SUSTAINABLE DEVELOPMENT GOALS

# Support activities to AACID-UNDP Initiative for SDG monitoring in Andalucía

## Methodological Report

Alberto Quintanilla Cabañero, Smart&City Solutions

AGENCIA ANDALUZA DE COOPERACIÓN INTERNACIONAL
PARA EL DESARROLLO
Consejería de Igualdad, Políticas Sociales
y Conciliación

Instituto de Estadística
y Cartografía de Andalucía
Consejería de Transformación Económica,
Industria, Conocimiento y Universidades

UNDP

# Contents

# 1 Introduction and Scope

This document aims to present and discuss both the process and the achievements of the consulting project with the Andalusian Institute for Statistics and Cartography (*Instituto de Estadística y Cartografía de Andalucía*, IECA) on the creation of public and official information for the follow-up and monitoring of the SDGs in municipal areas within Andalucía.

The three main targets of the project have been i) create a methodology for calculating 3 indicators identified from the European Handbook for SDG Voluntary Local Reviews, particularly adequate to use georeferenced spatial and population data available for IECA, ii) provide inputs on how to communicate and visualize the data and iii) assist on the exchange and capacity-building efforts in coordination with the Andalusian Agency for International Development Cooperation (AACID), IECA and the United Nations Development Program (UNDP) on statistical information production.

From the beginning of the collaboration, the emphasis has been on the collaborative work between the consultant and the IECA scientific staff, seeking to share the knowledge and design reusable, understandable and efficient processes. The project would require a later effort of socio-demographic interpretation of the results, which will is not within the scope of this document.

This report follows the following structure:

- The first part describes the software and hardware tools used in the project, so readers trying to reuse it can better dimension the needed resources.
- Each indicator is described, starting from the input data, methodology used and final discussion. The "Population without green urban areas in their neighborhood" consumes most of the document as it was the most challenging in all technical aspects, and the developed methodology was the base for the "Access to public transportation" indicator as well.
- A brief description of the visualization solution adopted for all indicators.
- Finally, the further efforts planned for information diffusion are seketched.

# 2 Tools

The computation needed by the project has been done using the following minimum hardware:

- Processor: Intel (r) Core ™ i5-6200U CPU @ 2.30GHz 2.40GHz
- RAM 8GB
- Operating System: Windows 10

The following software and tools have been used:

- QGIS (recommended version 3.20.0, or later): QGIS is an Open Source Geographic Information System. The platform was created in 2002 and was established as a project on SourceForge the same year. It currently runs on most Unix platforms, Windows, and macOS. The initial goal of the project was to provide a GIS data viewer with common functions and features, but has already reached the point in its evolution where it is being used for daily GIS data-viewing needs, for data capture, for advanced GIS analysis, and for presentations in the form of sophisticated maps, atlases and reports. QGIS supports a wealth of raster and vector data formats, with new format support easily added using the plugin architecture. It is released under the GNU General Public License (GPL).
- Working with OpenStreetMap requires upgrading QGIS with OSM Downloader Plugin to retrieve OSM data using threads with a simple area selection.

- Accessing the INSPIRE services via QGIS requires the installation of the Spanish Inspire Cadastral Downloader Plugin.
- PostgreSQL (version 13.3): PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that store and scale the most complicated data workloads. The origins of PostgreSQL date back to 1986 as part of the POSTGRES project at the University of California at Berkeley and has more than 30 years of active development on the core platform. PostgreSQL is released under the PostgreSQL License, a liberal Open Source license, similar to the BSD or MIT licenses. Currently, PostgreSQL is a software bundle including:
    - pgAdmin 4 (version 5.2): Open Source administration and development platform for PostgreSQL
    - postgis (version 3.1.2): spatial database extender for PostgreSQL. It adds support for geographic objects allowing location queries to be run in SQL. PostGIS is released under the GNU General Public License
    - pgrouting (version 3.2.0): pgRouting extends the PostGIS / PostgreSQL geospatial database to provide geospatial routing functionality

- Anaconda (versión 1.10.0). Python Development Platform including:
    - Jupyter Notebook 5.7.8: The Jupyter Notebook is an open-source web application that allows creating and sharing documents that contain live code, equations, visualizations and narrative text.
    - Python 3.7.3: Python is an interpreted high-level general-purpose programming language. Designed for emphasizing code readability with its notable use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python consistently ranks as one of the most popular programming languages, and has gained recently traction in the statistics and data science fields.

# 3 Population without green urban areas in their neighbourhood

**Main SDG** **Linked to:**

11 SUSTAINABLE CITIES AND COMMUNITIES

3 GOOD HEALTH AND WELL-BEING

15 LIFE ON LAND

## 3.1 Intro

This indicator describes the share of total population of a city which does not have green urban areas in its neighborhood. It is calculated by analyzing the surface of green urban areas within walking distance from people, and the served population. This indicator is computed considering an area of easy walking distance of approximately 10 minutes of walking time.

For this indicator, the most recent updated working paper (Poelman 2018) presents a methodology that takes into account the spatial distribution of both population and green areas throughout the cities' territory, and also produces indicators about the proximity of the green areas to the urban population.

IECA georeferenced information allows some extra sophistication, by combining the population distribution with a layer of land use data where green areas can be identified and using the roads grid analysis to compute how the former can access the latter.

### 3.1.1 Dijkstra's Algorithm

It is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. It was conceived by computer scientist Edsger W. Dijkstra in 1956 and published three years later. It is the base for optimal route calculation, and therefore it is worth explaining a little bit how it works.

The algorithm exists in many variants. Dijkstra's original algorithm found the shortest path between two given nodes, but a more common variant fixes a single node as the "source" node and finds shortest paths from the source to all other nodes in the graph, producing a shortest-path tree (Cormen 2001)

Let the node at which we are starting at be called the initial node. Let the distance of node Y be the distance from the initial node to Y. Dijkstra's algorithm will assign some initial distance values, usually an outlier value, and will try to improve them step by step.

1. Mark all nodes unvisited. Create a set of all the unvisited nodes called the unvisited set.
2. Assign to every node a tentative distance value: set it to zero for our initial node and to infinity for all other nodes. Set the initial node as current.
3. For the current node, consider all of its unvisited neighbors and calculate their tentative distances through the current node. Compare the newly calculated tentative distance to the current assigned value and assign the smaller one. For example, if the current node A is marked with a distance of 6, and the edge connecting it with a neighbor B has length 2, then the distance to B through A will be 6 + 2 = 8. If B was previously marked with a distance greater than 8 then change it to 8. Otherwise, the current value will be kept.
4. When we are done considering all the unvisited neighbors of the current node, mark the current node as visited and remove it from the unvisited set. A visited node will never be checked again.

5 If the destination node has been marked visited (when planning a route between two specific nodes) or if the smallest tentative distance among the nodes in the unvisited set is infinity (when planning a complete traversal; occurs when there is no connection between the initial node and remaining unvisited nodes), then stop. The algorithm has finished.

6 Otherwise, select the unvisited node that is marked with the smallest tentative distance, set it as the new "current node", and go back to step 3.

## 3.2 Input data

### 3.2.1 Green Areas

#### 3.2.1.1 Exploratory Analysis

According to The Copernicus Urban Atlas (Copernicus), green urban areas are "public green areas for, predominantly, recreational use such as gardens, zoos, parks, castle parks; suburban natural areas that have become and are managed as urban parks". This definition is particularly focused in the urban context, but, due to the regional scope of this project, we needed to expand the range of green areas to other natural spaces that, in many cases, provide the rural population with access to public spaces, such as protected areas. Therefore, we are including forests and similar spaces as green areas.

In order to build this indicator for the whole region, using Urban Atlas Itself as a source poses several issues. The scope of the atlas are mainly urban areas, in contrast to our project that seeks measuring the situation for all kinds of settlements in the first place. Besides, it is not updated regularly (last available data is from 2018). On top of that, the main issue is related to the accessibility of the analyzed areas: being the Atlas based on satellite imaging, it is not possible to tell what green areas are open to the public.

