

Tutorial Authors:

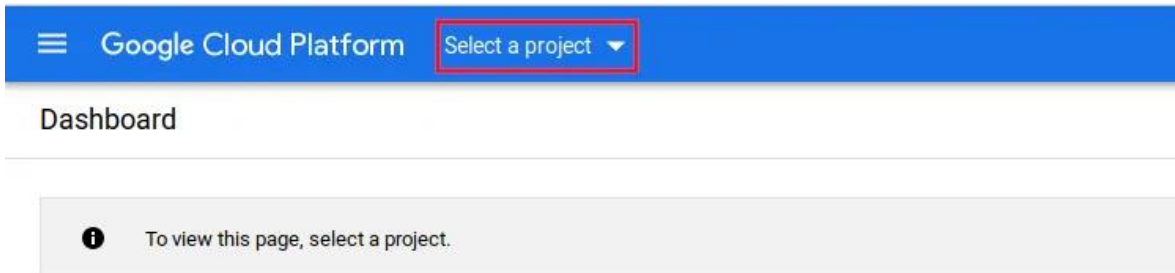
Hans Pech

Jorge Aque

Walter Vives

How to use Google Maps API

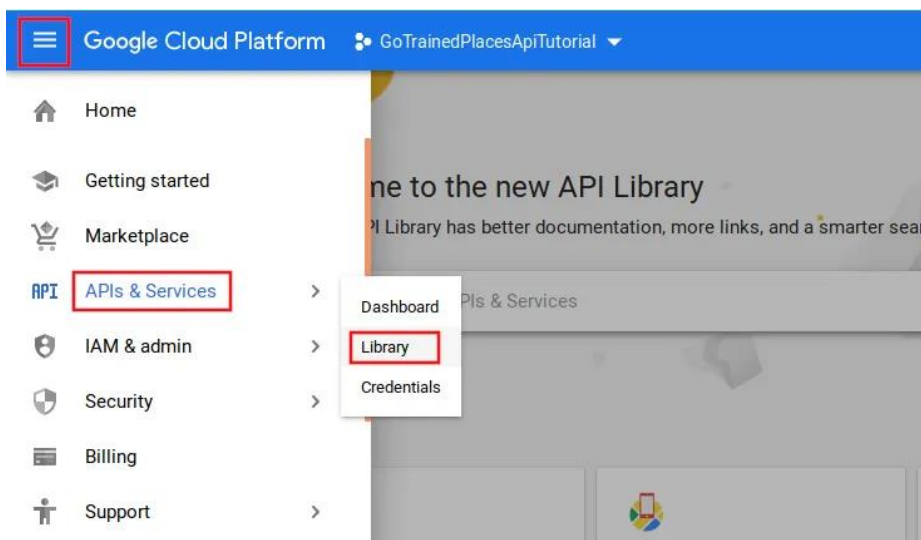
- First, we have to create an account in Google Cloud Console
- Once you are into the google cloud console from top navigation bar click “Select a project”



- In the new window click “New project” and create a new one



- From left side navigation go to “APIs & Services -> Library”



