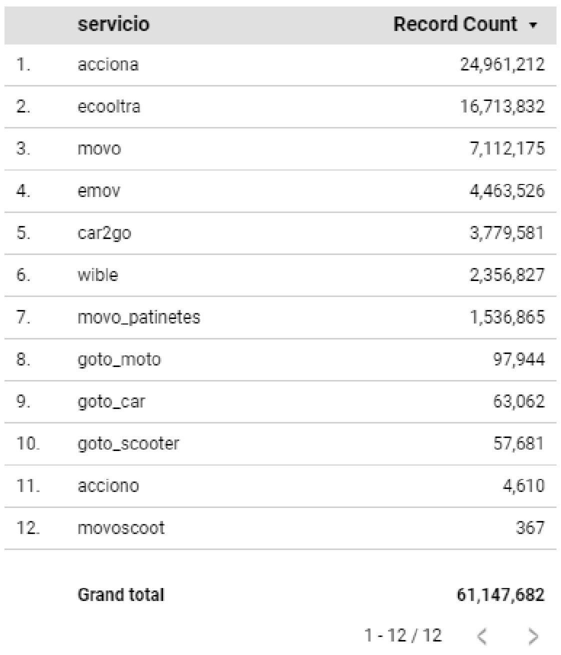
## Data from aggregator:

The data of the aggregator is the oldest and is bigger than the other source.

The code of the cloud function is in github, the cloud function is called function-1 and the data is saved in the partitioned table bulkData.

This table has data from april of 2021, 17 months. During this time, 60.000.000 rows were generated and this source give us info from different mobility service providers:

- Car
    - Wible
    - Emov
    - Car2go
    - Goto
- Moto
    - Movo
    - Acciona
    - Ecooltra
    - Goto
- Scooter
    - Movo
    - Goto



| | servicio | Record Count ▾ |
|---|---|---|
| 1. | acciona | 24,961,212 |
| 2. | ecooltra | 16,713,832 |
| 3. | movo | 7,112,175 |
| 4. | emov | 4,463,526 |
| 5. | car2go | 3,779,581 |
| 6. | wible | 2,356,827 |
| 7. | movo_patinetes | 1,536,865 |
| 8. | goto_moto | 97,944 |
| 9. | goto_car | 63,062 |
| 10. | goto_scooter | 57,681 |
| 11. | acciono | 4,610 |
| 12. | movoscoot | 367 |
| | **Grand total** | **61,147,682** |

(Image obtained from Datastudio → Origin 1 - Points of data / company)

## Data from carsharing company:

The code of the cloud function is in github, the cloud function is called cf-getfree2move and the data is saved in the partitioned table bulkData_b.

The data of the carsharing company is much more recent, starting the 12 of Jun, so 2.5 month of data. (The reason for this new data source will be commented on in the data quality section).
During this time 12.000.000 rows were generated from 7 different cities.

## BigQuery:

Before moving to the next section it is important to give some reminders about how to work with the database BigQuery instead of other databases.

BigQuery, as we commented before, is a database so powerful that it enables you to make queries of GBs of data in seconds but this velocity can make you not notice how much data is your query consuming.
In BigQuery you pay principally for storage and queries.

There are some good practice to reduce the consume:

- BigQuery is a columnar database: User * in Select is not recommended, it is better think first what columns are needed for your query.
- Partition tables: You can select partitions by a column or by ingestion date to query only specific parts of the database.

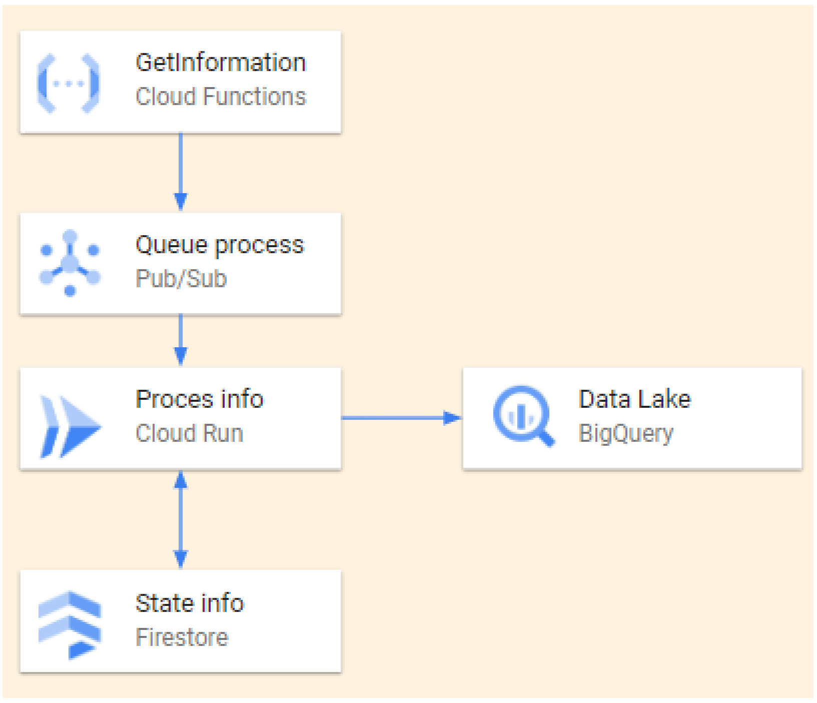
For example, in our case, we need to make 1 query every 15 minutes, that's are 2.880 request per month, with 20GB per query (all our bulkData table), a total of 55TB analyzed → 350€
Doing the same query only using the partition of today, one query are 200MB, a total per month of 0.5TB → 3€

# 4. Data processing:

After obtaining the data is needed do a postprocess of the data before have them ready to do analysis and predictions.

The data that we have now are rows of positions but we want trips.

To obtain this information the solution is cloud run.
Every time a new data is processed via pub/sub (now we are using cloud run but pubsub is a better solution because it maintains the order of the messages received) a new cloud run is launched.



When Cloud Run starts get the last position of every vehicle that is stored in the firebase and charge it in memory.
After Cloud Run read from BigQuery the new data of vehicle positions and compare both, if there is a difference write the TRIP in BigQuery.
At the end save all the new locations in Firestore again.

The data transformed is stored in trips_a for origin A and trips_b for origin B, in this case the most important columns of the data are:

- plate
- energy_start
- energy_end
- latitude_start
- latitude_end
- logitude_start
- longitude_end
- distance
- recharge
- And many other business variables like: price, type, category,...