To illustrate this, the following figure shows urban green areas present in the Atlas for the municipality of Salteras (Sevilla):

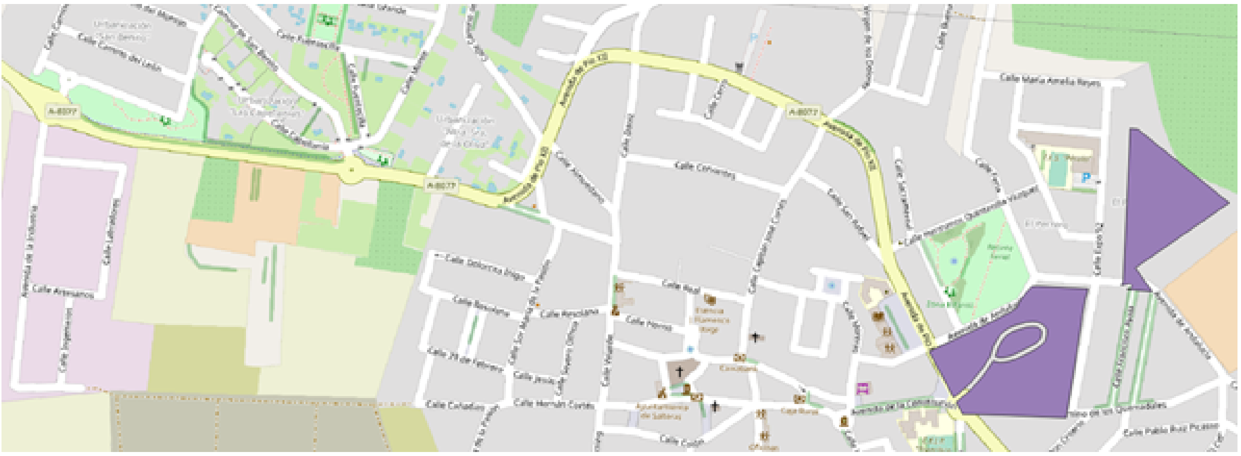


*Figure 1*

Both highlighted polygons are gardens part of private properties. Although they both are mostly green, they cannot be accessed by the general population. Moreover, this source could not identify any of the actual public parks in the town, highlighted in red in the next figure.

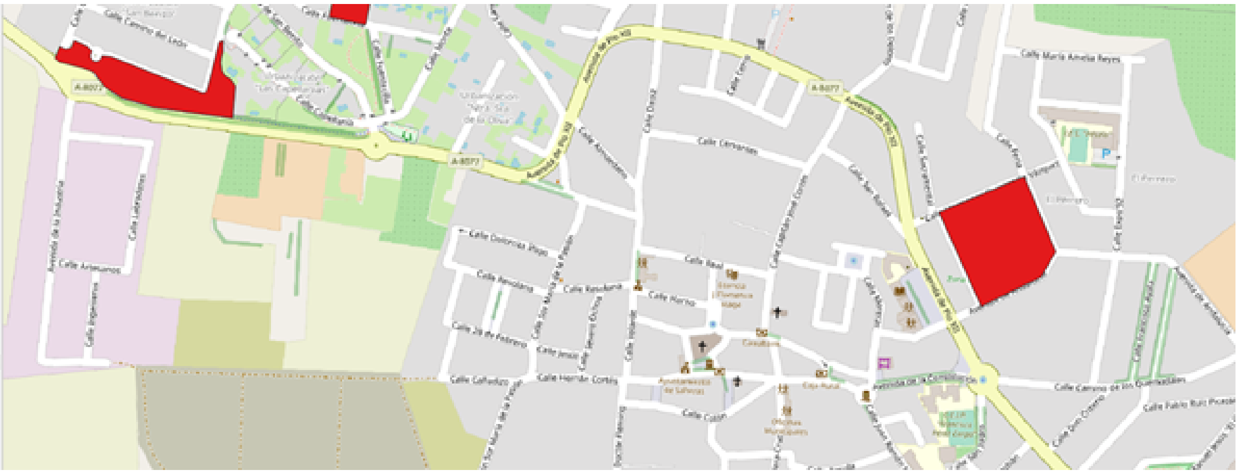


*Figure 2*

Another available source that was evaluated is the Andalucía Reference Spatial Data (*Datos Espaciales de Referencia de Andalucía*, DERA), a set of geographical layers of different nature concerning the Andalusian territory. The following datasets have been analyzed:

- **Urban system. Green Area** (*7. Sistema urbano. 07_06_ZonaVerde*) (DERA): Urban green areas over 0.5 ha. The original source to this layer is Open Street Map, processed afterwards by deleting all areas with a surface less than 0.5 ha and some areas with uses different to public recreation: dog parks, fair parks, roundabouts, country clubs, sport tracks, overlook points, skateparks. Gardens and squares have been included.
- **Natural Heritage** (*11. Patrimonio. 11_07_Enp*): Natural Protected Spaces in Andalucía with a region or state level protection formula: National Parks, Natural Parks, Natural Areas, Suburban parks, Natural monuments, Natural reserves and protected landscapes). Other protection formulas have been included as well (protected areas around Natural reserves, natural monuments, and Doñana National Park). This dataset originates in the Andalucía Environmental Information Network (*Red de Información Ambiental de Andalucía*, REDIAM).

There are some drawbacks in using data from DERA. The first one is the timeliness of the data, this layer was built in 2018 and updates have been detected on the OpenStreetMap original source since. For instance, in Sanlúcar de Barrameda (Cádiz) there are no green areas according to DERA data, while OpenStreetMap is currently listing some of them:

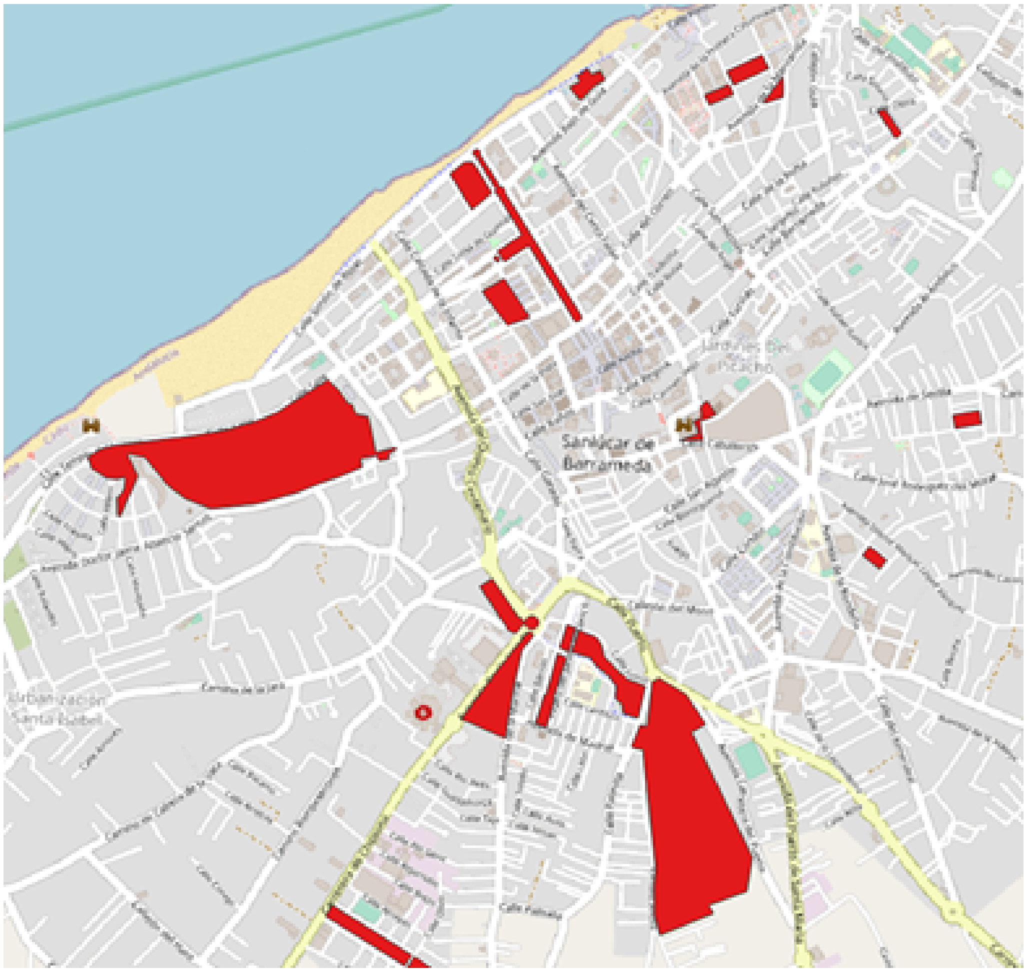


*Figure 3*

There is also some discordance between the sought typology of green areas according to the indicator's definition and those included in this layer.: some areas with recreational use are explicitly excluded, such as squares, dog parks, zoo parks and playgrounds. The minimum surface is also different, 0.5 ha compared to 0.25 ha in the Urban Atlas.

