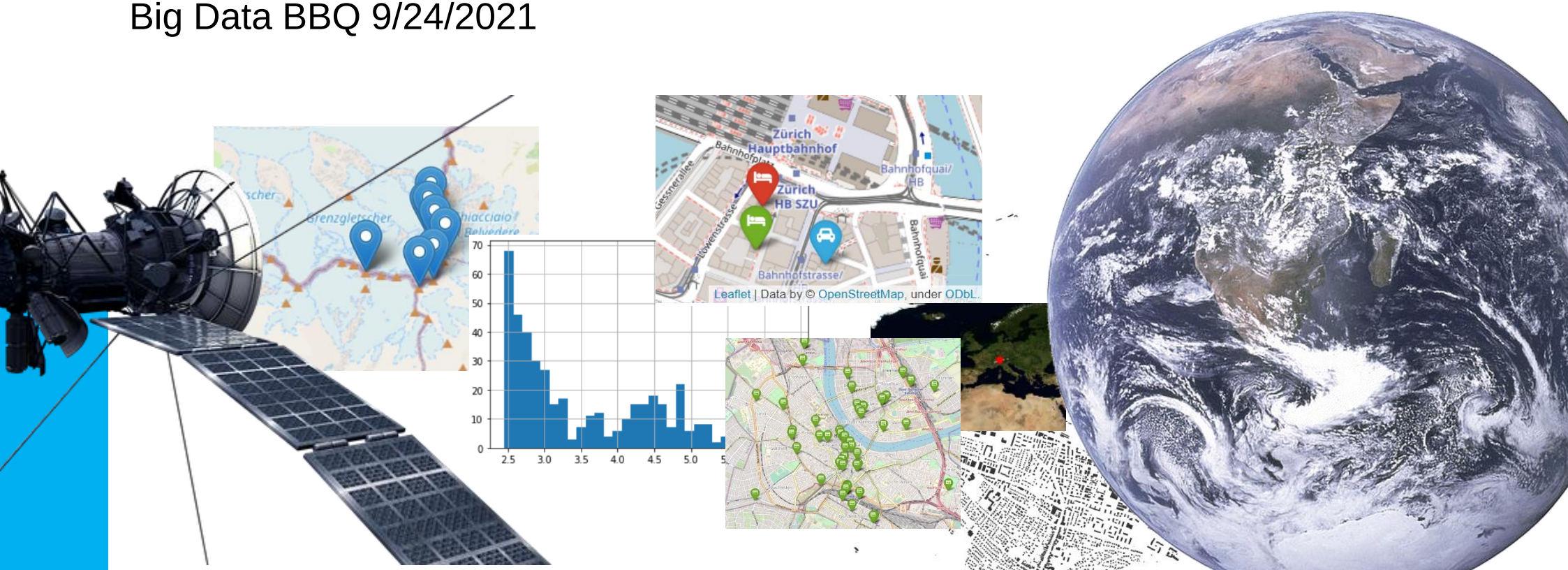
University of Applied Sciences and Arts Northwestern Switzerland
School of Architecture, Civil Engineering and Geomatics

Geospatial Analysis for the Smart City

Big Data BBQ 9/24/2021



Prof. Martin Christen
FHNW – University of Applied Sciences and Arts Northwestern Switzerland
School of Architecture, Civil Engineering and Geomatics
Institute Geomatics
martin.christen@fhnw.ch

@MartinChristen

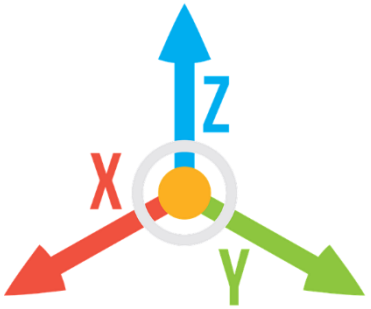
What is Geospatial Data ?

Geospatial data is data with a spatial component - it describes objects with a spatial reference to the earth's surface.

This data consists of a spatial component (**where**), attributes (**what**), and often a time reference (**when**).

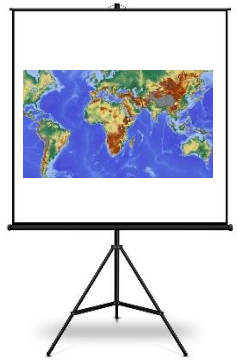
What is geospatial data?

Geospatial data is **multidimensional**, we need at least two coordinates to define a position (for example x and y).