ATTENTION: The data columns of trips_a are wrong because a failure in the calculation of date, the accuracy date is epochTime.

TripsA code is in github and google cloud.
TripsB code is in github and google cloud.

## Special columns

plate, energy_start, latitude_end,... are rows that comes directly from bulk data but there are two columns that are calculated:

- Distance: We use a math formula to calculate the distance of the origin and end of the trip because we need to remove the false trips (GPS system are not 100% precise and a change in latitude-longitude can be a error in the system instead of a trip)
  The limit to consider a valid trip is 200 meters, this number has been chosen after checking manually some trips.
- Recharge: Vehicles need to be recharged to be ready for the customer. To remove this data from the dataset (these trips don't answer to a real demand, answer a specific situation at a punctual moment).
  - The math operation to calculate a recharge trip is a increase of 20 points in battery
  - A lower level of battery should not be use because the calculation of remaining battery is not precise in electric cars because depend also on the recent effort (for example a trip with a negative difference of level over the sea)
  - An amount of time should not be used because some companies don't use electric cars and these cars do not have long hours of charge.

## Other architecture option

We put a big quantity of efford building a Dataflow process (similar than Spark) to calculate this change in real time but this approach has no sense for this reason:

- We are trying to use the MapReduce process of a task that has common variables. The location of the vehicles are a variable shared between all the processes so it is not so easy asilate it to do parallel processing. Also doing this parallel processing the velocity increase will be low because the majority of the process time is reading and storing in the database.

An approach using Dataflow will be valid for example if we want to evaluate how many trips have been done to trigger an alert if in the last 30 minutes there has been less than a specific value.
Also it is not easy to use Dataflow because the pricing of having always on Dataflow is high for the use of this project (in a real project could be a better option).

## 5. Data quality:

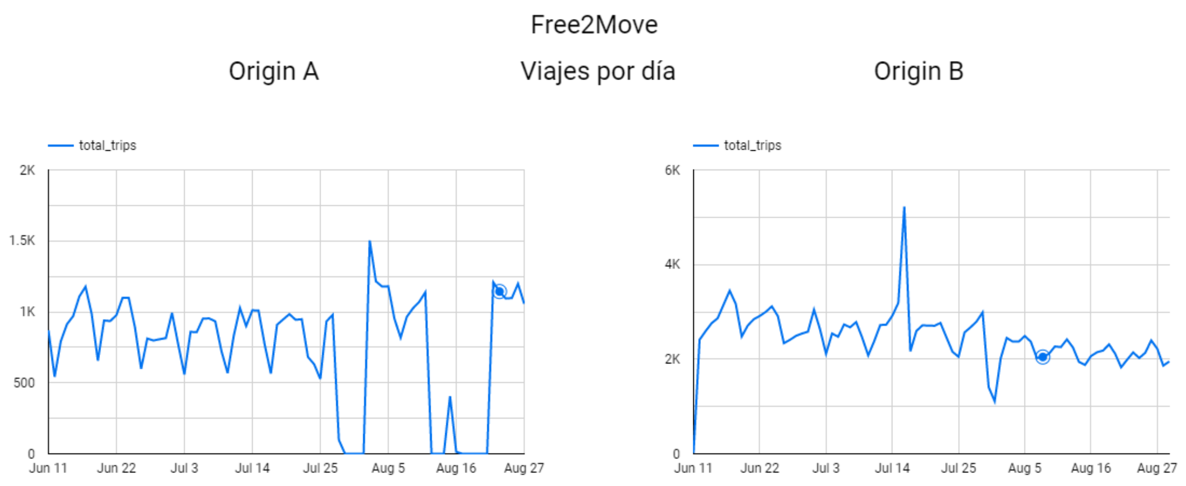
We are obtaining information from a source that is public but has no agreement services, SLAs or documentation, so it is important have some dashboards to ensure the quality of the data,

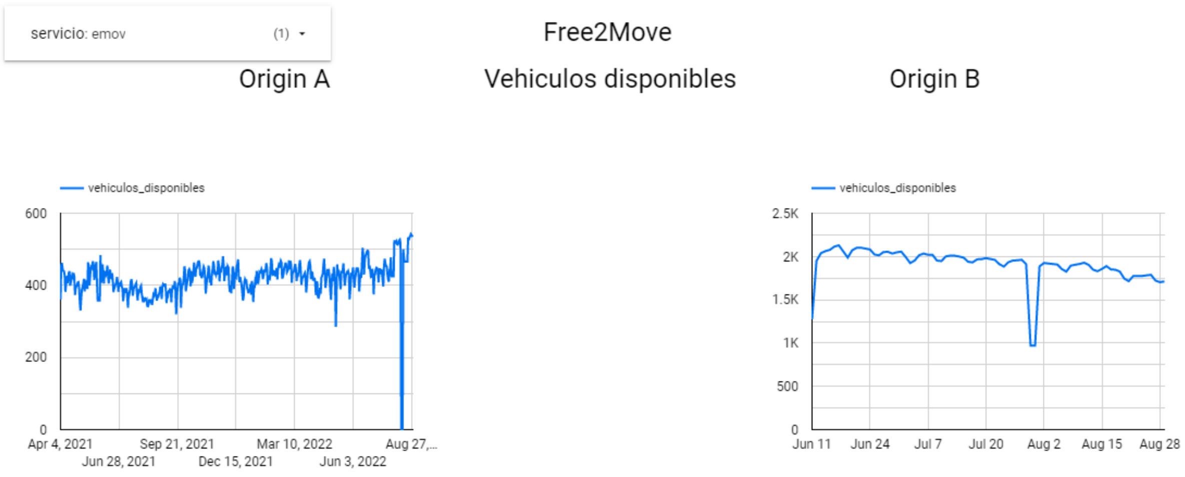
### Real time data:

One of the objectives of data acquisition and data processing is to get the actual situation in real time to continuously make predictions instead of batch prediction or only predict closed datasets. → This point is achieved, the data is refresh every 15 minutes.