However, this data source looks more appropriate than others available and it has been decided to use it by updating it with up-to-date OpenStreetMap data, and refine the process to tell which areas are publicly accessible and which are not.

#### 3.2.1.2 Data layer build-up

QGIS plugin QuickOSM was used to capture the polygons from OpenStreetMap needed as input, by filtering some of the available tags. Tags in OpenStreetMap describe specific features of map elements (nodes, ways, or relations) and are duples comprising a key and a value. Both items are free format text fields. The key is used to describe a topic, category, or type of feature (e.g., highway or name). The value provides detail for the key-specified feature. Commonly, values are free form text (e.g., name="*Parque de María Luisa*")

Tags used in our query are:

| **Key** | **Value** | **Description** |
|---|---|---|
| leisure | dog_park | A designated area, with or without a fenced boundary, where dog-owners are permitted to exercise their pets unrestrained. |
| leisure | fitness_station | An outdoor facility where people can practise typical fitness exercises |
| leisure | garden | A place where flowers and other plants are grown in a decorative and structured manner or for scientific purposes. |
| leisure | park | A park, usually urban (municipal). |
| leisure | playground | A playground: an area designed for children to play. |
| landuse | cemetery | Place for burials |
| landuse | recreation_ground | An open green space for general recreation, which may include pitches, nets and so on, usually municipal but possibly also private to colleges or companies |
| place | square | A town or village square: a paved open public space, generally of architectural significance, which is surrounded by buildings in a built-up area such as a city, town or village. |
| tourism | zoo | A zoological garden, where animals are confined for viewing by the public. |
| landuse | forest | Managed forest or woodland plantation |
| natural | wood | Tree-covered area (a 'forest' or 'wood') |

Once the layer has been downloaded, it is required to adapt its content to the definition in the Urban Atlas. The tool postgis has been used to perform the following actions:

1. Remove polygons with different tags.
2. Combine (fusion) all smaller polygons that build up a larger one (e.g. those overlapping or separated by narrow streets or paths). This has been achieved by creating a 4.5 m buffer area around the remaining polygons, dissolving them and removing the buffer area again (Poelman 2018).

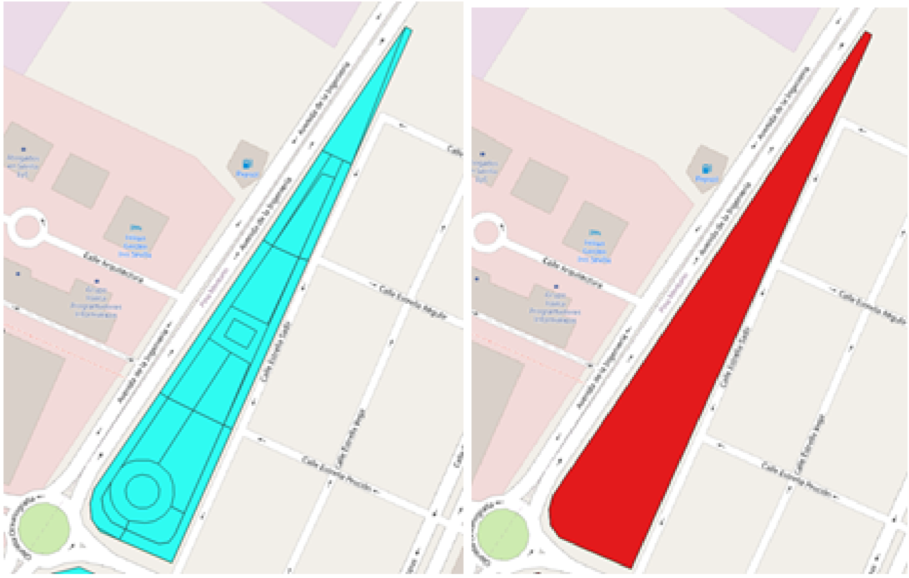


*Figure 4: Overlapping polygons.*

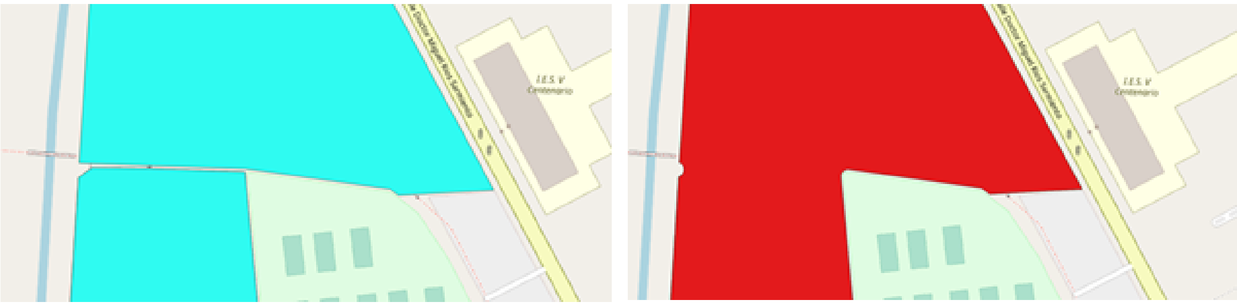


*Figure 5: Green area formed by 2 polygons separated by a narrow path*

3 A minimum mapping unit of 0.25 ha has been considered (that may be split into several smaller patches configuring a single green area, as seen above), so smaller polygons are deleted.

After the layer has been built, some quick checks easily disclose green areas that are not part of the public domain. The key "access" from OpenStreetMap (primary source to tell whether the park is accessible or not) is unfortunately not always provided. Some examples of this are the gardens in *Isla Mágica* (Sevilla), private gardens attached to hotels and inner yards in residential developments. Therefore, another iteration will seek to remove privately-owned land from the currently generated layer.

The Cadastre will be the source of information to find out the ownership of the different areas. In order to carry out this operation, land plots with all real estate privately owned according to Cadastre table 46 (DG Catastro) are labelled as private, considering private owners those with tax identification numbers not beginning with the following letters:

- P. Local Corporations
- Q. Public Administrations

- S. General State and Regional Administration bodies

However, there some cases where these assumptions will be wrong, and some error probability needs to be considered:

- Not all privately owned land has restricted access and vice versa. The *San Telmo* Palace gardens, for instance, belong to a public institution and are not accessible though.
- There are also some publicly owned companies whose tax identification number begins with A (Public Limited Companies) and their property will thus be considered as not accessible. In some cases, e.g. the parcel where the *Cartuja* Olympic Stadium (Sevilla) is located, whose owner is "*Estadio la Cartuja de Sevilla* S.A.". However, it is not possible to tell these public companies from those that are private.
- Parcels hosting mixed ownership properties (there is both public and private real estate inside) have been completely discarded. These cases belong, in general, to large parks in which small shops or restaurants are located, but all share the same cadastral parcel. Since the Cadastre geometry is not available at the real estate level, but at parcel level, it is not possible to tell the polygons of private real estate from those that are public.

Once all polygons of privately owned cadastre parcels have been selected, the following operations are carried out:

1. The intersection between the green areas resulting from the previous section and the private plots is obtained.
2. In some very large polygons made up of several cadastral parcels, it is observed that the plots are separated by a few meters, possibly due to small paths, rivers, so we proceed the same as step 2 above.
3. The spatial difference is made between green areas and the intersection of those green areas with the private Cadastreparcels obtained in the previous step
4. Resulting polygons with less than 0.25 ha are removed again

In some cases, there are minor differences between the Cadastre cartography and the OpenStreetMap, creating some micro polygons where it is slightly smaller.

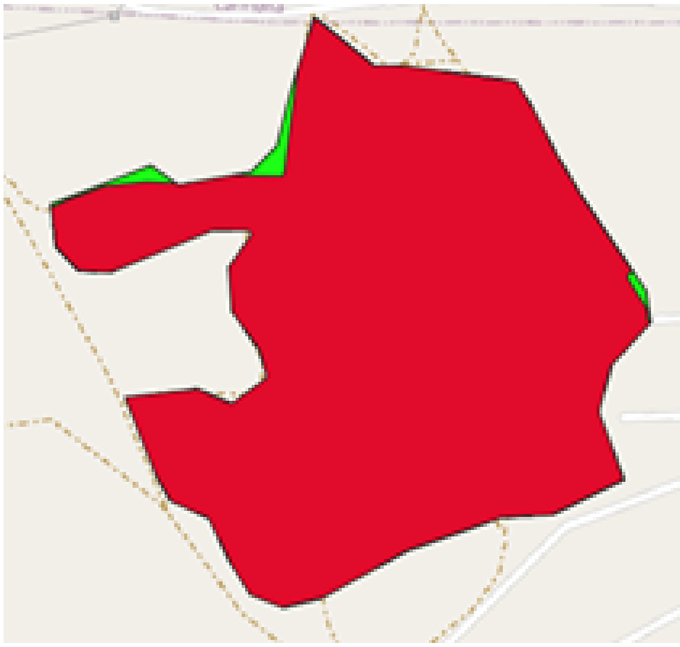


*Figure 6: The resulting green areas layer is shown in green. The removed private areas are shown in red. The small green polygons are very likely mistaken due to differences between cartographies.*

These remaining polygons generally have a very high perimeter to surface ratio, and their surface is small compared to the complete green area. In order to discard them, the resulting polygons representing less than 10% of the total area of the original green area whose perimeter/surface ratio is greater than 1,000 have been deleted. This algorithm works particularly well in urban areas, but there are still a lot of micropolygons in rural areas, which will create some issues when processing this layer in the network analysis below.