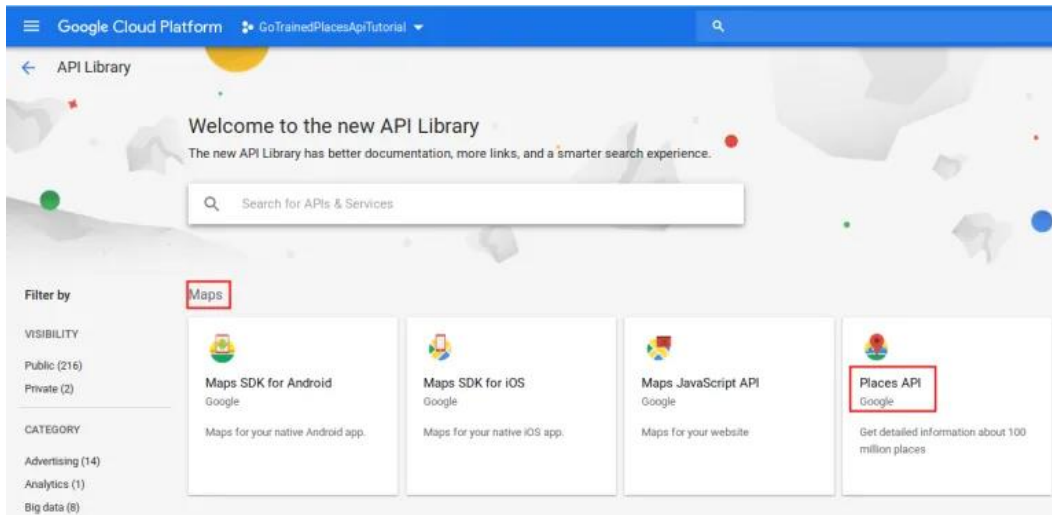
Tutorial Authors:

Hans Pech

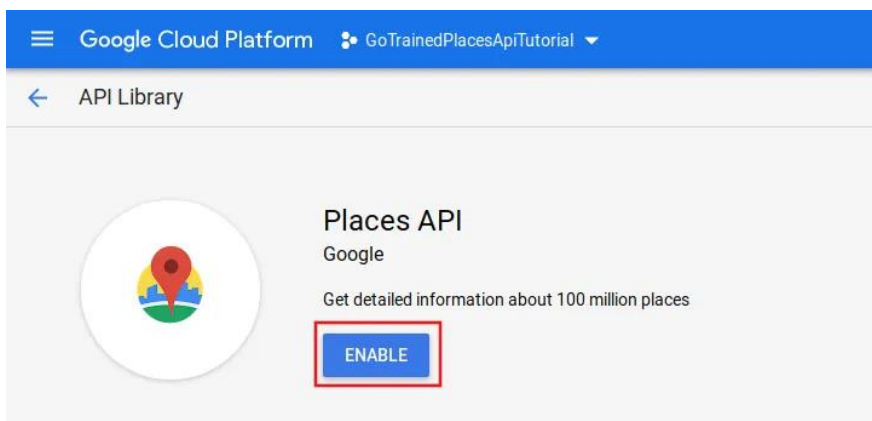
Jorge Aque

Walter Vives

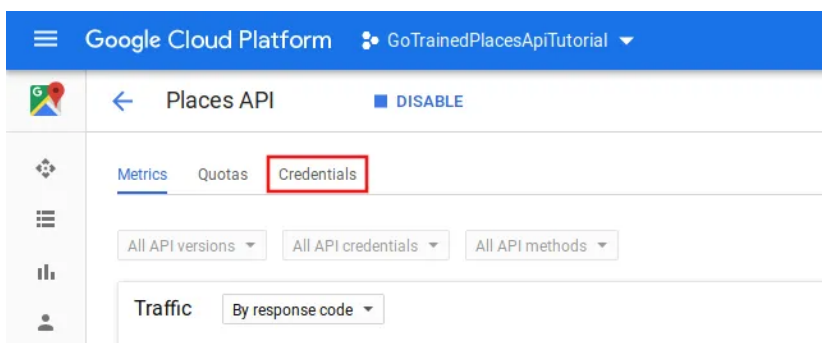
- From Maps section select “Places API” or search for “Places API” in the search box.



