

## Inteligencia Artificial

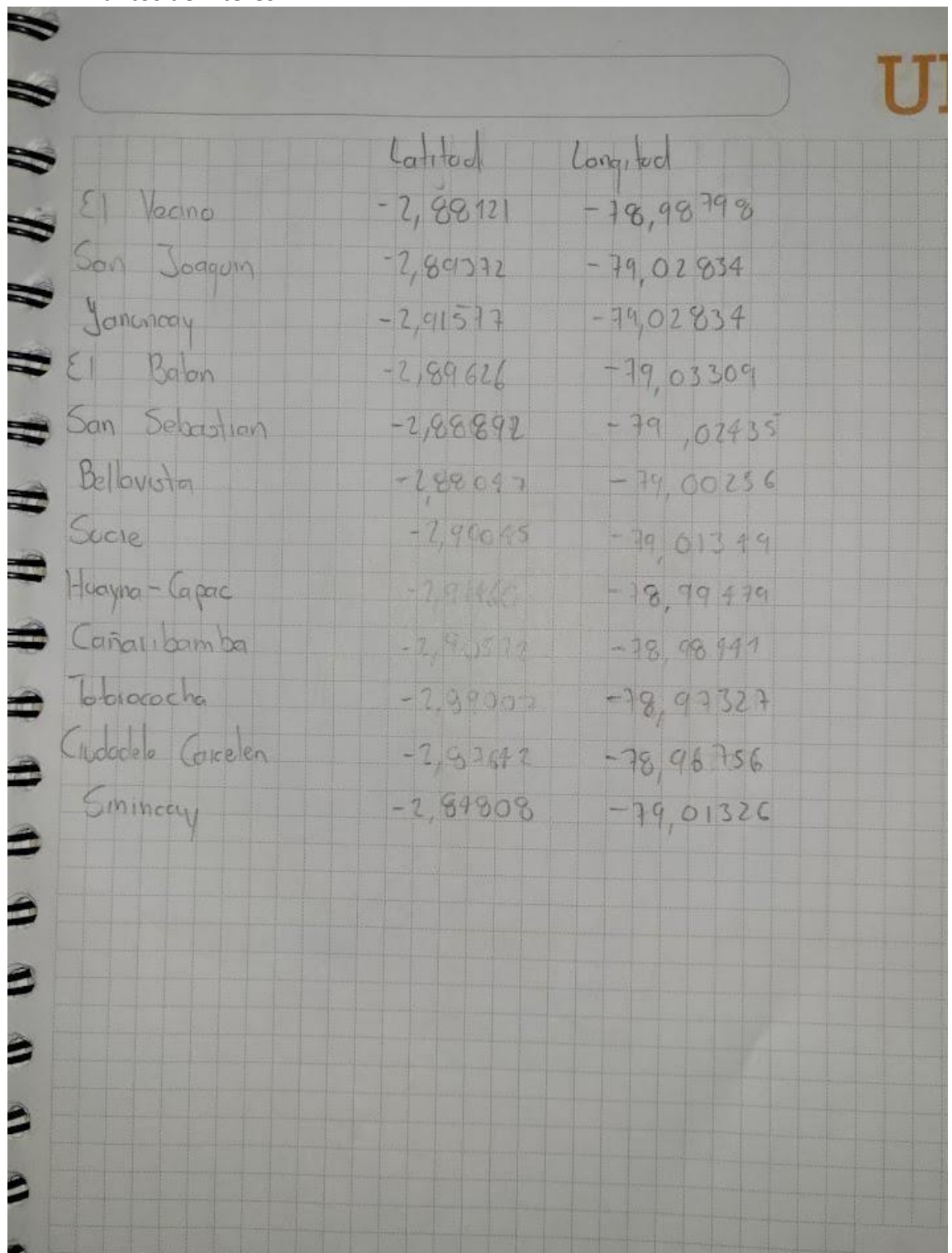
### Busque da Por Anchura

- Diseñe un grafo similar al que se ha presentado en este ejercicio partiendo de las siguientes coordenadas de latitud y longitud: -2.8801604,-79.0071712. Para ello deberá realizar las siguientes tareas:

Emplear la herramienta Google Maps (R) con las coordenadas antes indicadas.