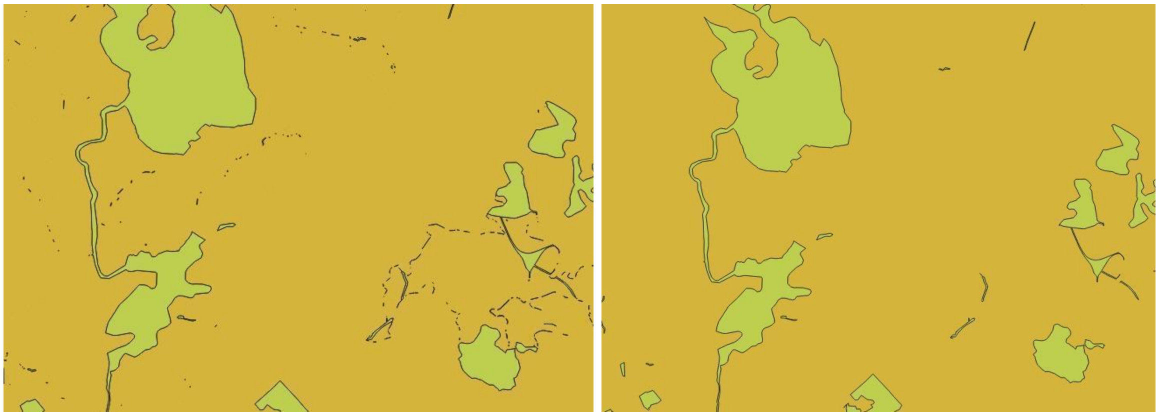


*Figure 7: remaining micropolygons due to minor differences between cartographies*

Last iteration will seek to make sure all areas subject to Public Hydraulic Domain (*Dominio Público Hidráulico*) (CH Guadiana, 2020) are included. In order to do that, we consider all polygons with key landuse=forest that besides have the word "*rio"* in their key "name" (if provided).

### 3.2.2 Buildings

The main innovation in this indicator compared to the Urban Atlas Methodology is that we have access to the location and population structure of individual buildings in Andalucía. Therefore, this data will be used to determine what share of population lives in the area that can be reached on foot from a park in the specified time.

Buildings data is compiled in one single layer, but has different sources:

- Digital Unified Street Map of Andalucía (*Callejero Digital de Andalucía Unificado,* CDAU): this is the set of geographical data of Andalucía allowing the location in the territory of any geographical object (and its associated variables) with a postal address, with an approximation at the building level. In order to generate this data, CDAU linked GESTA (Administration for Territorial Entities of Andalucía) database with a variety of geographical databases, such as local and regional street maps. GESTA is a complete repository of postal addresses created and managed by IECA with information from the Population Registry of Andalucía, the Street, the Electoral Census, the Directory of Companies and Establishments with Economic Activity in Andalucía and other alphanumeric information of cadastral origin.
- Cadastral data.
- Census 2011

Therefore, there is some heterogeneity in the buildings' data that will need to be managed when processing it.

### 3.2.3 Roads

The only source of data used for the roads grid is CDAU. Streets and roads information in CDAU comes from the General Infrastructure Administration of Andalucía and the Local Administrations. The indicator aims to measure pedestrian accessibility, so it makes sense removing some roads that cannot be walked such as highways and speedways. Therefore, all lines labelled "*Autopista*" or "*Autovía*" in the CDAU tables have been deleted.

It is worth mentioning that CDAU is continually updated via the maintenance web platform developed specifically for the project with the aim of reducing the effort of local technicians as much as possible.

## 3.3 Methodology

### 3.3.1 Approach

The geographical proximity to green areas or parks may be misleading when trying to find out whether there is access: some architectural or infrastructure barriers can be found, the urban grid can be disconnected or inefficiently configured, causing segregation of the population. Network analysis is thus necessary to determine real access to the resources.

That said, two different approaches or computation methods can be applied:

1 Single common service area around the green spaces.

2 Finding a path (or paths) to the nearest green area individually for each building.

The first approach is adopted, as it intuitively seems more efficient to perform a single calculation than iteratively compute network analysis for a huge amount of buildings. However, this method also has some drawbacks (in practice, service area computation needs to be done also a similar number of times when starting from the green areas due to the lack of information on green areas' entrance points), and an approximation to the second approach will be discussed and sketched below as well.

One of the main challenges of the project is dealing with large datasets, which can lead to long computation times. Our aim is running the algorithms with the largest possible data batches, in order to avoid undesired edge effects (Figure 8). When possible, all operations have been performed using layers and tables containing all Andalucía data. However, some time-consuming steps could not be done this way and have been split into provinces. This will be highlighted during the methodology description.

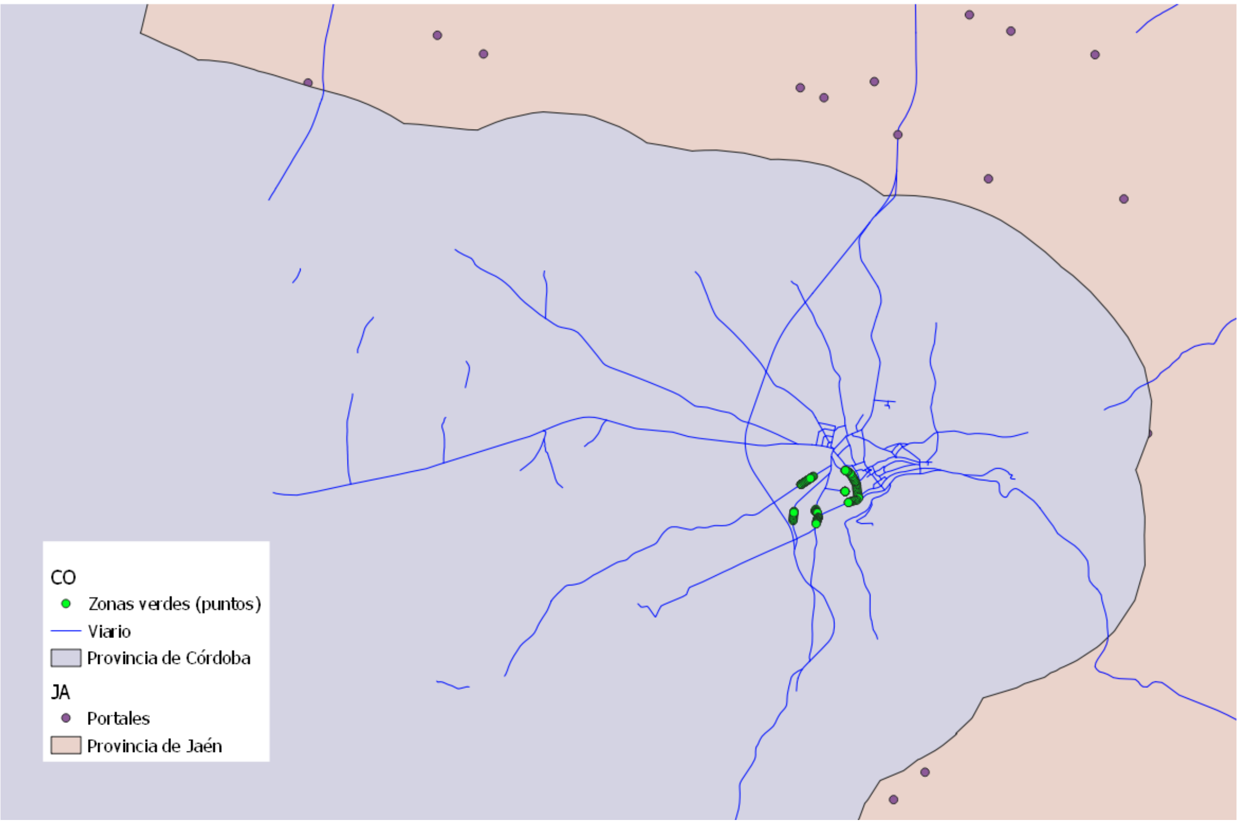


*Figure 8: Edge effect: buildings in Jaén whose closest green area is in another province*

### 3.3.2 Data simplification and fine tuning

The following procedure has been designed thus to minimize the amount of data managed. Parts of it are not essential in the methodology and the results would be similar if not applied. The reader who wants to repeat this process for a single city or a smaller area will probably not need these simplifications to achieve acceptable computing times.