- Click on “Enable”



- Go to Credentials tab



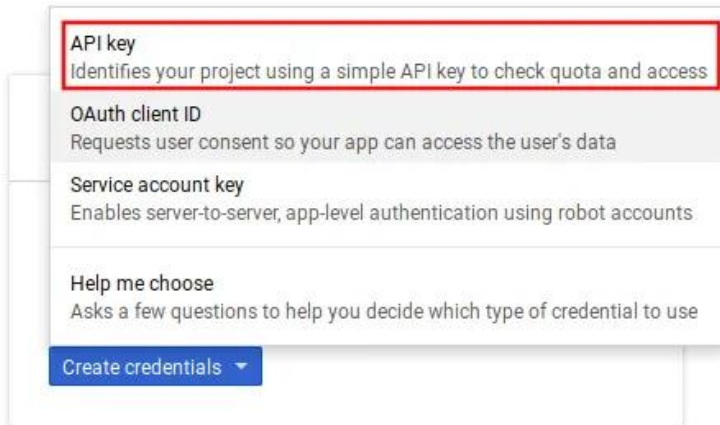
Tutorial Authors:

Hans Pech

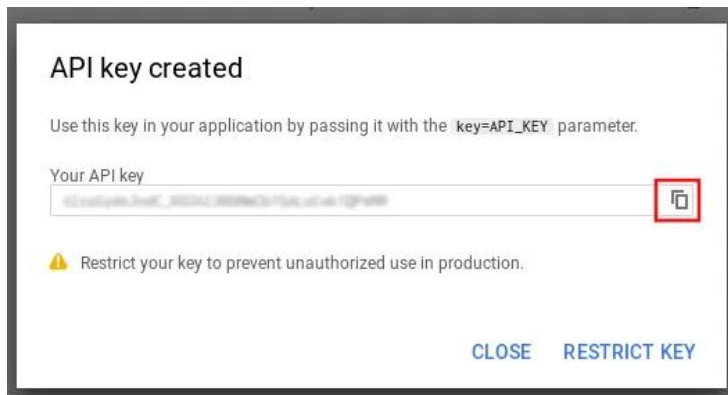
Jorge Aque

Walter Vives

- Click on Create credentials and then “API Key”



- So right now you’ve got your Google API key successfully. Now let’s get started with coding.



Nearby Search Method

The Places API allows you to query for place information on a variety of categories, such as: establishments, prominent points of interest, geographic locations, and more. You can search for places either by proximity or a text string. A Place Search returns a list of places along with summary information about each place; additional information is available via a Place Details query.

- Before start to code we need to import some libraries

Tutorial Authors:

Hans Pech

Jorge Aque

Walter Vives

```
#libraries that i will use
import requests
import json
import pandas as pd
```

- The code is divided by functions which return a response with the data from the places in a json format, also we have to use the url which is for nearbysearch that we can look for in the documentation of google (<https://developers.google.com/places/web-service/search>).

- This one is a function to find nearby restaurants to the given location in a radius in meters, we introduce some parameters which are: location, radius, key and the type which is restaurant,

```
#function to localize nearby places to the upy
def search_nearby_places(location, radius, key):
    url=('https://maps.googleapis.com/maps/api/place/nearbysearch/json'
        '?location=%s'
        '&radius=%s'
        '&type=restaurant'
        '&key=%s') % (location, radius, key)
    res = requests.get(url)
    results= json.loads(res.content)
    return results
```

- Once we code the function we need to proceed to declare our key, the location of the place and the radius in meters

```
#google key
key= 'AIzaSyAPDV0pYAop5UQ64HL_PWio9C-4vmoiUXQ'
```

```
#puttin coords and distance
places= search_nearby_places('20.988459, -89.736768', 2000 ,key)
```

- Once we have the results, we have to convert them into a data frame, and here I drop and rename some columns for the data frame aesthetic

Tutorial Authors:

Hans Pech

Jorge Aque

Walter Vives

```
#convert to df
df = pd.read_json(json.dumps(places['results']))
```

```
#dropping and renaming cols
dropCols = ['icon', 'geometry', 'photos', 'opening_hours', 'plus_code', 'reference', 'scope', 'vicinity', 'id']
nameCols = {'name':'Name', 'place_id':'Place Id', 'rating':'Rating', 'types':'Place Types', 'user_ratings_total':'Total of User Ratings'}
df = df.drop(columns = dropCols).rename(columns = nameCols)
df.head()
```