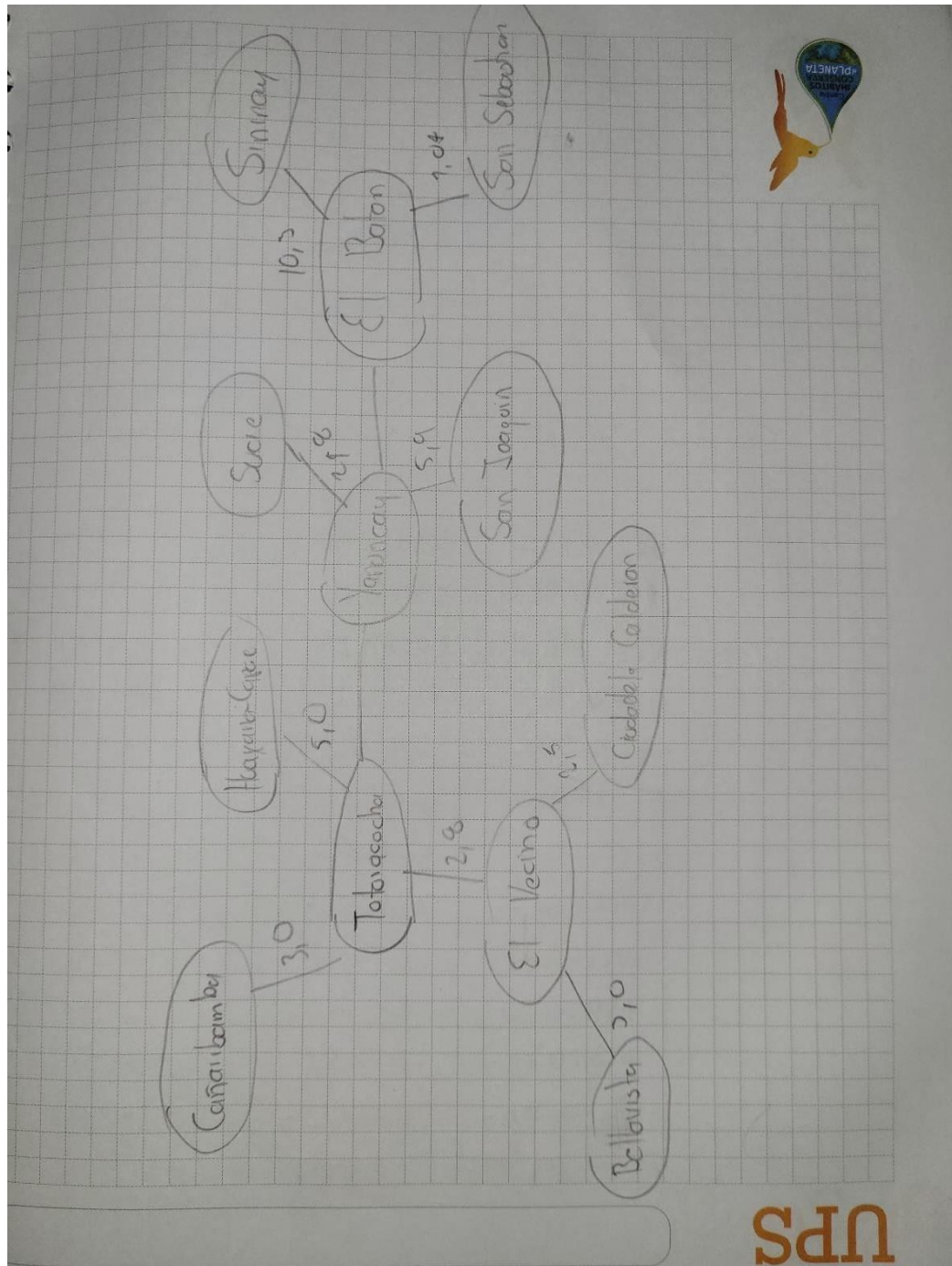
Definir 11 puntos de interés y armar el grafo.

11 Puntos de interés



	Latitud	Longitud
El Vecino	-2,88121	-78,98798
San Joaquin	-2,89572	-79,02834
Yanuncay	-2,91577	-79,02834
El Balon	-2,89626	-79,03309
San Sebastian	-2,88892	-79,02435
Bellavista	-2,88047	-79,00256
Sucie	-2,90045	-79,01349
Huayna-Capac	-2,91460	-78,99479
Cañalibamba	-2,89972	-78,98991
Tobiarocha	-2,89002	-78,97327
Ciudadela Carelen	-2,87642	-78,96756
Smincay	-2,84808	-79,01320

Grafo



Especificar como punto de partida al sector "San Sebastián" y como objetivo "Totoracocho".

- 1) Punto de partida: San Sebastián"
- 2) Punto objetivo: Totoracocho

Realizar el proceso de búsqueda de forma similar a cómo se ha explicado en este apartado, almacenando para ello los datos de la lista Visitados y de la Cola.