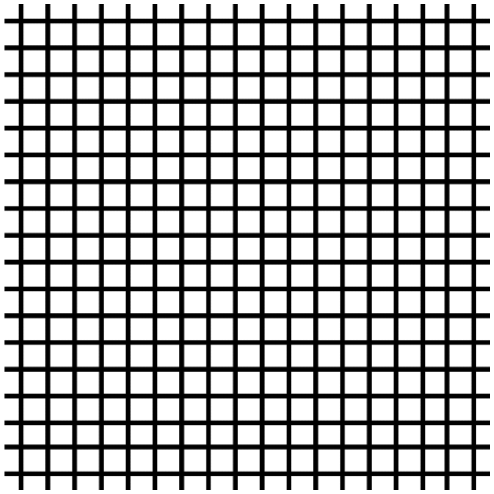
Geospatial data can be huge - some datasets may be in the terabyte to petabyte range, which also makes it really hard to display it or to update it.

And when we want to view the data it is usually projected to a flat surface.



Geospatial Data Representation

There are two main methods of representing geospatial data digitally. Both methods try to reduce the reality of something we can store on the computer, the first kind is called **raster data** and the second **vector data**.



raster data

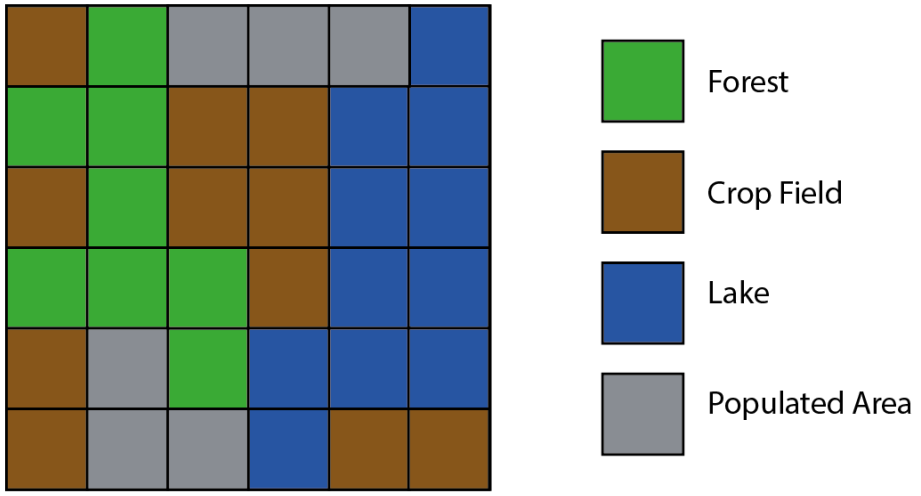
vector data

Raster Representation

In a raster representation, space is divided into square cells. Each cell then receives properties or attributes to this cells.

Often this raster data comes from remote-sensing satellites.

Working with this gridded representation is often quite easy.

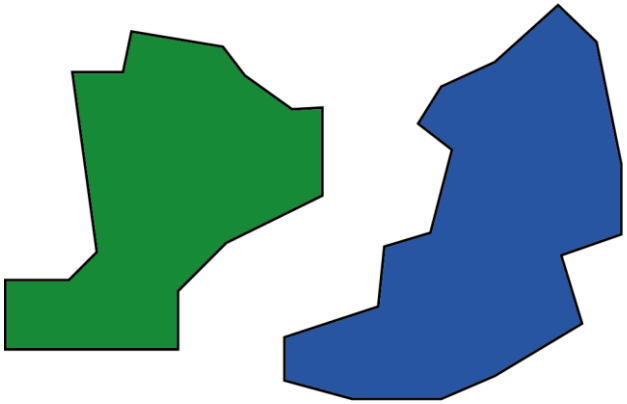


Formats:

.jpg
.png
.tiff (Geotiff)

Vector Representation

In a vector representation, we would have points captured around the lake and connect them. For the lake in the previous raster example, we would have a polygon instead of grids. Vector data contains points, lines, polygons and more.



Formats:

Shapefile
GeoJSON
GeoPackage
...

Where to get data ?