	Name	Place Id	Rating	Place Types	Total of User Ratings
0	La Glorieta (Cocina y Pizzería)	ChIJC9i7XEsNV08Ro2-sdS45rBg	4.1	[restaurant, food, point_of_interest, establis...	14.0
1	Mr. Cazuelas	ChIJ3WHHeJkMV08RO_z2pfoDFZg	4.0	[restaurant, food, point_of_interest, establis...	3.0
2	El Rincón Del Tabas	ChIJn4UFHpwMV08RZOjFHQCulwo	5.0	[restaurant, food, point_of_interest, establis...	2.0
3	Moyoyos burger	ChIJqxQ91h0NV08R52sc8a1HivE	NaN	[restaurant, food, point_of_interest, establis...	NaN
4	Cocina economica "El sabor de doña antonia"	ChIJJaQJm928NV08RAN8mnDmvYEK	NaN	[restaurant, food, point_of_interest, establis...	NaN

- The next is a function to find the id from a place when we input the name and location data, the url is changed to findplacefromtext and the parameters needed are name, location, key and the inputtype

```
#this function finds the id of a place when is giving the name and Loc
def findId(name, location, key):
    url="https://maps.googleapis.com/maps/api/place/findplacefromtext/json"
    parameters={
        'key':key,
        'input':name,
        'inputtype':'textquery',
        'locationbias':location
    }
    res=requests.get(url, params=parameters)
    result=json.loads(res.content.decode('utf-8'))
    return result['candidates'][0]['place_id']
```

```
#finding restaurant id
findId("McCarthy's Irish Pub - Cauce", "21.000164,-89.682577", key)
'ChIJ3yKKOKB0Vo8RCQcxkUw6hp8'
```

- The following function is to find the details from a place which includes review, rating, etc. Here we have to change in the url to details and put the language default in spanish or English, the parameters we need are the place id that we can get with the last function and our key

```
#this function will give the reviews and refer data of a restaurant with the id
def find_reviews(place_id, key):
    url="https://maps.googleapis.com/maps/api/place/details/json?place_id={}&language=en,es&key={}'.format(place_id, key)
    res = requests.get(url)
    results = json.loads(res.content)
    return results
```

Tutorial Authors:

Hans Pech

Jorge Aque

Walter Vives

-Once we get the data we have to turn the json response into a df for a better visualization, here is the code that shows the example, also if you want you can drop and rename some columns for a better visualization.

```
#converting the data into a df
place_data = find_reviews('ChIJJyKKOKB0Vo8RCQcxkUw6hp8',key)
df = pd.read_json(json.dumps(place_data['result']['reviews']),orient='records')

#dropping and renaming cols
dropCols = ['author_url', 'profile_photo_url', 'relative_time_description']
nameCols = {'author_name':'Name', 'language':'Language', 'rating':'Rating', 'text':'Text', 'time':'Time'}
df = df.drop(columns = dropCols).rename(columns = nameCols)
df.head()
```

	Name	Language	Rating	Text	Time
0	David Guillermo	en	3	They need person to regulate the sound inside ...	1566612760
1	Patrick	en	5	One of the best pubs around.\nGreat service, g...	1518688093
2	Islas	en	5	I love this place, one of the best places to d...	1522860496
3	Toni Aguilar	en	5	Rockers, music lovers in general will enjoy th...	1516852970
4	monica ravell	en	5	Great place to listen to good rock	1531876412