### Between data providers:

To evaluate the quality of the data one idea is check two different data sources showing the same information: Data of the carsharing company Free2Move
(Origin A and Origin B are the sources - In datasource A free2move is called emov because is the old name of the company)

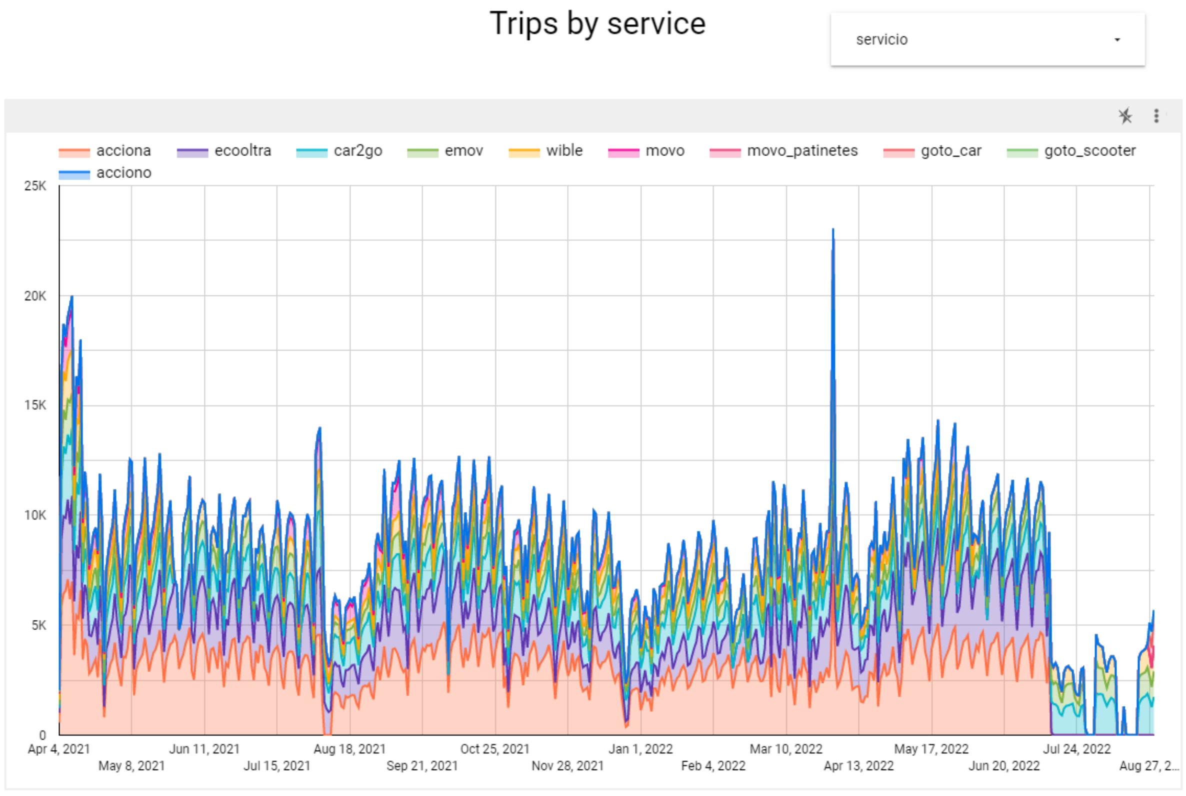




(Image obtained from Datastudio → Compare origins - Trips by day)
As we can see both charts have very different data. The data of source A(aggregator) is less than half of the trips that source B (company) gives us so the quality of source A is bad.

## Provider A:

Now we are going to check the quality of the data origin A (aggregator).



(Image obtained from Datastudio → Origin 1 - Trips by service)

In the above image we can see the number of trips per provider per day of year.

As we can see we have another problem here, our data provider is not giving data of some companies since the half of july 2022. After checking the official app we notice that it is a problem of the service provider not us.

Also there was some service interruption of 2-3 days last month, as we can see at the end of the graph.

## Severity of detected errors:

After finding these errors we need to evaluate the severity of this errors to make a decision about if this data is valid or need to be removed from this analysis.

Interruption of data:

- This error is catastrophic because we are going to use temporal series to predict the demand of the days in the future and without data there are no predictions.
- Fortunately this error seems to be temporary and may be influenced by the holidays of the team. On the other hand these interruptions can mean the end of the service and we should be prepared for this case.

Reduce of mobility providers:

- This error is catastrophic because if we use to estimate 10 providers and after that number of providers is reduced to less than half the predictions will not follow the same distribution.

Difference of data between aggregator (source A) and provider (source B):

- This seems to be a huge error at the start but it is not so important if we analyze it in detail.
  At the start the idea was not to use data A but after checking the data we can conclude that the data lost is distributed proportionally (like if we slide data for training and test) for this reason we can use this data (knowing not use for example to say the number of real trips).
  For this reason, and as we will see in the analysis part, this datasource will be selected to do all the estimations instead of one provider data because the amount of trips of one provider is too low to do an analysis.

# 6. Analysis:

Data analysis has been done using different tools:

- **Data Studio**: Used to do fast visualization to ensure that the data is received and processed ok.
- **Vertex AI:** AI Google Cloud solution. Used to train models to check if the accuracy is better than models from notebooks.
- **Colab:** Notebooks environment. There is no special reason to use Colab instead of a local option.
- **BigQuery:** After a fast prototype with colab some data frame were rebuildid with BigQuery in order to have real time insights used in Data Studio or other notebooks.

## Datastudio:

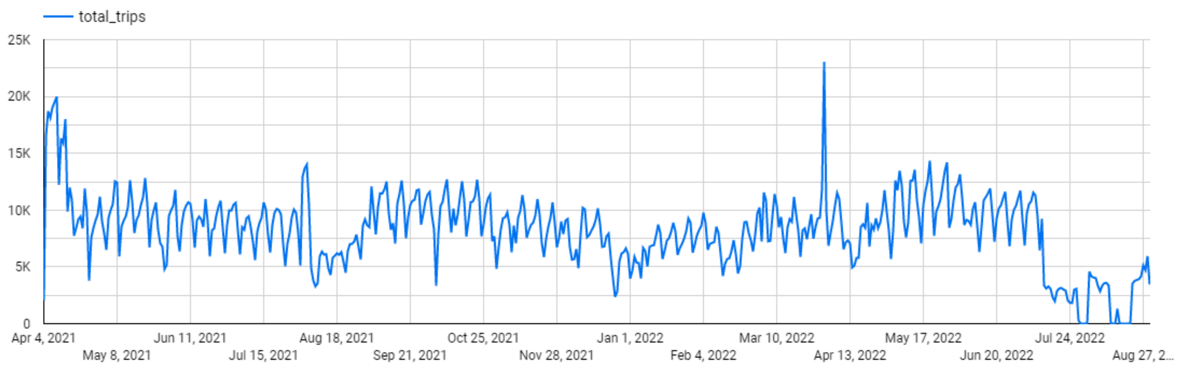
Can be found in this url: Datastudio
As explained in the introduction Datastudio was primarily used as a fast tool to check the data quality and distribution.