Nowadays there are many global and local open (and closed) data portals, geospatial data is present everywhere.

https://index.okfn.org/dataset/

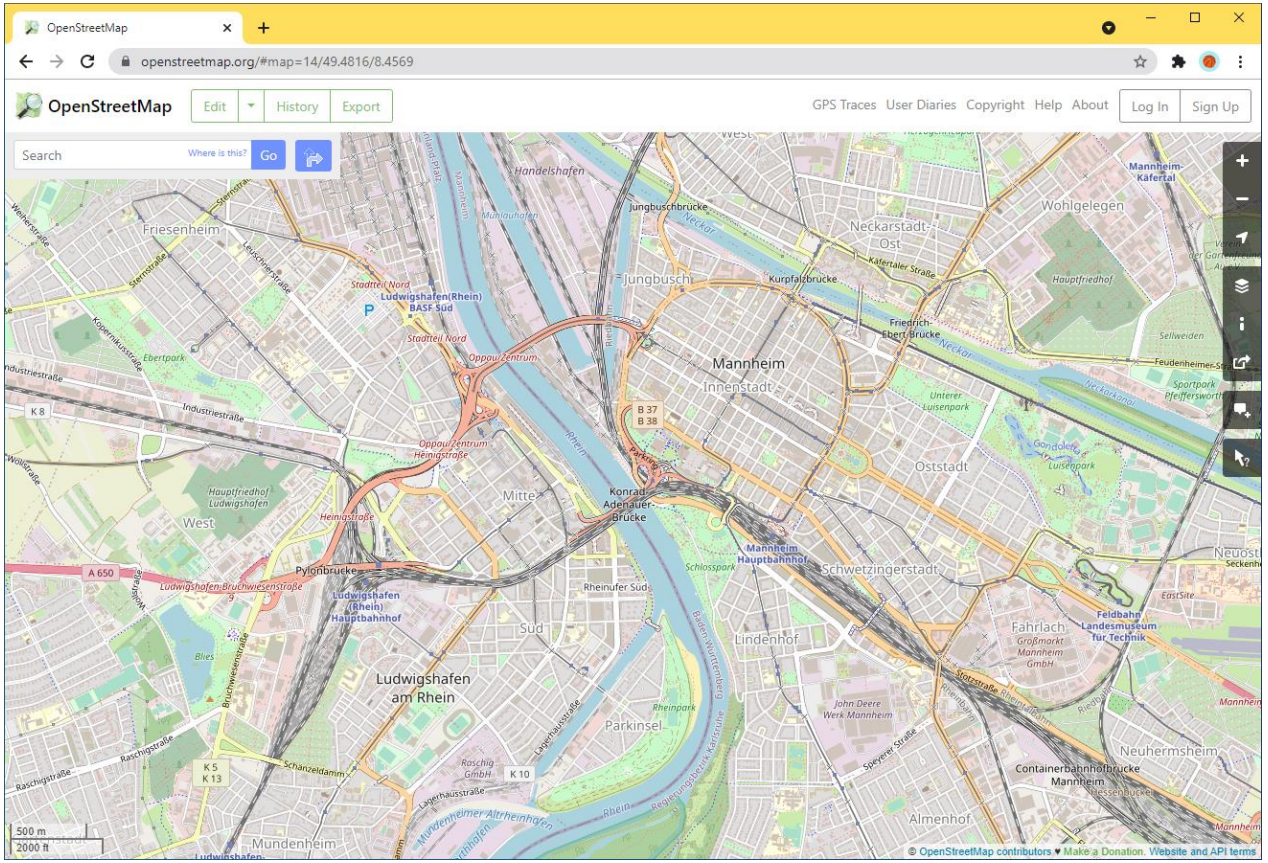
https://data.world/datasets/open-data

https://www.gis-mannheim.de/ (GeoPortal der Stadt Mannheim)

https://www.swisstopo.admin.ch/de/geodata.html (Switzerland)

OpenStreetMap

OpenStreetMap (OSM) is a collaborative project to create a free editable geographic database of the world.



OpenStreetMap & Python

OSMnx Python for Street networks

A Python module to retrieve, model, analyze and visualize street networks and other geospatial data from OpenStreetMap

https://github.com/gboeing/osmnx

https://osmnx.readthedocs.io/en/stable/

Installation:

```
conda install osmnx folium -c conda-forge
```

Initialize – Always use caching

```
import folium
import osmnx as ox
import networkx as nx
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
ox.config(use_cache=True, log_console=False)
```

Retrieve Street Network from OSM Data

- drive - get drivable public streets (but not service roads)
- drive_service - get drivable streets, including service roads
- walk - get all streets and paths that pedestrians can use (this network type ignores one-way directionality)
- bike - get all streets and paths that cyclists can use
- all - download all non-private OSM streets and paths
- all_private - download all OSM streets and paths, including private-access ones

```
place = "Mannheim, Germany"
graph = ox.graph_from_place(place, network_type='drive')
```

```
fig, ax = ox.plot_graph(graph)
```

```
fig, ax = ox.plot_graph(graph)
```