#### 3.3.2.1 Roads Data

According to the definition, road lines further than 833 m (distance that can be walked in 10 minutes by the average person) from any green area will not be used in the analysis, as it would be the maximum theoretical distance that could be covered in case there was a straight road starting from the green area. Therefore, these stretches can be removed to lighten up the data tables.

This has been performed using QGIS, by creating an 833 m buffer around all green areas and selecting only the lines that intersect with it.

Before exporting the resulting layer (or layers) to the postGIS database, QGIS has also been used to convert to singlepart geometry.

#### 3.3.2.2 Green Areas Data

There are two main issues for the use of the green areas in pgrouting network analysis. Polygons (such as those forming the green area layer) cannot be used as parameters for pgrouting-based functions in the first place. Moreover, green areas usually do not intersect with the roads (green areas boundaries are parallel to streets in the most common cases). Therefore, we need to define how we transform these areas into points within the roads that can be used as the origin of our network analysis. The following steps have been performed using QGIS:

1 Polygons have been transformed into their perimetral lines, and these forced to singlepart.

2 The obtained lines have been split into points separated by a regular distance. Two different point steps have been applied:

   2.a Every 10 m for green areas smaller than 102 ha. This limit has been chosen to make sure all urban parks are processed this way, and this is the largest surface found in the DERA's Urban System green areas layer.

   2.b Every 50 m for all other parks

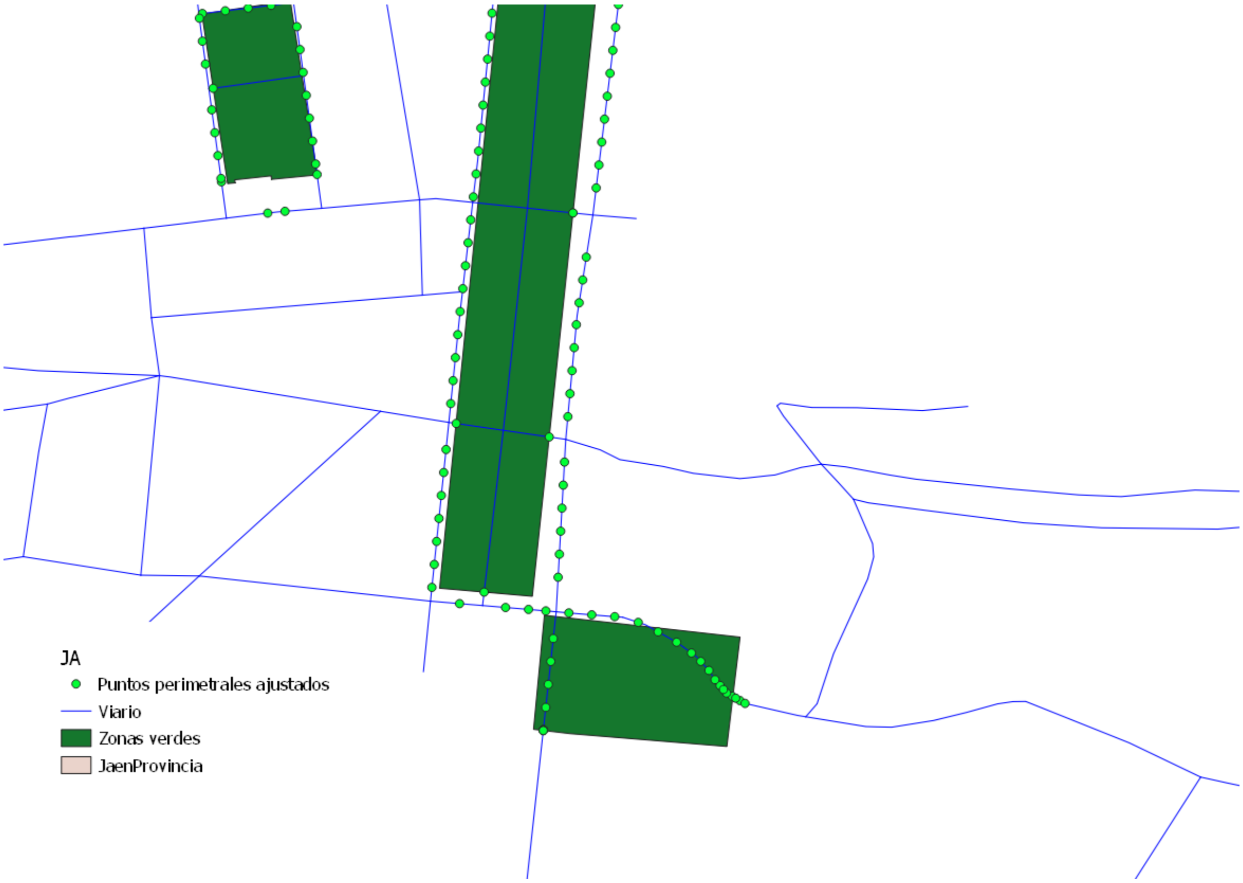


*Figure 9: green areas modelled as perimetral points in Huelma (Jaén)*

3. Complementary to the roads filter discussed above, green area perimetral points further than 833 from any road are deleted.
4. QGIS has finally been used to snap the resulting points to the nearest road.

After creating the green area perimetral points layer, a quick inspection shows many isolated points deriving from the micropolygons that remained in the green area data layer creation. In some provinces these points are almost 50% of the total, artificially increasing computation time.

### 3.3.2.3 Buildings data

Buildings layer is formed by points (supposed to be the building access point). Same as with the green areas perimetral points, they need to be next to a road, so they have been moved to the closest road using the Snap to geometry QGIS tool.

There is an alternative with postgis code but after some tests it has shown worse performance.