In this document we are going to explain some diagrams of the data studio but not all please check it by yourself.

Datastudios dashboards are organized in:

- Origin 1 → Related to origin 1
- Origin 2 → Relate to origin 2
- Compare origins → Check quality of data
- Other/Old visualizations → Visualization with possible failures

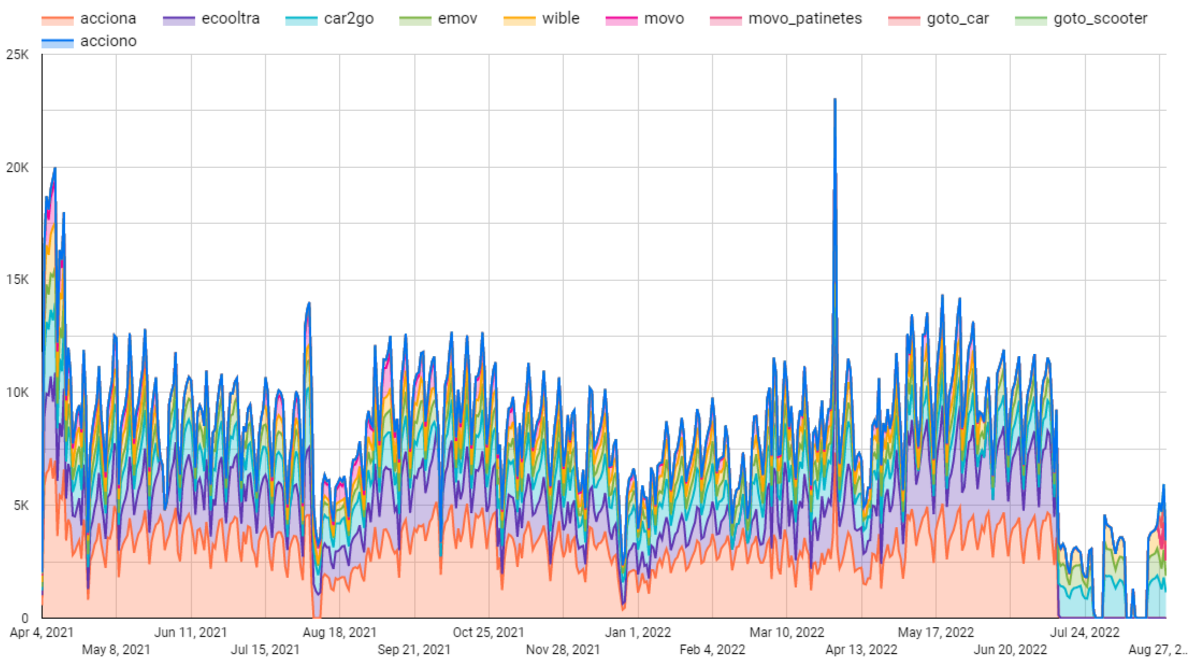
### Origin 1 - Trips by day:



This dashboard is so useful because is easy analyze the quality of the data:

- On 1 April 2022 we can see more trips than expected, this can mean more detailed data on this day.
- Also at the start of august 2021 there was a extrange data event because this day has more trips than others and next days a very low amount of data.

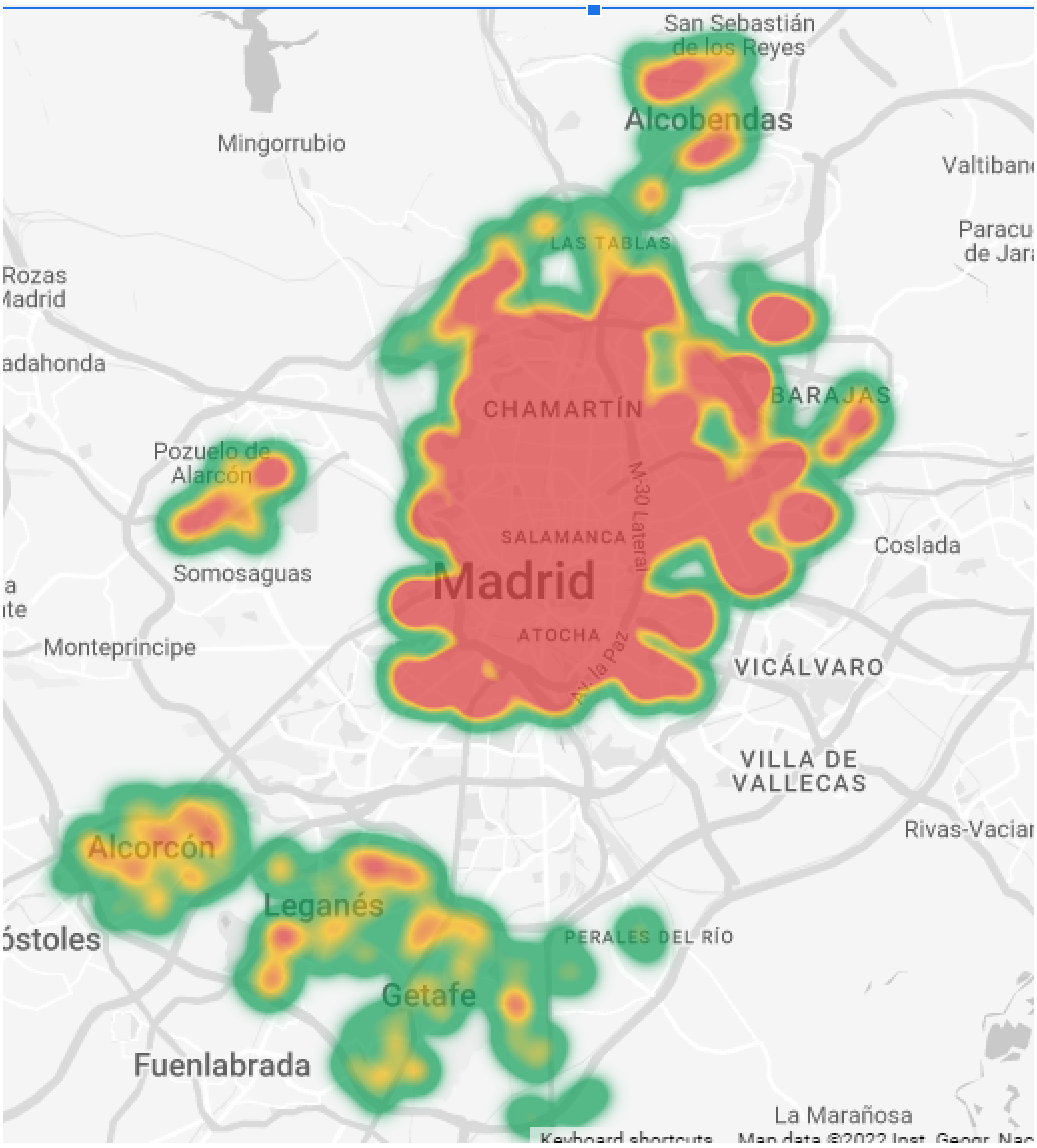
## Origin 1 - Trips by service:



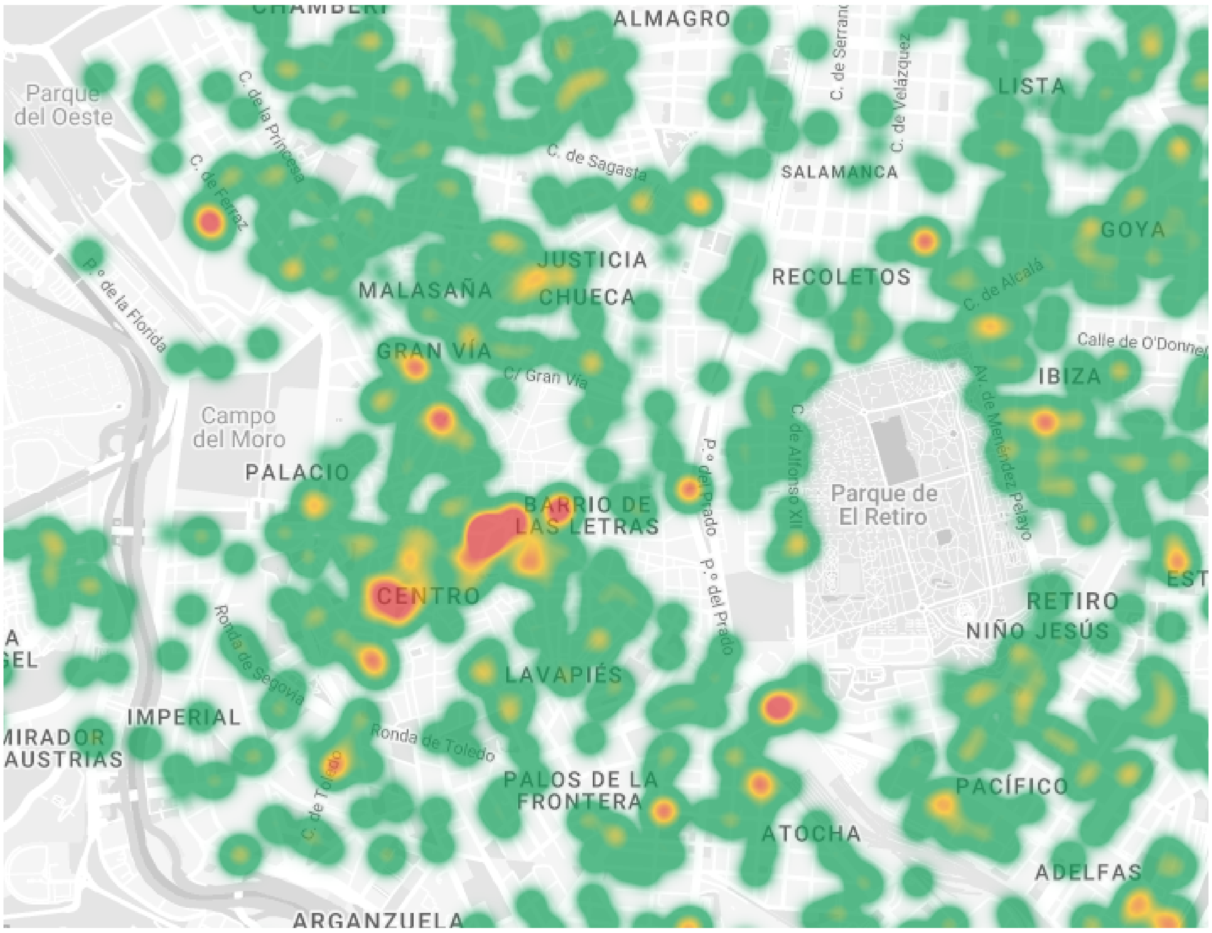
This dashboard is the same as before but broken down by providers. This representation was hugely important to notice the problema after july 2022.
The new colors at the end of the representation are the new vehicles of goto, added to data at the end of august 2022.

## Origin 1 - Location of vehicles:

This is a good visualization to check the areas of service of different providers.

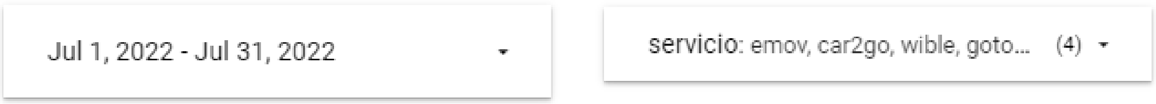


With a far zoom it is easy to know where the services are provided.



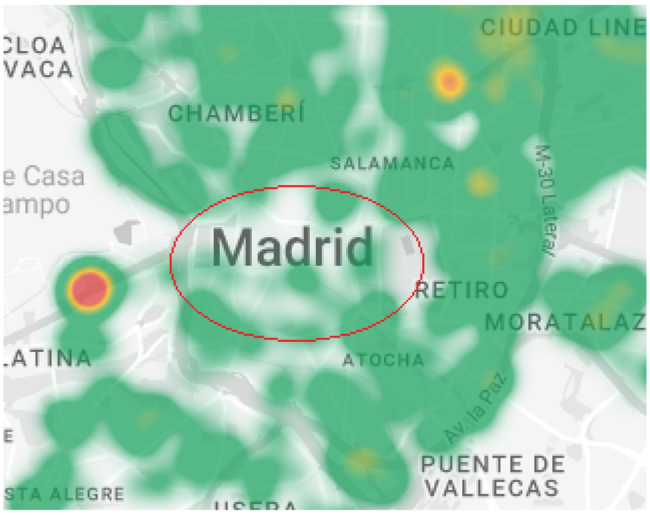
With a closer visualization it is easy notize important points of the city like Atocha (if you have been there, Atocha always is full of motorcycles), Templo de debod or the city center.