Export and retrieve some info

```
ox.save_graph_shapefile(graph, filepath='network-shape_mannheim')
```

```
nodes, streets = ox.graph_to_gdfs(graph)
```

```
len(streets)
```

```
14441
```

GeoPandas DataFrame

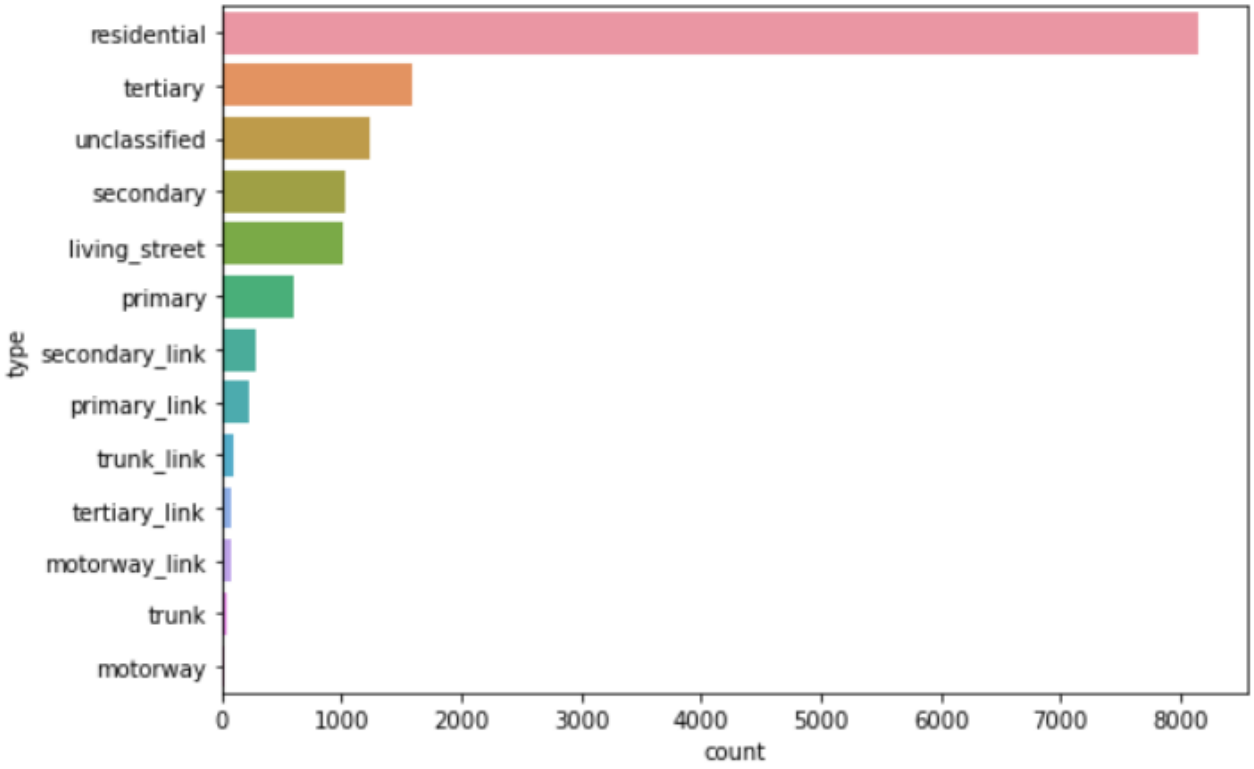
```
streets.head()
```

	osmid	oneway	lanes	ref	highway	maxspeed	width	length	geometry	bridge	name	access	tunnel	junction	u	v	key
0	143471353	True	2	A 6	motorway_link	none	7.5	263.893	LINESTRING (8.55986 49.52003, 8.55989 49.51939...	NaN	NaN	NaN	NaN	NaN	488245	502473	0
1	[186651648, 186651649]	True	[2, 3]	A 6	motorway	none	NaN	379.228	LINESTRING (8.44393 49.55334, 8.44128 49.55327...	NaN	NaN	NaN	NaN	NaN	500071	500072	0
2	[265305354, 183467651, 192683715]	True	[5, 4]	A 6	motorway	none	[15, 18.5]	863.368	LINESTRING (8.56011 49.51767, 8.56015 49.51681...	NaN	NaN	NaN	NaN	NaN	502473	411034960	0