UPS

1) Lista Nodos { El Balon }  
Visitados { San Sebastian }

2) Lista Nodos { Varanagar, Siminagar }  
Visitados { San Sebastian, El Balon }

3) Lista Nodos { Siminagar, Totaraochoa, Surco, San Joaquin }  
Visitados { San Sebastian, El Balon, Varanagar }

Solucion { San Sebastian, El Balon, Varanagar, Totaraochoa }



# Importar la API py2neo

Para el ingreso de los datos que se encuentran dentro de la lista

## Conexión con Neo4j

Configure la URL de conexión con la base de datos de Neo4j:

Creación de los 11 lugares con sus relaciones.

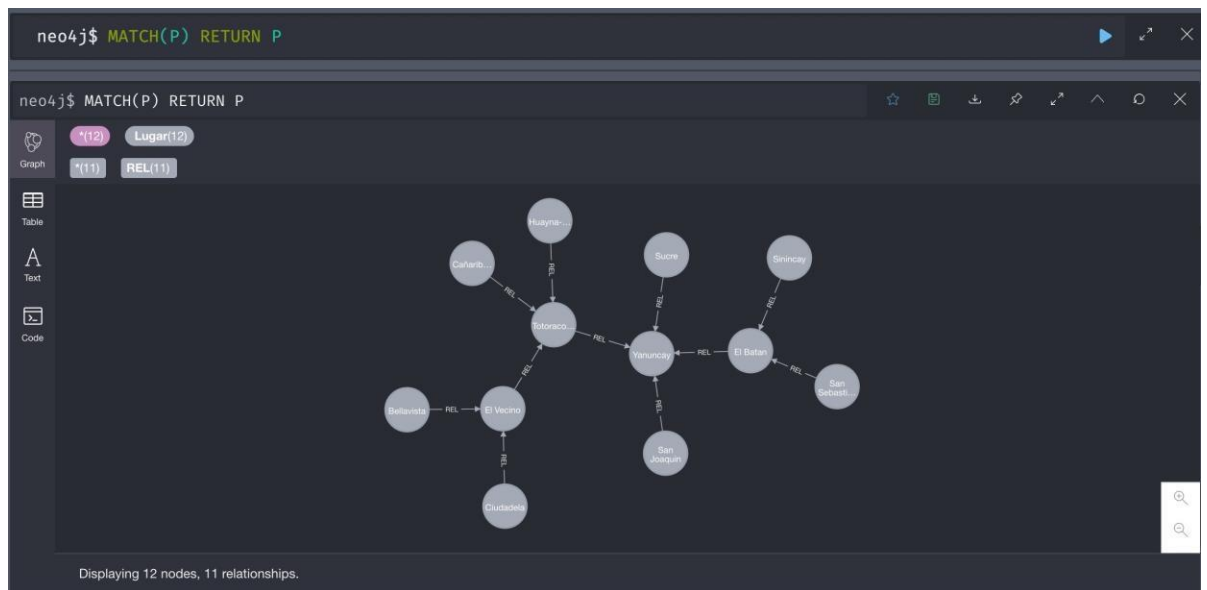
In [1]: 1 *#IMPORTAR py2neo*

```
2 from py2neo import Node, Relationship, Graph
3
4
5 # connect to authenticated graph database
6 graph = Graph("bolt://localhost:7687", aut="neo4j", password="cuenca")
```