If we chose another selector (only cars) the image would change drastically because in the city center it is easier to park a moto than a car.



Jul 1, 2022 - Jul 31, 2022 ▾

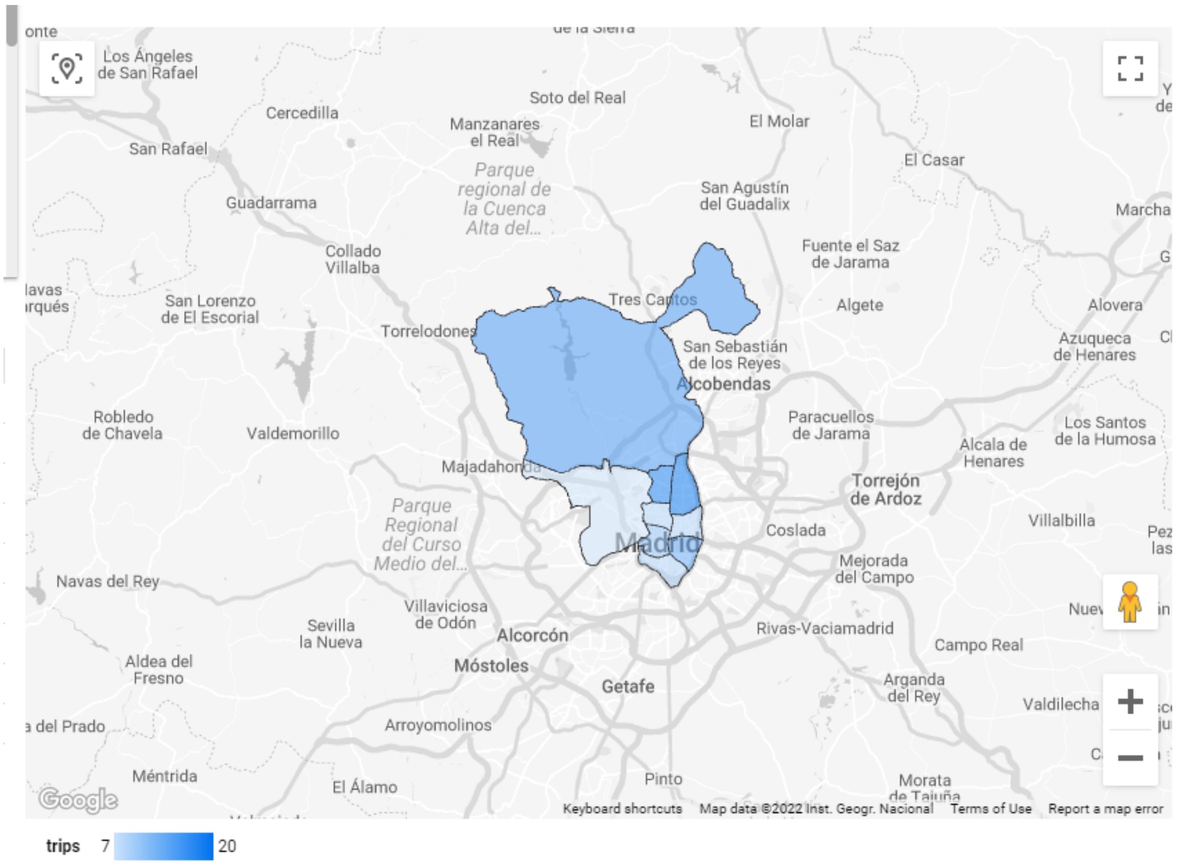
servicio: emov, car2go, wible, goto... (4) ▾



In the first image there are a lot of vehicles in the city center (more than outside), in the second image it is the opposite. The reason is that it is easier to park a moto in the center than a car.

## Origin 1 - Trips by district:

This view, as we will check later, is the view of the prediction.
Shows the number of trips that arrive at a specific hour in a district.



## Compare origins - Trips by day:

(We have shown this view before in data quality)

## BigQuery:

In BigQuery we develop many views.

The views are distributes following: (start with)

- V1 → Views related to trips A
- V2 → Views related to trips B
- V → Old views

There are 24 views for data A and 12 views for data B. These views were built after a discovery in a notebook as a start for the next notebook → With this it is not needed to save data between the notebooks, only do another query.

In this document it is not possible to explain all the details of the queries because it will be a hard task and boring for the reader. If you need some details of the view please follow the details of the query.

Some queries are especially important because they are used as the beginning of a notebook, these queries will be explained in that notebook.

## Vertex AI:

Google Cloud has an AI solution built in.
In this project the service Prediction Temporal Series was used to try to predict the next demand. The price of this option (30€/hour training with a minimum of 1hour per training makes that we didn't explore this option enough, so there are no results to show. Although some models were trained and can be explored in the cloud console (in region europe-west-1 specially)

## Notebooks:

All the notebooks were developed in Colab. There is no specific reason for that in terms of processing or permissions, the only reason is comfortability, because it is easier to develop without the need to take care about dependencies.

The notebooks have been developed sequentially solving different requirements. In this document we are going to detail the purpose and the main topics of each notebook. Reading this part sequentially you can find how we have been reasoning all the process of discovery and analysis.

(In this section we will refer to Datasorce as Data 1 and Datasource B as Data 2)