Street Types in Mannheim / OSM

```
street_types
```

	type	count
0	residential	8147
1	tertiary	1590
2	unclassified	1238
3	secondary	1023
4	living_street	1007
5	primary	595
6	secondary_link	286
7	primary_link	230
8	trunk_link	95
9	tertiary_link	86
10	motorway_link	77
11	trunk	44
12	motorway	23



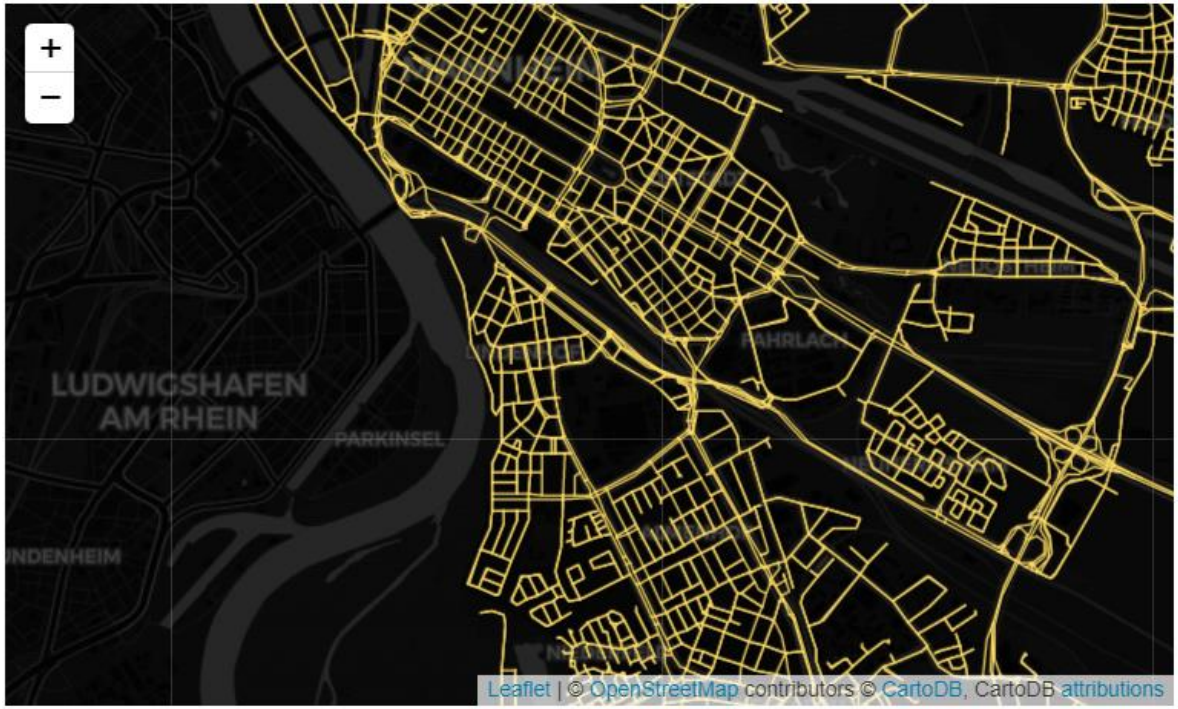
Display Street Network on a Map using Folium

```
m = folium.Map([49.473, 8.475],zoom_start=13,tiles="CartoDB dark_matter")

jsondata = streets.to_json()

style = {'color': '#FFDD66',
         'weight':'1'}

folium.GeoJson(jsondata, style_function=lambda x: style).add_to(m)
m
```