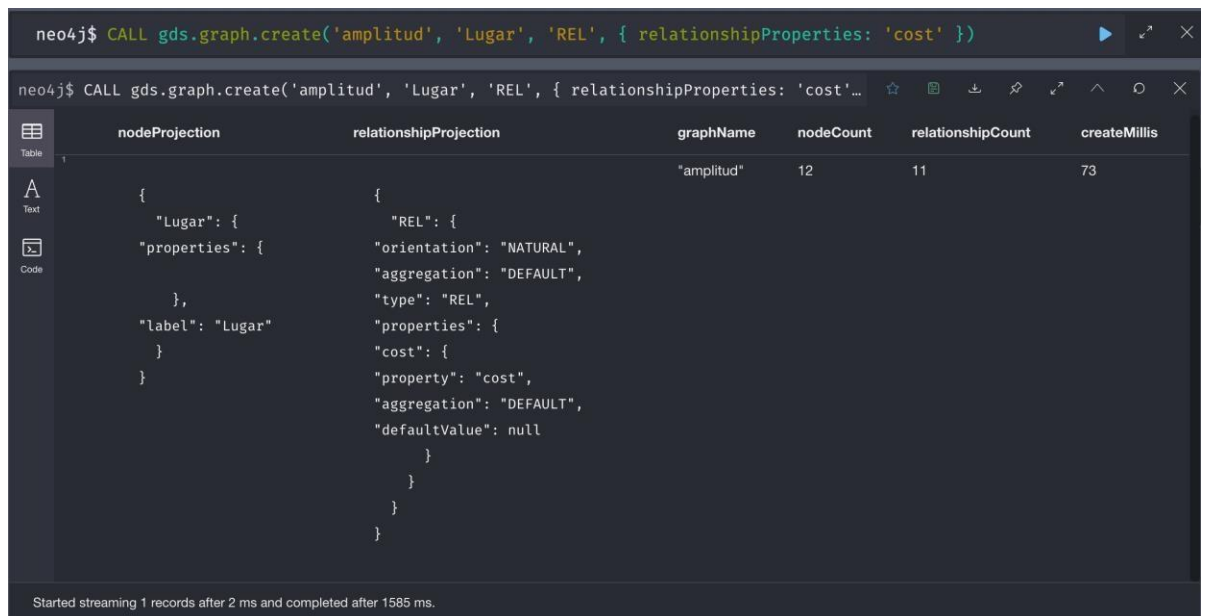
```
In [5]: 1 graph.run(" CREATE (a:Lugar {name: 'El Vecino', latitude: -2.8812, longitude: -2.89372, longi
2                                     '(b:Lugar {name: 'San Joaquin',
3                                     latitude: -2.89372, longi
4                                     '(c:Lugar {name: 'Yanuncay', latitude: -2.91577, longitud
5                                     '(d:Lugar {name: 'El Batan',latitude: -2.89626, longitude
6                                     '(e:Lugar {name: 'San Sebastian',latitude: -2.88892, long
7                                     '(f:Lugar {name: 'Bellavista',latitude: -2.88047, longitu
8                                     '(g:Lugar {name: 'Sucre',latitude: -2.90045, longitude: -
9                                     '(h:Lugar {name: 'Huayna-Capac',latitude: -2.91460, longi
10                                    '(i:Lugar {name: 'Cañaribamba',latitude: -2.90512, longit
11                                    '(j:Lugar {name: 'Totoracocha',latitude: -2.89002, longit
12                                    '(k:Lugar {name: 'Ciudadela Calderon',latitude: -2.87642,
13                                    '(m:Lugar {name: 'Sinincay',latitude: -2.84808, longitude
14                                    '(e)-[:REL {cost: 1.04}]->(d)," +
15                                    '(m)-[:REL {cost: 10.3}]->(d)," +
16                                    '(d)-[:REL {cost: 4.2}]->(c)," +
17                                    '(b)-[:REL {cost: 5.9}]->(c)," +
18                                    '(g)-[:REL {cost: 2.8}]->(c)," +
19                                    '(j)-[:REL {cost: 10.8}]->(c)," +
20                                    '(h)-[:REL {cost: 5.0}]->(j)," +
21                                    '(i)-[:REL {cost: 3.0}]->(j)," +
22                                    '(a)-[:REL {cost: 2.8}]->(j)," +
23                                    '(f)-[:REL {cost: 3.0}]->(a)," +
24                                    '(k)-[:REL {cost: 2.5}]->(a) ").data()
25
26
27
```

Out[5]: []

Consultar la creación correcta de los nodos:



Crear el gráfico el cual almacenará un catálogo de gráficos .



Lo siguiente ejecutará el algoritmo y transmitirá los resultados:



```

1 MATCH (a:Lugar{name:'San Sebastian'}), (d:Lugar{name:'Totoracocha'})
2 WITH id(a) AS startNode, [id(d)] AS targetNodes
3 CALL gds.alpha.bfs.stream('amplitud', {startNode: startNode, targetNodes: targetNodes})
4 YIELD path
5 UNWIND [ n in nodes(path) | n.name ] AS Nombre
6 RETURN Nombre

```

neo4j\$ MATCH (a:Lugar{name:'San Sebastian'}), (d:Lugar{name:'Totoracocha'}) WITH id(a) AS...

	Nombre
1	"San Sebastian"
2	"El Batan"
3	"Yanuncay"

Started streaming 3 records after 1 ms and completed after 2 ms.

```

In [8]: 1 graph.run("MATCH (a:Lugar{name:'San Sebastian'}), (d:Lugar{name:
2         " WITH id(a) AS startNode, [id(d)] AS targetNodes"+
3         " CALL gds.alpha.bfs.stream('amplitud', {startNode: st
4         " YIELD path"+
5         " UNWIND [ n in nodes(path) | n.name ] AS Nombre"+
6         " RETURN Nombre").data()

```

Out[8]: [{'Nombre': 'San Sebastian'}, {'Nombre': 'El Batan'}, {'Nombre': 'Yanuncay'}]

In []:  1