### Final data:

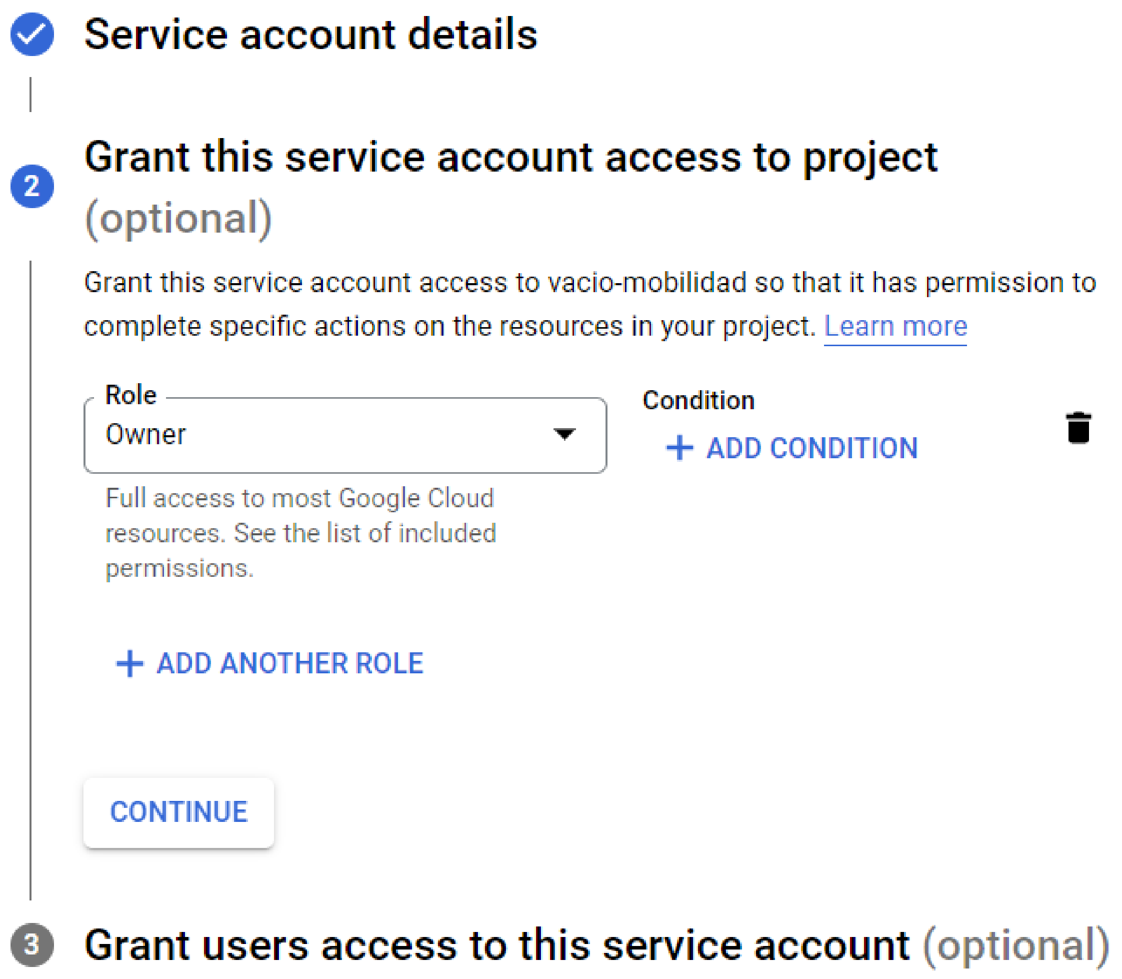
The final data and model is:
Data source A - Aggregator → Only data from the tree providers that still have service nowadays.
Grouped by district to have more data to estimate.
Predicted using a RandomForest.

### 01 - Mobility - Data 1 - Connect to BigQuery

The objective of these notebooks is only to test the conexion between Colab and BigQuery.

Here I want to explain how to relaunch the notebooks. All the notebooks need conexion with Google Cloud to get the data, this is done using a service account.

Please, go to IAM → Service Accounts on Google Cloud, create a service account and add a Owner permissions.



Service account details

2 Grant this service account access to project (optional)

Grant this service account access to vacio-mobilidad so that it has permission to complete specific actions on the resources in your project. Learn more

Role: Owner

Condition: + ADD CONDITION

Full access to most Google Cloud resources. See the list of included permissions.

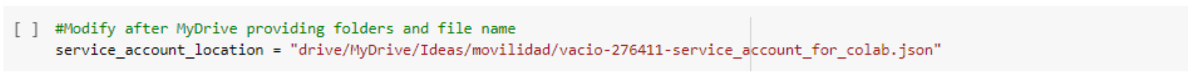
+ ADD ANOTHER ROLE

CONTINUE

3 Grant users access to this service account (optional)

This is not a good practice of GCP but as a fast test will be the fast option.

After selecting the service account created, go to manage keys to create a key, download to your laptop and upload to Google Cloud.
Every notebook has a cell at the start where you need to change the location of this file in drive.



```
[ ] #Modify after MyDrive providing folders and file name
    service_account_location = "drive/MyDrive/Ideas/movilidad/vacio-276411-service_account_for_colab.json"
```

## 02 - Mobility - Data 1 - Connect to BigQuery

In this notebook we are going to analyze the data using a location cluster.

When the dataset 1 is created we added a column called cluster, cluster_longitude, cluster_latitude. All the data is divided in squares of location. (The size of the square is 0.01

using latitude and longitude. Example: on square is build of 4 points 40.485,-3.715 - 40.485,-3.705 - 40.475,-3.715 - 40.475,-3.705 - 40.485,-3.705)

The data used in this notebook is generated from the view V_data_cluster: This view shows the number of  trips per hour per square /cluster.
This analysis don`t have satisfactory results, in next notebooks we will see that the reason is a too low number of data because it is divided in too many clusters.

## 03 - Mobility - Data 2 - Analysis with time series

With the bad result of the notebook before we have the idea of changing to de dataset2 because at this point we think that the Dataset 1 maybe is useless because it has a bad quality.

We try to use time series prediction but analyzing with pacf y acf we notice that there is not a good correlation so is not a good idea to continue on this side.

## 04 - Mobility - Data 2 -  BruteForce

With the bad result of the last notebook I want to check predicting with the 24 hours ago the exact value for the following hour (as a base line). The idea here is to have a baseline to compare future models.

The result obtained, checked with the heatmap, is a terrible prediction as bad as an aleatory method.

## 05 - Mobility - Data 2 -  Classification Model

First we design some diagrams to see the movement between clusters among the different hours, the idea behind this is to check visually if there is a pattern depending on the hour.

After build LogistricRegresion that predict if exist at least one trip or not. → The model predicts no trips to minimize the error because there is much more data with 0 data.

The predictions of these models (after balancing the data) are so bad.

## 06 - Mobility - Data 2 -  District

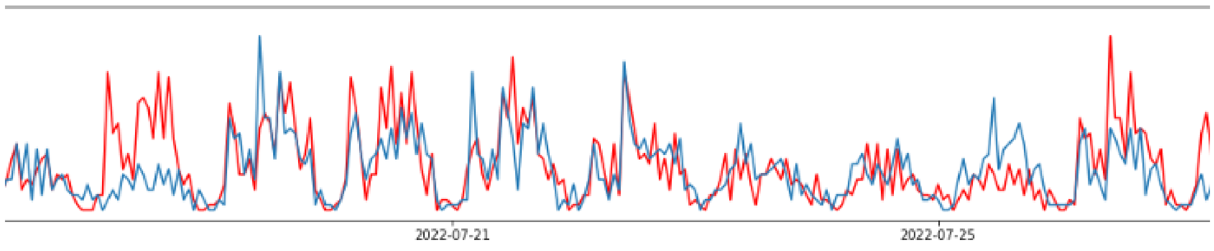
In this notebook we decide that the old clusters are not a good idea because the quantity of data is too low and predicting between 0 and 1 trips (with more 0 than 1 trips is so hard).