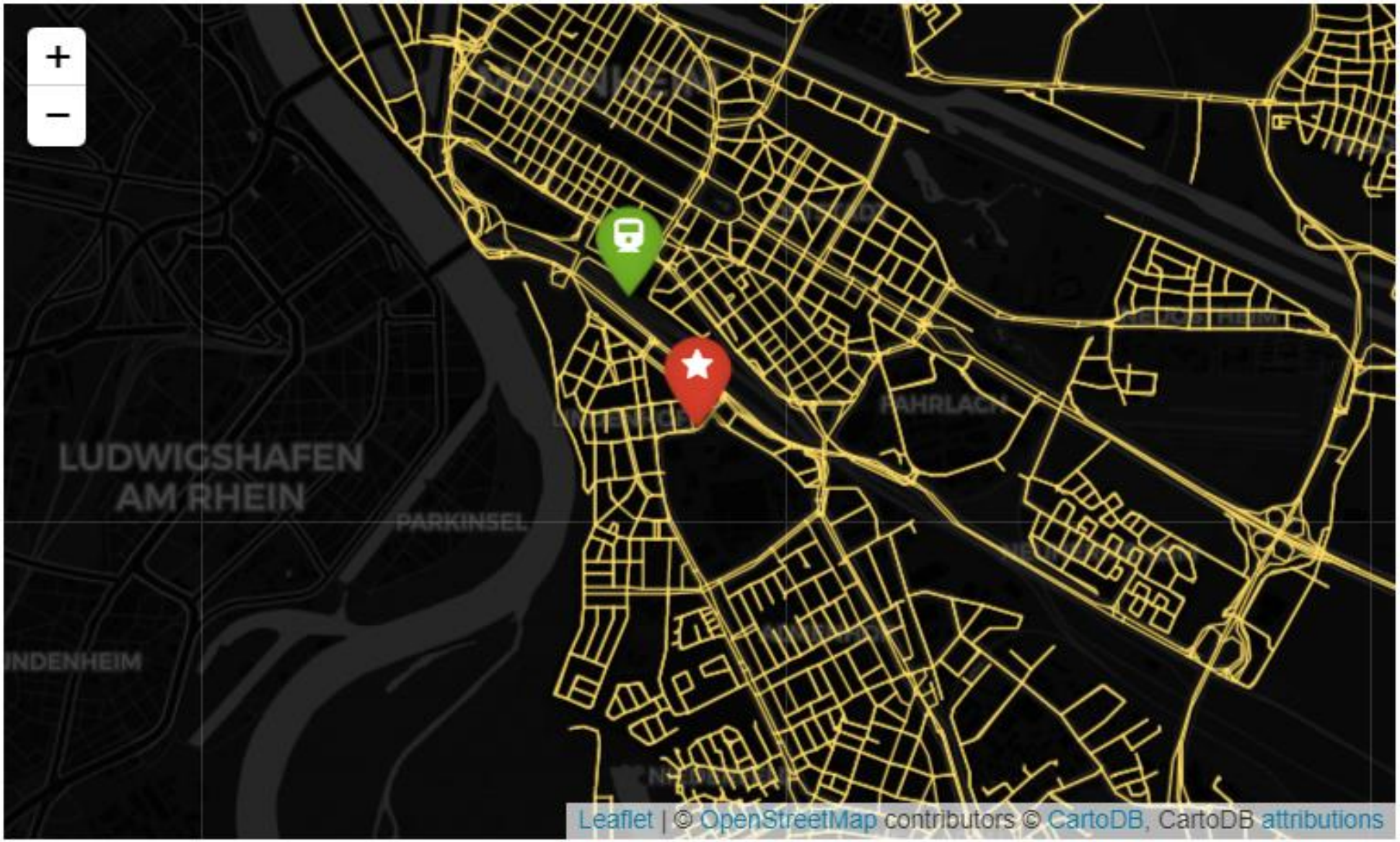
Geocoding

```
# Train Station
pos_1 = ox.geocode('Mannheim HBF, Mannheim, Germany')
pos_1
```

```
(49.4792363, 8.4694107)
```

```
# MAFINEX
pos_2 = ox.geocode('Julius-Hatry-Straße 1, 68163 Mannheim, Germany')
pos_2
```

```
(49.472934, 8.474632)
```


LUDWIGSHAFEN
AM RHEIN
PARKINSEL
Leaflet | © OpenStreetMap contributors © CartoDB, CartoDB attributions