- To continue here there is a very similar function called place_detail and the difference here is that we add an extra parameter which is the price level, this one it will be returned into the json response and is the same process to convert it into a data frame

```
In [308]: #this function will give the place details when the id is giving
def place_details(place_id,key):
    url='https://maps.googleapis.com/maps/api/place/details/json?place_id={}&fields=type,price_level,rating&key={}'.format(place_id,key)
    res = requests.get(url)
    results = json.loads(res.content)
    return results
```

```
In [309]: #turning the data into a df
placedetails = place_details('ChIJRwKizrxVo8R1EULFLWZ2JQ', key)
df = pd.read_json(json.dumps(placedetails['result']),orient='records')
df
```

```
Out[309]:
```

	price_level	rating	types
0	2	4.5	restaurant
1	2	4.5	food
2	2	4.5	point_of_interest
3	2	4.5	establishment

- Applying the last function we can also look for similar places in base to the price level so in this new function we use the price level as a new parameters and the function will return the data from similar places.

Tutorial Authors:

Hans Pech

Jorge Aque

Walter Vives

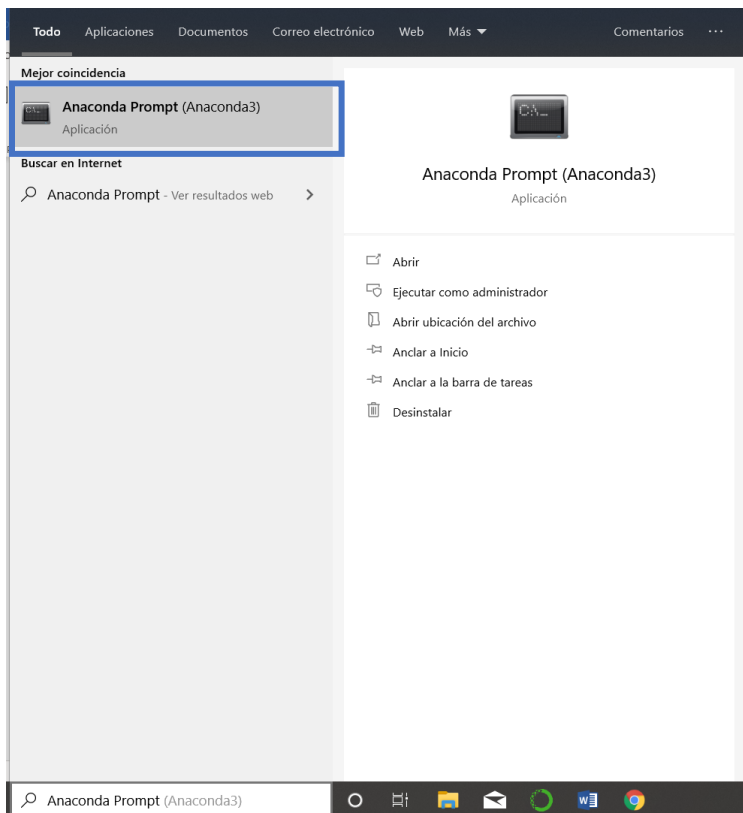
```
#this function gives us similar places data in base to the price level
def find_similar(price,key):
    url = 'https://maps.googleapis.com/maps/api/place/textsearch/json?query=restaurant&maxprice={}&type=food,point_of_interest_e
    res = requests.get(url)
    response = json.loads(res.content)
    return response
```

```
#converting the data into a df
price = df['price_level'][0]
similars = find_similar(price,key)
df = pd.read_json(json.dumps(similars['results']),orient='records')
```

```
#dropping and renaming cols
dropCols = ['formatted_address','geometry','icon','id','photos','reference','plus_code','opening_hours','price_level','types']
nameCols = {'name':'Place Name','place_id':'Place Id','rating':'Rating','user_ratings_total':'Total of User Ratings'}
df = df.drop(columns = dropCols).rename(columns = nameCols)
df.head()
```

	Place Name	Place Id	Rating	Total of User Ratings
0	Pizza Hut	ChIJm_cucXh0Vo8RqDAhDf6G6o	4.1	291
1	Alejandro's Pizzas	ChIJmx3FqBFxVo8RfQw8B4BXcml	4.6	241
2	Pancho's	ChIJ7WwShWfXVo8RZHeDYz1QD2M	4.4	737
3	La Chaya Maya	ChIJtzCcrGFxVo8RZLu9elke5cE	4.5	11236
4	Hot Wings Oriente	ChIJl2ywpH5xVo8RCc2c18iqI-8	4.1	566

- Now, We will work whit the Google's API but using the google maps library.
- First, we need to install the google maps library, so, in this case we write in the windows search "anaconda prompt".