With this new idea we merge the trips dataset with districts.

After we remove the districts with a low number of trips. As we commented before, a reduced number of trips make bad predictions.

With this  new data in Acf and Pacf the capacity of prediction seems better.

We do it using only district 4 and after a brute force approximation (use yesterday data) to compare to the previous notebook and the prediction was much better:



After that, we follow a guide to do multiple time series forecasting but finally don't end it because as we will see in next notebooks one series predicts better than multiple series.

### 07 - Mobility - Data 2 -  District Time Series

In this notebook we try to train specific time series and after using Arima, as we will see later the models were not good because we don't have enough data in datasetB.

The result with the RandomForest is good.

### 08 - Mobility - Data 1 -  District Time Series

In this notebook we change the dataset to data1 because with data B we have half of the data A.

### 09 - Mobility - Data 1 -  District Time Series  - Only three

Same cells like last notebook but here only use the three providers that are availables now to have a good model to predict the current demand.

### 10 - Mobility - Data 1 -  District Time Series  - Only three - Last

Now we predict using time end trip instead of time start trip in order to show these predictions in real time from streamlit..
Also tried to estimate using ForecasterAutoreg but the result was not good.
After evaluating the model export to streamlit.

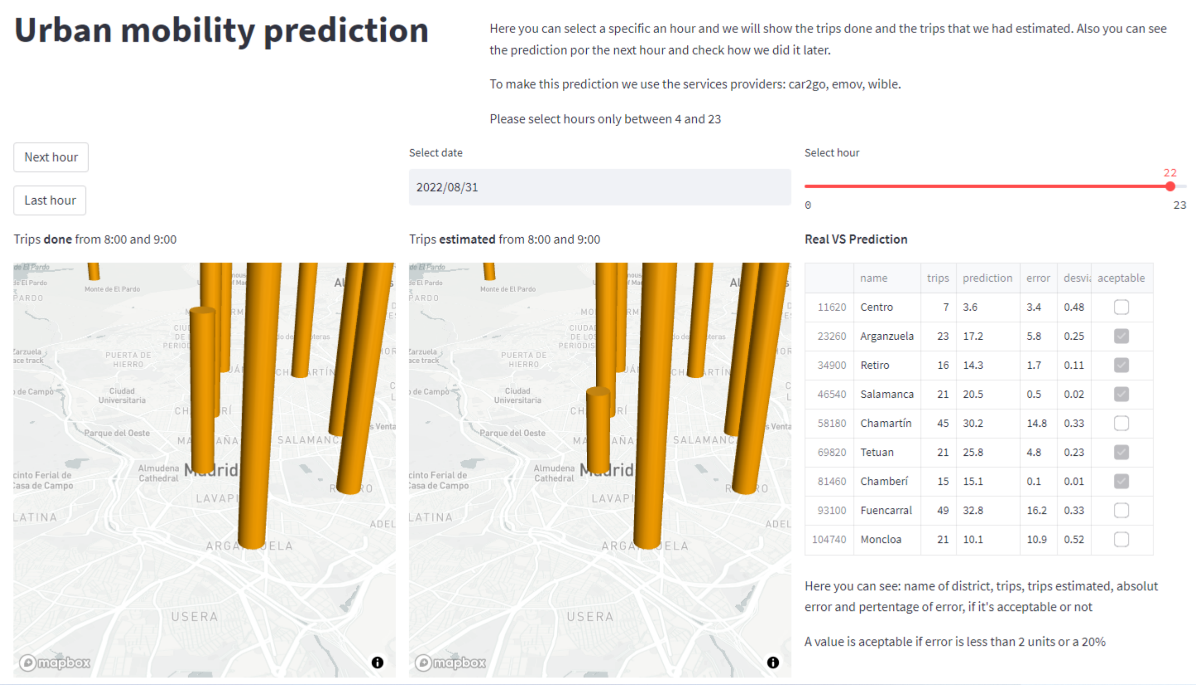
# 7. Visualization:

The objective of the visualization is to enable the user to view the capacity of the model developed.
The user can select an hour and and see the real data and the data that the model predicts.

The visualization can be accessed here. [ATTENTION: The website takes some second to be ready]

Is developed using streamlit and is deployed in Cloud Run (the reason for deploying in Cloud Run instead of Streamlit is that Streamlit has a limit of size and the model that we use weighs 7GB). The code is in github.

In the visualization you can specify a specific hour en the website will show you the real data and the data that the model estimate:



**Urban mobility prediction**

Here you can select a specific an hour and we will show the trips done and the trips that we had estimated. Also you can see the prediction por the next hour and check how we did it later.

To make this prediction we use the services providers: car2go, emov, wible.

Please select hours only between 4 and 23

**Real VS Prediction**

| | name | trips | prediction | error | desvi | aceptable |
|---|---|---|---|---|---|---|
| 11620 | Centro | 7 | 3.6 | 3.4 | 0.48 | ☐ |
| 23260 | Arganzuela | 23 | 17.2 | 5.8 | 0.25 | ✓ |
| 34900 | Retiro | 16 | 14.3 | 1.7 | 0.11 | ✓ |
| 46540 | Salamanca | 21 | 20.5 | 0.5 | 0.02 | ✓ |
| 58180 | Chamartín | 45 | 30.2 | 14.8 | 0.33 | ☐ |
| 69820 | Tetuan | 21 | 25.8 | 4.8 | 0.23 | ✓ |
| 81460 | Chamberí | 15 | 15.1 | 0.1 | 0.01 | ✓ |
| 93100 | Fuencarral | 49 | 32.8 | 16.2 | 0.33 | ☐ |
| 104740 | Moncloa | 21 | 10.1 | 10.9 | 0.52 | ☐ |

Here you can see: name of district, trips, trips estimated, absolut error and pertentage of error, if it's acceptable or not

A value is aceptable if error is less than 2 units or a 20%

ATTENTION: The button next hour show you data of real trips that are temporal because at 18:30 trip of 18:45 will not be ready

There are other tabs with more information:

- All providers: Is the same visualization but with all providers instead of only the three that continue providing data
- Visualize trips: Here you can see trips with the start and end point
- Districts by time: Small visualization of amount of trips, is useful to see top hours and top districts