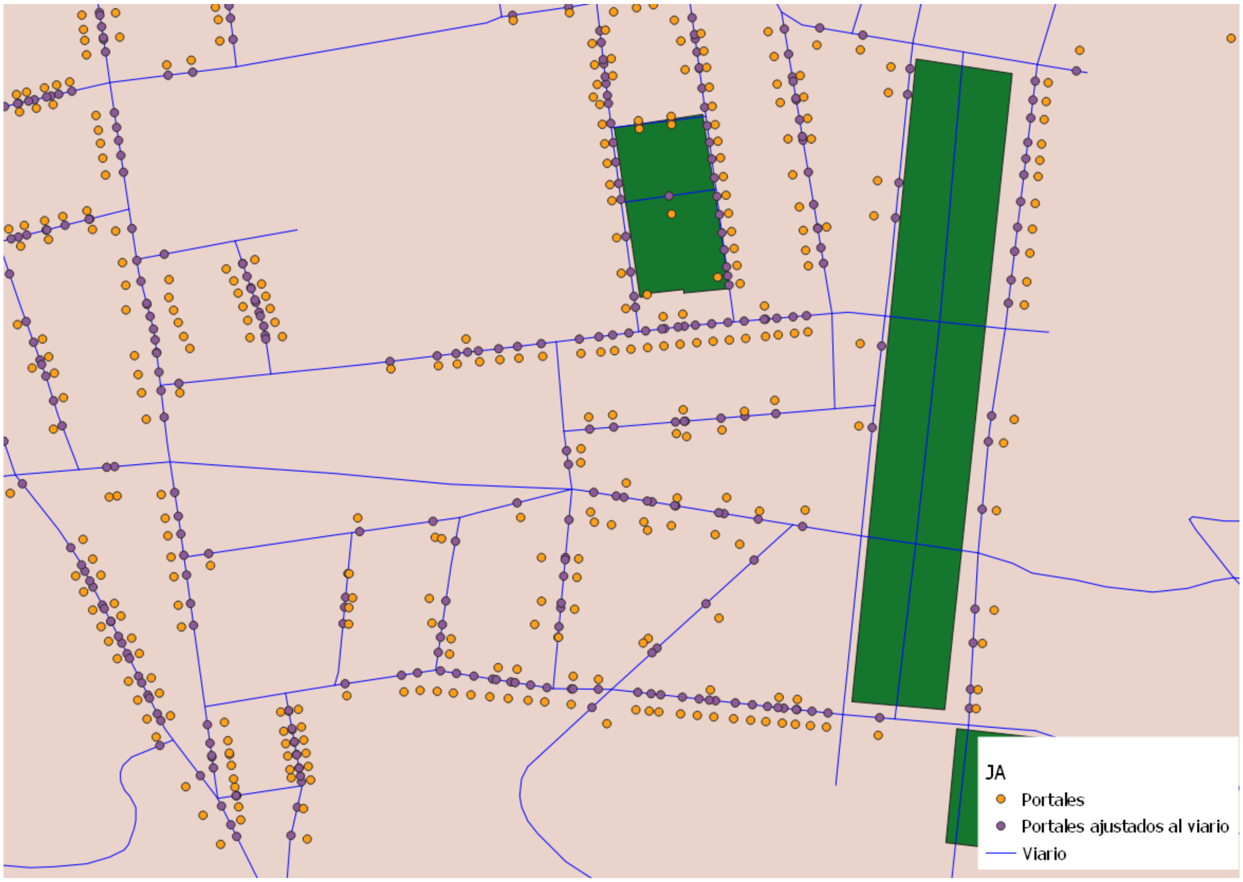


*Figure 10: building access points snapped to the roads geometry*

## 3.3.3 Topology creation

Routing is a graph problem, not a pure geometry problem. Starting from the the original road geometries a graph representatoin is built (Network Topology), suitable for the use of traditional routing algorithms. Depending on the focused problem we may choose different weights assigned to traversing the edges, put restrictions on turns around edges, assign different classes to the edges (residential, freeway, toll bridge, whatever), etc.

The process of extracting the necessary information from the spatial tables is referred to as "building network topology". Pgrouting provides the function pgr_createTopology to build this based on the geometry information. It requires the following parameters:

- Edge_table: network table. In our case, it is our road grid table.
- Tolerance: Snapping tolerance of disconnected edges. (in projection unit). It is used for deciding that multiple nodes within the tolerance are considered the same node.
- the_geom: Geometry column name of the network table. Default value is the_geom.
- Id: Primary key column name of the network table.
- Source: Source column name of the network table.
- Target :Target column name of the network table. These two columns are built during the execution.

As a result, the topology is stored in two tables:

- Edge_table: modified by the function (filling source and target columns with the vertices defining the edge. Edges are the links connecting the nodes of the grid.
- Vertices_table: storing the nodes of the topology.

Pgrouting also provides functions to analyze and repair (if required) the topology.

- pgr_analyzeGraph: Analyzes the network topology, and finishes filling information in the vertices table that will lighten the following steps.
- pgr_nodeNetwork: usually GIS data is often not "noded" correctly. This will create invalid topologies, which will result in routes that are incorrect when trying to use them with pgrouting. With "noded" we mean that at every intersection in the road network all the edges will be broken into separate road segments. There are cases like an over-pass and under-pass intersection where you cannot traverse from the over-pass to the under-pass, but this function does not have the ability to detect and accommodate those situations. This function reads the Edge table we have created in the first step, that has a primary key column id and geometry column named the_geom and intersect all the segments in it against all the other segments and then creates a table edge_table_noded.

After the corrected ("noded") table has been selected, the topology is generated again to create all new vertices.

### 3.3.4 Starting points processing

Most pgrouting functions are oriented to work with nodes in the topology. However, this limitation will not work for us as distance to vertices from both buildings and green area perimetral points is relevant to the problem. The pgrouting family of functions "withpoints" allows us to provide also points located in the edges, but as far as network analysis does not work with geometries, they need to be transformed into something the functions can understand. We need to create duples:

- Edge: the edge where the point is located
- Fraction: Value in <0,1> that indicates the relative position from the first end point of the edge. If 0, the point will be located in the source point of the edge. If 1, it will be in the target.

Creating these duples is a time-consuming process we are performing with postgis using the ST_LineLocatePoint on batches of 10,000 points.

In order to further reduce the amount of points processed, once all the fractions have been computed we can group the similar ones, as illustrated in the figure.

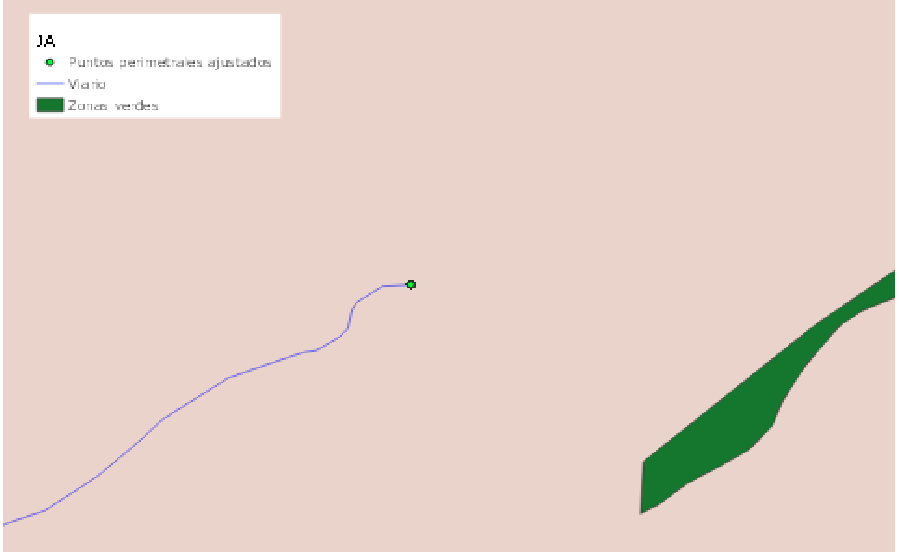


*Figure 11: all perimetral points coming from the green area are snapped to the same one in the road close to it.*

### 3.3.5 Network analysis

The function pgr_withPointsDD allows the use of points and, using Dijkstra algorithm, extracts all the nodes and points that have costs less than or equal to the value distance from the starting point. The edges extracted in the execution build the corresponding spanning tree.

The function is called with the following parameters:

- edges_sql: query (as text) providing the complete topology used. This coincides with our noded topology table adding cost columns (in this case, they are simply the length of the edge).

```
SELECT id, source, target, st_length(geom) as cost,-st_length(geom) as
reverse_cost FROM viario_simple_noded
```

- points_sql: query (as text) providing the vertices in the topology and the points that should be considered. In our case, we are only providing points

```
SELECT id AS pid, id_via AS edge_id, fraction from fractions
```

- start_vid: an array of the points and/or vertices that will be taken as starting points. As a convention of signs, if positive they are understood as vertices and if negative, as points. Therefore, in our case we are only providing negative ids.
- distance: maximum distance to be traveled
- directed: if TRUE, edges can be traveled in both directions.
- details: if TRUE, the output includes the intermediate nodes.

Execution time of pgrouting functions increases exponentially as the density of the topology (number of vertices and nodes) increases and proportionally with the amount of starting points (Nestar 2015). Therefore, we have divided the process again into batches of 10,000 fractions.

### 3.3.6 Pgrouting execution results analysis and processing

pgr_withpointsDD result is only partial, and cannot be used directly to create a service area and intersect it with the building points:

- The calculation only includes all vertices within the distance, and the consumed cost (distance) to reach them. There might be points beyond those vertices that can still be reached.
- Not all possible edges are used to arrive at these vertices.