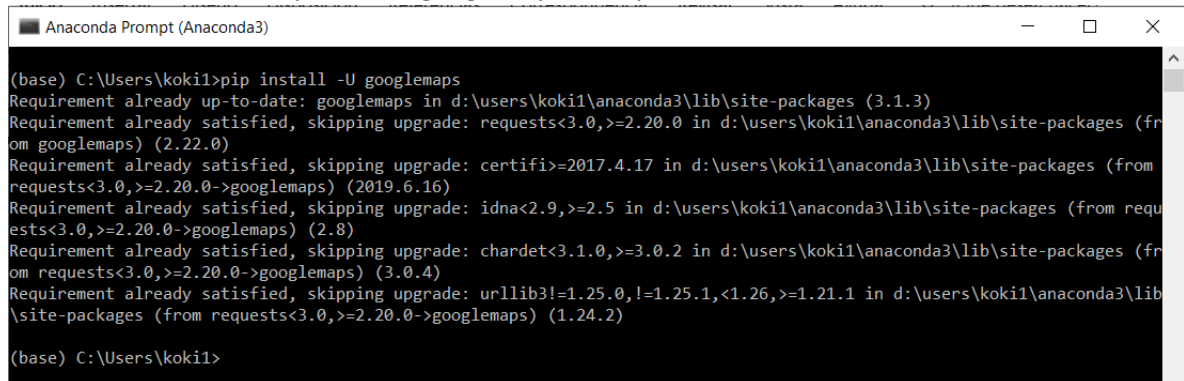
Tutorial Authors:

Hans Pech

Jorge Aque

Walter Vives

- Then we click it. Once inside we need to write “pip install -U googlemaps” and press enter. And we will have already installed googlemaps library



```
Anaconda Prompt (Anaconda3)

(base) C:\Users\koki1>pip install -U googlemaps
Requirement already up-to-date: googlemaps in d:\users\koki1\anaconda3\lib\site-packages (3.1.3)
Requirement already satisfied, skipping upgrade: requests<3.0,>=2.20.0 in d:\users\koki1\anaconda3\lib\site-packages (from googlemaps) (2.22.0)
Requirement already satisfied, skipping upgrade: certifi>=2017.4.17 in d:\users\koki1\anaconda3\lib\site-packages (from requests<3.0,>=2.20.0->googlemaps) (2019.6.16)
Requirement already satisfied, skipping upgrade: idna<2.9,>=2.5 in d:\users\koki1\anaconda3\lib\site-packages (from requests<3.0,>=2.20.0->googlemaps) (2.8)
Requirement already satisfied, skipping upgrade: chardet<3.1.0,>=3.0.2 in d:\users\koki1\anaconda3\lib\site-packages (from requests<3.0,>=2.20.0->googlemaps) (3.0.4)
Requirement already satisfied, skipping upgrade: urllib3!=1.25.0,!1.25.1,<1.26,>=1.21.1 in d:\users\koki1\anaconda3\lib\site-packages (from requests<3.0,>=2.20.0->googlemaps) (1.24.2)

(base) C:\Users\koki1>
```

- Those are the libraries that we need to import, this is the way to use them.

```
import googlemaps as gm
from datetime import datetime
```

- Here is you will put your key, you will use it (“your key “ = ****)

```
KEY = gm.Client(key = ****)
```