Routing

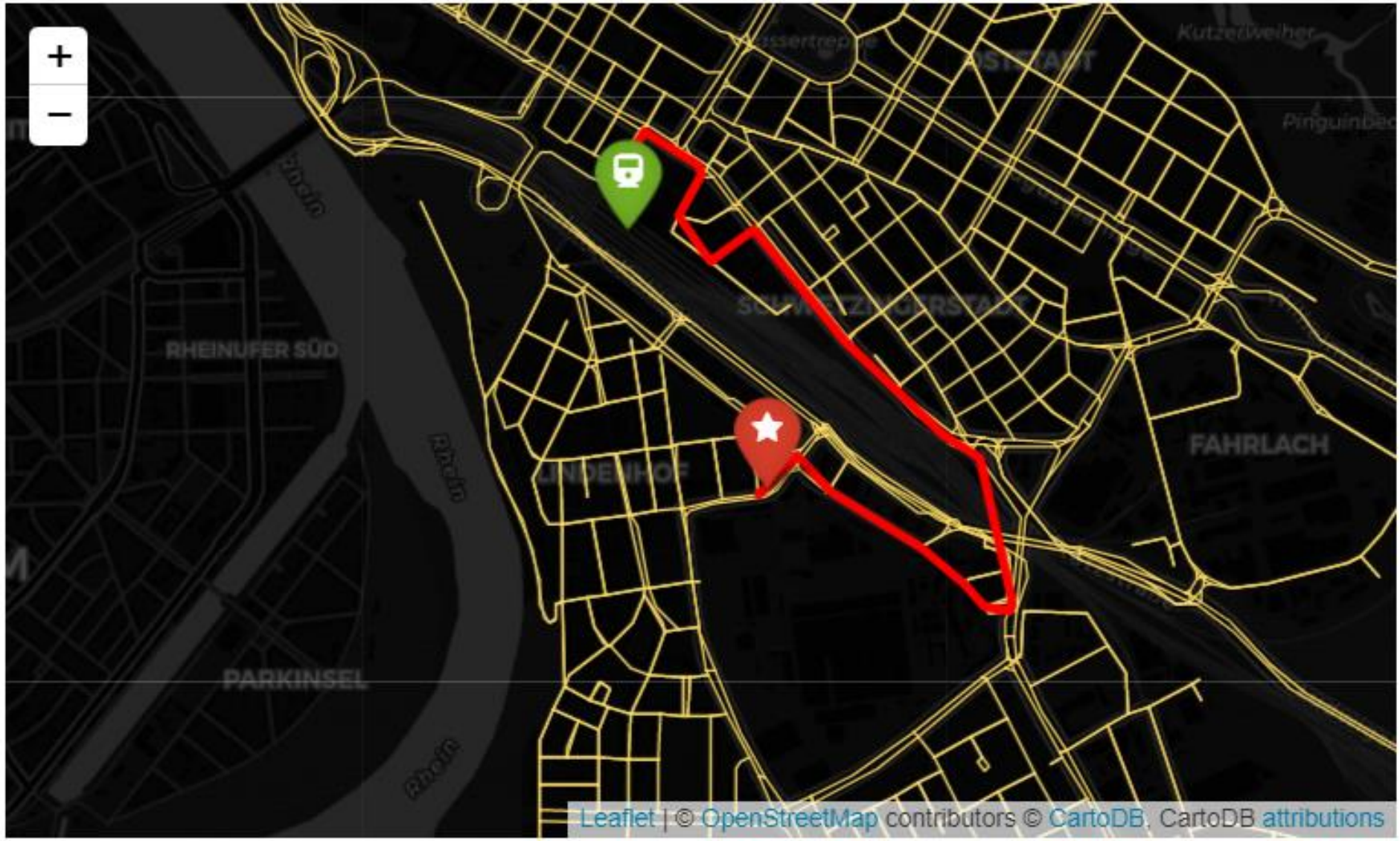
Get the nearest node in the graph (street network)

```
origin_node = ox.get_nearest_node(graph, pos_1)
destination_node = ox.get_nearest_node(graph, pos_2)
```

Then find the route (shortest path):

```
route = nx.shortest_path(graph, origin_node, destination_node)
```

Result (Connected Nodes)



Retrieving Building Footprints

```
buildings = ox.geometries_from_place("Mannheim, Germany", tags={'building':True})

buildings.shape
```

This download takes a while. Result: around 68000 geometries

```
ox.plot_footprints(buildings, figsize=(16,15));
```