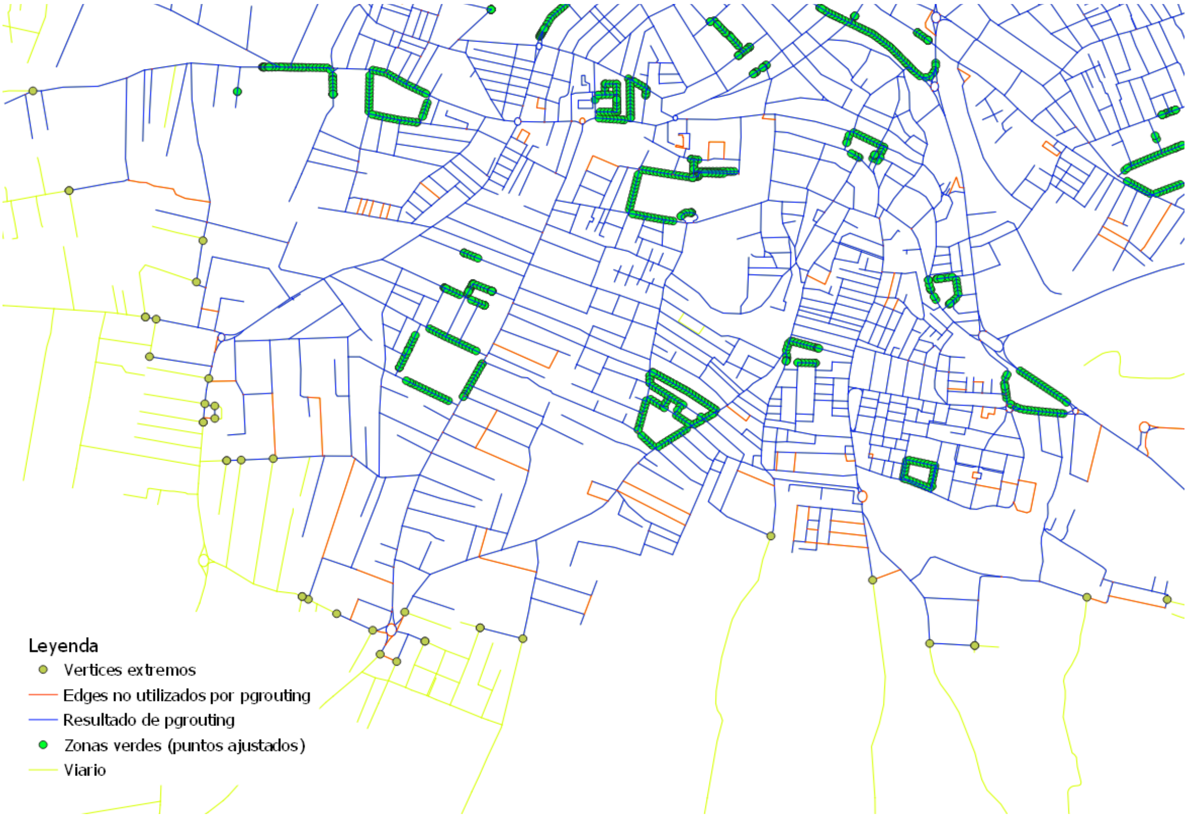

Leyenda
● Vertices extremos
— Edges no utilizados por pgrouting
— Resultado de pgrouting
● Zonas verdes (puntos ajustados)
— Viario

*Figure 12: service area computed by pgrouting, non-used edges and furthest reached vertices in Chiclana de la Frontera (Cádiz)*

As a matter of fact, not all edges are required to arrive at all the intermediate vertices, and pgrouting chooses only the ones with least cost. Therefore, some streets are not included in the results table. To complete this table, we need to find these unused edges that still can be traveled within the specified distance:

1. Query for the edges that have not been included in the result but whose two end nodes have, along with their length.
2. Query the minimum distance necessary to reach these nodes.
3. Finally, add all the edges whose length is less than what was left up to the limit distance from either of the two nodes.

### 3.3.7 Final intersection with building points

The edges found so far are those included completely within the parks service area, so it is already possible to intersect the building points with the resulting geometry.

However, there are still some points within the distance starting from the last reached vertices, so the last step is creating a buffer around those vertices and intersecting it with both the buildings layer and the edges starting in those vertices. This process has been coded with a Python Jupyter Notebook, as it is necessary to provide a different radius for each vertex (the result of subtracting to 833 m the cost needed to reach that specific vertex).

### 3.3.8 Indicator calculation

The union of both results is the complete set of buildings with access to green areas. It is possible to join this table with the confidential data maintained by IECA about people living in each portal segregated by gender and age groups. The municipality where the building is located is added to the table too, so the population (per group and total) with access to green areas is computed.

In parallel, the population living in the buildings for each municipality and gender/age group is calculated and linked to the former table.

3 indicators are then calculated:

- Total share of population without access to green areas in their neighborhood:

*(total population - population with access) / total population*

- Gender difference in access:

*[(total women - women with access) / total women ]- [(total men- men with access) / total men]*

- Total share of population older than 65 without access to green areas in their neighborhood:

*(total over 65 - over 65 with access) / total over 65*

## 3.4 Alternative algorithm: iterative calculation of building service areas

As mentioned above, the opposite approach (starting to define the service areas from the building access points) could be, in principle, not so different in terms of the amount of analyzed points and thus, computation time. In fact, there are approximately 2 million buildings in the analyzed layer, and the total green area perimetral points used for the calculation is near that figure. However, the strategy is completely different. It is not the aim of this report to document this alternative algorithm, but we will try to sketch how it might be implemented, as the algorithmic design is a little more complex.

1. First, it makes no sense just mindlessly analyzing all building points sequentially. Although pgrouting functions support one-to-many signatures (one starting point, many targets) they should be somehow hierarchically associated to the green areas so we limit the amount of target points. The easiest way is to sequentially loop all green areas (when splitting them into points, the original id of the park can be saved) and analyze its area of influence.
2. By definition, this area of influence could be implemented as a buffer of 833 m around the park (for simplification, this buffer could be created around the centroid of the park with

*radius = 833 + distance (centroid, furthest point)*

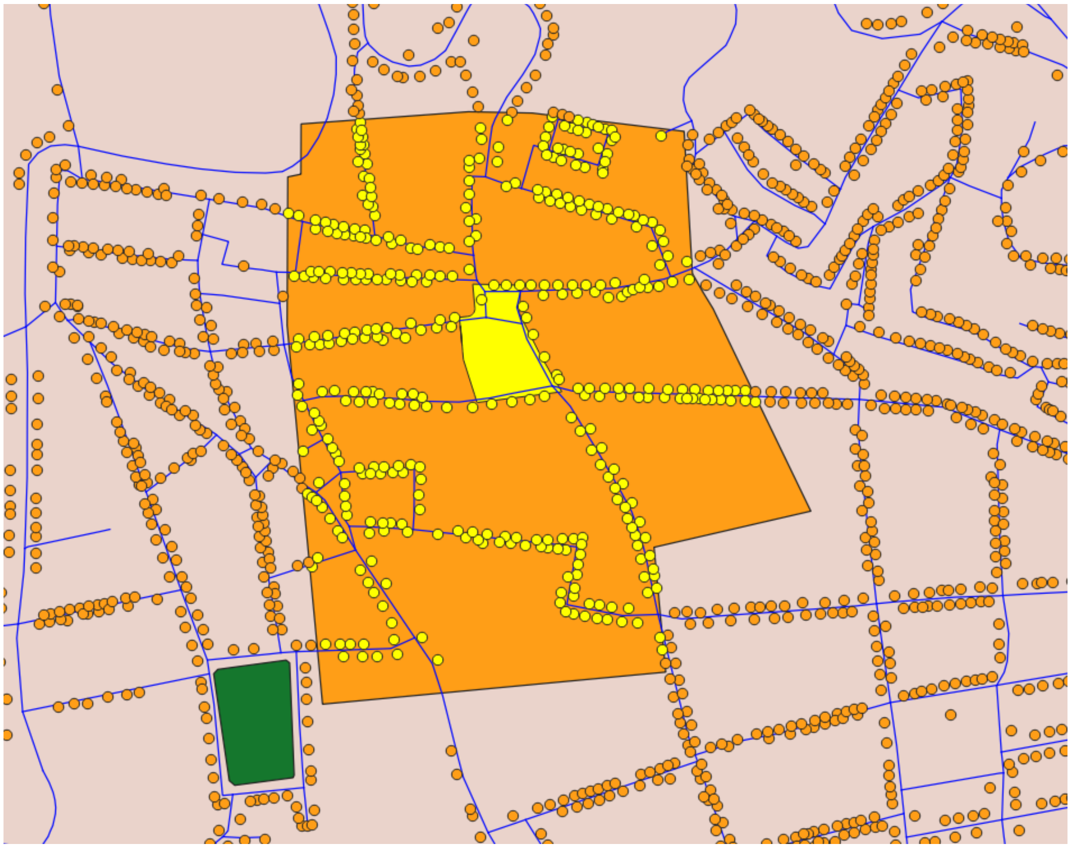


*Figure 13*

3 Once all the building points within this buffer have been located, network analysis begins from the furthest to the closest, with all the park points as destination in the pgr_withpointsdd function.

4 If the function succeeds, the building is labelled as accessible, and (and this is the key optimization of this algorithm) **all buildings intersecting with the followed path are also labelled** the same.