- So, first we create the function and declare the parameter as the google’s documentation

```
def search_nearby_google(location, radius, keywords):

    sem = KEY.places_nearby(location=(location), radius=radius, keyword=keywords)
    return sem
```

- Here we will create a variable and we will use the function, we need to put the parameters in the same position

```
sem = search_nearby_google(location, 3000, 'restaurant')
```

- Once done we create a Data Frame to show the results, and here we deleted the columns that we won’t use with .drop()

Tutorial Authors:

Hans Pech

Jorge Aque

Walter Vives

```
df = pd.read_json(json.dumps(sem['results']),orient='records')
cols = ['geometry','icon','id','opening_hours','photos','plus_code','reference','scope','vicinity']
df = df.drop(columns=cols)
cols = {'name':'Name','place_id':'Place ID','rating':'Rating','types':'Types','user_ratings_total':'Total of ratings'}
df = df.rename(columns=cols)
df
```

	Name	Place ID	Rating	Types	Total of ratings
0	Antojitos El Cielo	ChIJKSZhQVwLV08RiNqp6upowD4	4.5	[restaurant, food, point_of_interest, establis...	26
1	Restaurant El Parguito De Celestún	ChIJi6qRzU4NV08RCHrvRJcvtYU	0.0	[restaurant, food, point_of_interest, establis...	0
2	LA PLAYA	ChIJOUmGIFyLV08RqgAl-tRdUmg	4.8	[restaurant, food, point_of_interest, establis...	21

- This are the function using the parameters. Something important is that we always use the API key as you can appreciated.

```
def find_place_google(input1, inputtype,location):
    sem = KEY.find_place(input=input1,input_type=inputtype,location_bias=location)
    return sem
```

- We Will use the find_place_google function to find those places, so we can declare one variable for each place. We need to put the parameters to find the places correctly.

```
sem1 = find_place_google('restaurant','textquery','point:20.9999339,-89.6847603')
place_id1 = sem1['candidates'][0]['place_id']

sem2 = find_place_google('restaurant','textquery','point:20.9845009,-89.6189244')
place_id2 = sem2['candidates'][0]['place_id']

sem3 = find_place_google('restaurant','textquery','point:21.030424, -89.641694')
place_id3 = sem3['candidates'][0]['place_id']
```

- And this is the form that we can do the output. As we can see, we print the place's name and the Place ID

Tutorial Authors:

Hans Pech

Jorge Aque

Walter Vives

```
"McCarty's - Cauce1:",place_id_Mc
```

```
("McCarty's - Cauce1:", 'ChIJOWd5R6B0Vo8RmWD-WJPMpEE')
```

```
"Starbucks Montejo:",place_id_Sb
```

```
('Starbucks Montejo:', 'ChIJ_1FNC1pxVo8RUYX6mmUfwuk')
```

```
"Los Trompos:",place_id_Lt
```

```
('Los Trompos:', 'ChIJgbGOKq12Vo8RqhK2SdKJzdQ')
```

- Here we use a Place ID as an example to do methon Place Detail, Here we deleted the columns that we will no use and make use of the library datetime to change the time format.

```
sem = KEY.place(place_id='ChIJJyKKOKB0Vo8RCQcxkUw6hp8')
```

```
datf = pd.read_json(json.dumps(sem['result']['reviews']),orient='records')
cols = ['author_url','profile_photo_url','relative_time_description']
datf = datf.drop(columns=cols)
cols = {'author_name':'Author','language':'Language','rating':'Rating','text':'Comment','time':'Date'}
datf = datf.rename(columns=cols)
datf['Date'] = pd.to_datetime(datf['Date'],unit='s')
```

- This is the outputs form, as we can see, the time format is easier to read

```
datf
```