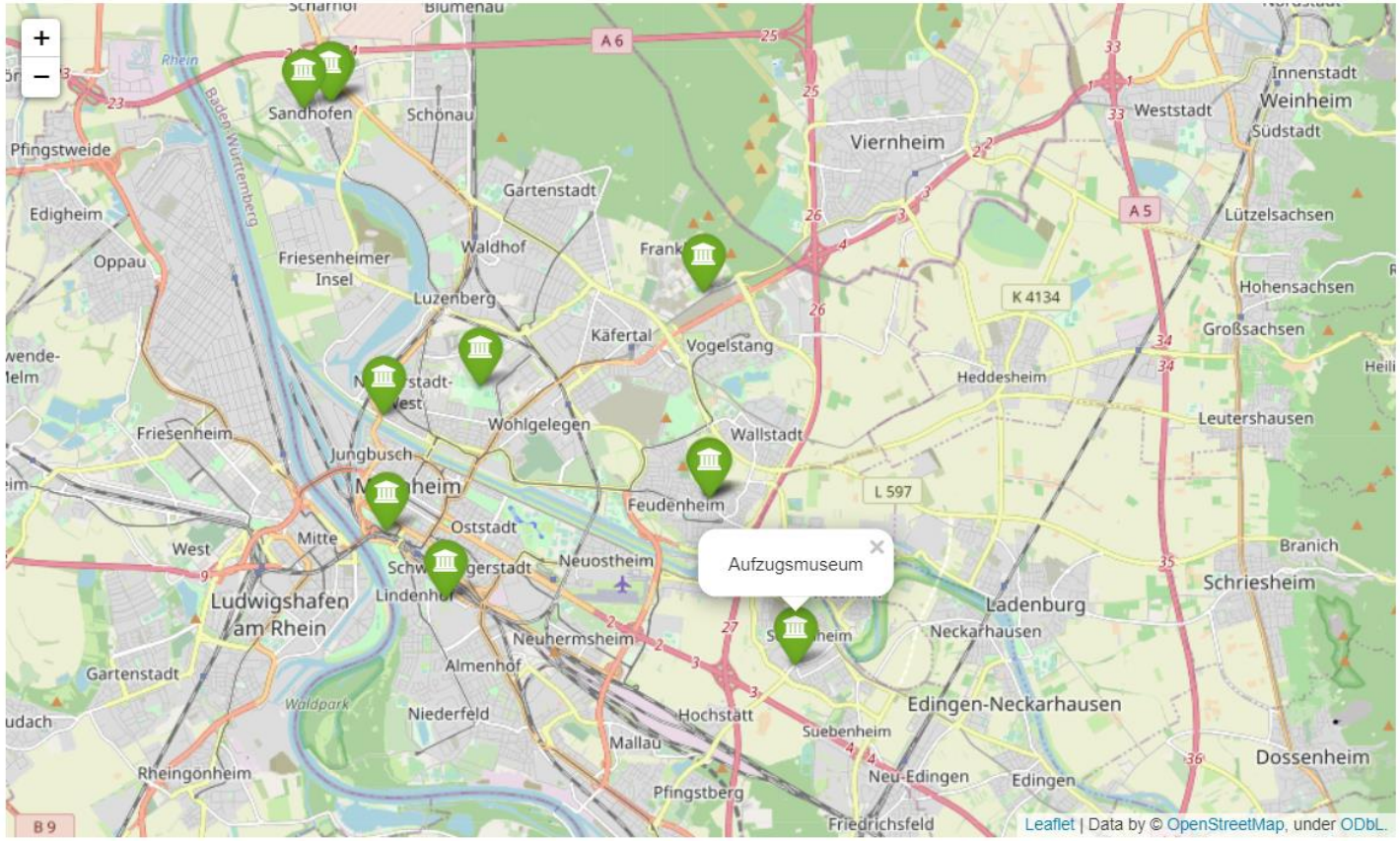
Query specific building footprints

```
museum = buildings.query("tourism == 'museum'")
museum = museum[['name', 'geometry']]
museum.head(20)
```

name	geometry
Technoseum	POLYGON ((8.49748 49.47656, 8.49746 49.47657, ...
Museumsschiff Mannheim	POLYGON ((8.46841 49.49497, 8.46846 49.49493, ...
Museum Zeughaus	POLYGON ((8.46151 49.48768, 8.46142 49.48771, ...
Zephyr - Museum für Photographie	POLYGON ((8.46286 49.48793, 8.46273 49.48779, ...
Kunstverein	POLYGON ((8.48750 49.47906, 8.48792 49.47889, ...
eum Bassermannhaus für Musik und Kunst	POLYGON ((8.46254 49.48758, 8.46239 49.48764, ...
Reiss-Engelhorn-Museum	POLYGON ((8.46161 49.48878, 8.46174 49.48892, ...
Schillerhaus	POLYGON ((8.46089 49.48700, 8.46077 49.48705, ...
Heimatmuseum Seckenheim e.V.	POLYGON ((8.56170 49.46466, 8.56180 49.46458, ...
Bohrmann'sche Schmiede	POLYGON ((8.53425 49.48748, 8.53421 49.48739, ...

If you want locations: pois_from_place

```
museum = ox.pois_from_place("Mannheim, Germany", tags={'tourism': 'museum'})
```



Thanks

Link to the Notebook:

https://github.com/martinchristen/bigdatabbq2021