Data Collection - Population Density (P1)

This code file is dedicated to the second stage of the project, focusing on **collecting and analyzing population density** per station area. The primary goal of this section is to validate the **P1 value**, which aligns with the "Population Density" dimension of the Node-Place model.

Table of Contents

1. Libraries

A comprehensive suite of libraries is imported to provide robust tools for data manipulation, geospatial processing, and visualization.

6.2. Adjusting Population Data Based on Area Proportion

> This code creates a list `adjusted_populations_CB` to store dictionaries:
>
> - It iterates over each station in the `gdf_CB` GeoDataFrame;
> - For each station, it clips the grid cells from `gdf_Population` using the station area's geometry;
> - It calculates the area of each clipped cell and determine the proportion of the clipped area, and adjusts the population `Ind` of each grid cell based on the area proportion.

In [29]:
```
# Initializing a list to store adjusted populations
adjusted_populations_CB = []
```

In [30]:
```
# Iterating over each station
for index, station in gdf_CB.iterrows():
    # Clip the grid cells
    clipped_cells_CB = gpd.clip(gdf_Population, station.geometry)

    # Calculating the area of each clipped cell
    clipped_cells_CB['clipped_area'] = clipped_cells_CB.geometry.area # in square meters

    # Calculating the original area of each cell
    clipped_cells_CB['original_area'] = clipped_cells_CB.geometry.area # in square meters

    # Determining the proportion of clipped area relative to the original area of each cell
    clipped_cells_CB['area_ratio'] = clipped_cells_CB['clipped_area'] / clipped_cells_CB['original_area']

    # Adjusting the population of each cell based on the area proportion
    clipped_cells_CB['adjusted_pop'] = clipped_cells_CB['Ind'] * clipped_cells_CB['area_ratio']

    # Calculating the total adjusted population for each station
    sum_population_CB = clipped_cells_CB['adjusted_pop'].sum()

    # Adding the result to the list as a dictionary
    adjusted_populations_CB.append({
        'Name': station['Name'],
        'Total population': sum_population_CB
    })
```

In [31]:
```
# Converting the list into a DataFrame
df_adjusted_populations_CB = pd.DataFrame(
    adjusted_populations_CB
)

# First rows
df_adjusted_populations_CB.head()
```

Out[31]:

	Name	Total population
0	Arneke	2511.0
1	Cassel	3879.0
2	Renescure	3668.0
3	Annappes	19233.5
4	Sous le Bois	8231.0