	Author	Language	Rating	Comment	Date
0	David Guillermo	en	3	They need person to regulate the sound inside ...	2019-08-24 02:12:40
1	Patrick	en	5	One of the best pubs around.\nGreat service, g...	2018-02-15 09:48:13
2	Islas	en	5	I love this place, one of the best places to d...	2018-04-04 16:48:16
3	Toni Aguilar	en	5	Rockers, music lovers in general will enjoy th...	2018-01-25 04:02:50
4	monica ravell	en	5	Great place to listen to good rock	2018-07-18 01:13:32

Tutorial Authors:

Hans Pech

Jorge Aque

Walter Vives

- Here we need to create the function and declare the parameter, in this case the place Id. We need to use Los Trompos ID. We fill the parameter and then we create a data frame to watch the results.

```
def find_similar_google(place_id,):  
    sem = KEY.place(place_id=place_id)  
    return sem
```

```
sem = find_similar_google('ChIJKa_Q_zh0Vo8R5HNk0URi_zE')  
price = sem['result']['price_level']
```

```
df = pd.read_json(json.dumps(sem['result']['types']),orient='records')  
df
```

	0
0	restaurant
1	food
2	point_of_interest
3	establishment

- The second point is to create a function that return similar place, we need to declare the parameter to return the correct results that we want.

```
def find_similar_google_Det(Cus,maxim,tipe):  
|  
    sem = google.places(query=Cus,max_price=maxim,type=tipe)  
    return sem
```

```
sem = find_similar_google_Det('restaurant',price,'restaurant,food,point_of_interest,establishment')
```

- Once done we create a data frame to watch the results of the coincidences. This is the way that we can use the google maps library.

```
dataFrame = pd.read_json(json.dumps(sem['results']),orient='records')  
cols = ['formatted_address','geometry','icon','id','opening_hours','photos','place_id','plus_code','reference']  
dataFrame = dataFrame.drop(columns=cols)
```

```
dataFrame
```

Tutorial Authors:

Hans Pech

Jorge Aque

Walter Vives

	name	price_level	rating	types	user_ratings_total
0	Colibri	2	4.4	[restaurant, food, point_of_interest, establis...	524
1	Garage Restaurant & Bar	2	4.2	[bar, restaurant, food, point_of_interest, est...	636
2	Del Jardin	2	4.1	[restaurant, food, point_of_interest, establis...	251
3	Zandunga	2	4.5	[restaurant, food, point_of_interest, establis...	884
4	Los Chavales de la Barriada	2	4.4	[restaurant, food, point_of_interest, establis...	1115
5	El Fogoncito	2	4.2	[restaurant, food, point_of_interest, establis...	424
6	Restaurante Catedral	2	4.6	[restaurant, food, point_of_interest, establis...	1352
7	Río de Janeiro (Buffet de Espadas-Parrilladas-...	2	4.0	[restaurant, food, point_of_interest, establis...	229
8	Terranova	2	4.1	[restaurant, food, point_of_interest, establis...	180
9	La Biznaga	2	4.2	[restaurant, food, point_of_interest, establis...	757
10	El Biche Pobre 1	2	4.0	[restaurant, food, point_of_interest, establis...	106
11	La Rustica	2	4.2	[restaurant, food, point_of_interest, establis...	314
12	El Morocco	2	4.5	[restaurant, food, point_of_interest, establis...	125
13	El Escapulario	2	4.3	[restaurant, food, point_of_interest, establis...	228
14	Tr3s 3istro Restaurant & Oyster Bar	2	4.3	[restaurant, food, point_of_interest, establis...	516
15	Pizza Leggera Ristorante	2	4.4	[restaurant, food, point_of_interest, establis...	87
16	La Teca	2	4.3	[restaurant, food, point_of_interest, establis...	206
17	Hong Kong	2	4.3	[restaurant, food, point_of_interest, establis...	286
18	Gozobi	2	4.2	[cafe, bar, restaurant, food, point_of_interes...	821
19	Burger Grill	2	4.4	[restaurant, food, point_of_interest, establis...	553