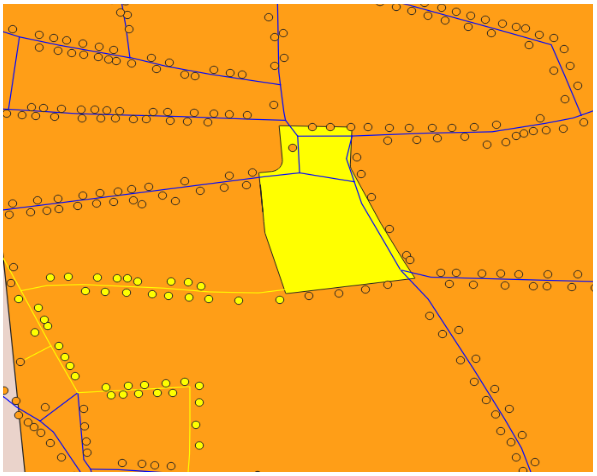


*Figure 14*

5 Testing continues towards the centroid of the park, analyzing only the buildings that are not labelled, until there are no more unlabeled buildings in the buffer. Hence, there are less buildings to test as the tested points are closer to the park.
6 Process continues with the next park, testing only unlabeled buildings.

The main advantage of this algorithm is that the network analysis is performed for small batches of points and a limited area (and thus we have the capacity to limit the road grid used), precisely the two items increasing computation times. Besides, there is a significant optimization by removing points during execution.

The potential risks, in comparison with the green-area-centric approach are derived of the number of operations that must be repeated many times (in the order of 10,000 - 100,000 for Andalucía): ordering points by distance, individually executing pgrouting functions (compared with the ca. 100 times required in the other algorithm) etc. that may cut down the time savings.

## 3.5 Conclusions and further development

This study could be repeatable for any municipality in Spain provided availability of OSM data, Cadastre information and population distribution data. The authors identify general availability of first two sources, whereas population distribution data is not frecuently produced by national or regional statistical offices. However this indicator and its methodology require further analysis to conclude its meaningfullness to describe rural areas context. Initial results reveal a potential bias of the indicator towards urban contexts, not capturing properly social, recreational or open air spaces available in rural areas.

This indicator was previously calculated in the framework of the referenced paper *A walk to the park? Assessing access to green areas in Europe's cities* (Poelman 2018) using 2018 data and a simpler approach to population distribution. There is some overlap in the most populated Andalusian cities, compared in the following table:

| | *Poelman 2018* | *IECA 2021* |
|---|---|---|
| *Algeciras* | 32,31% | 23,87% |
| *Almería* | 7,52% | 4,83% |
| *Cádiz* | 3,64% | 0,00% |

| | | |
|---|---|---|
| *Córdoba* | 6,80% | 6,46% |
| *Dos Hermanas* | 9,27% | 4,77% |
| *Fuengirola* | 10,39% | 13,47% |
| *Granada* | 4,24% | 0,40% |
| *Huelva* | 2,49% | 0,43% |
| *Jaén* | 12,49% | 1,30% |
| *Jerez de la Frontera* | 3,57% | 4,50% |
| *Línea de la Concepción, La* | 14,40% | 56,38% |
| *Málaga* | 5,15% | 2,42% |
| *Puerto de Santa María, El* | 7,16% | 14,76% |
| *Sanlúcar de Barrameda* | 16,67% | 19,48% |
| *Sevilla* | 5,21% | 0,50% |
| *Torremolinos* | 0,54% | 3,19% |

# 4 Access to public transportation

**Main SDG** **Linked to:**

11 SUSTAINABLE CITIES AND COMMUNITIES

3 GOOD HEALTH AND WELL-BEING | 8 DECENT WORK AND ECONOMIC GROWTH | 10 REDUCED INEQUALITIES

## 4.1 Intro

The indicator estimates the share of the population living within the administrative boundaries of a city or an urban center having access to a specific level of public transport service. It is based on the frequency and the mode of transport and it establishes several levels of service. The proposed indicator can be calculated with the method elaborated by Poelman and Dijkstra (Poelman 2015), similar to the one used in the green areas indicator.

Therefore, the findings and methodology from the former indicator are also applicable to this one. However, the main barrier is the lack and heterogeneity of data sources: urban transit systems are operated by a variety of administrations with different policies regarding transparency and data sharing. Hence, the focus with this indicator will not be a thorough calculation at the region level, but exploring data sources and showing the possible results and advantages of performing such assessment.

## 4.2 Input Data

Due to data transparency and availability of different transportation modes, the largest cities in Andalucía are suitable for demonstrating this indicator. The city of Málaga has been chosen as a pilot due to the availability of both Metro and Bus networks, the good quality of their open data publication and, particularly, the adoption of a *de facto* data sharing standard, the Google Transit Specification. The city of Sevilla has been researched afterwards and also meets the same features, but their open data service was unavailable during the early stages of this development.

The General Transit Feed Specification (GTFS), also known as GTFS static or static transit to differentiate it from the GTFS real-time extension, defines a common format for public transportation schedules and associated geographic information. GTFS "feeds" let public transit agencies publish their transit data and developers write applications that consume that data in an interoperable way.

A GTFS feed is composed of a series of text files collected in a ZIP file. Each file models a particular aspect of transit information: stops, routes, trips, and other schedule data. From the specification, we will be using the following files

- Stops.txt: Stops where vehicles pick up or drop off riders. Also defines stations and station entrances. Contains several fields that allow stop identification and geographical location.
- Stop_times.txt: Times that a vehicle arrives at and departs from stops for each trip. Besides identifying the stop and trips, it stores all the information regarding arrival and departure times at a specific stop for a specific trip on a route. If there are not separate times for arrival and departure at a stop, the same value is used for both. For times occurring after midnight on the service day, enter the time as a value greater than 24:00:00 in HH:MM:SS local time for the day on which the trip schedule begins.

This information is available in Málaga only for bus service. The metro stops information has been created manually from Google Maps data.

## 4.3 Methodology

### 4.3.1 Creation of frequency class information per stop

Bus stops are classified according to how frequently the vehicles depart from there. This process still has no geographical component (although latitude and longitude data are saved for later processing) and is thus performed with postgis.

1. Import stops and stop times files into postgresql tables.
2. A new table is created with the amount of times a bus departs between 6:00 and 20:00 (grouping the stop times by stop).
3. A new column is calculated with the hourly departures rate (considering 14 hours of service).
4. Another column is added with the classification of the stop according to the frequency
    - Class 3: more than 10 services / hour
    - Class 2: 4-10 services / hour
    - Class 1: less than 3 services / hour
    - Class 0: no services

In order to translate all this data to geographical format, it has been imported into QGIS specifying the columns lat and long.

### 4.3.2 Network Analysis

The network analysis is performed exactly the same way as in the green areas indicator, considering the stops as equivalent to the green areas perimetral points. In the Málaga environment it has not been necessary to optimize the datasets by deleting parts of the road grid. However, for further calculations with larger scopes, the same kind of previous treatment is advised.

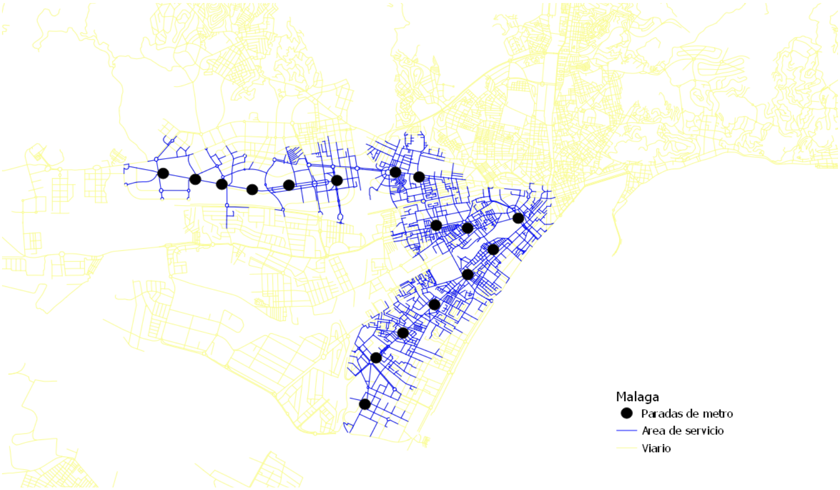


*Figure 15: Metro of Málaga service area*

The applied thresholds are different according to the analyzed transportation mode:

- Bus stops: 5 minutes walking distance (416.5 m)
- Metro: 10 minutes (833 m)

### 4.3.3 Indicator Calculation

The final set of buildings included in the service area must be determined separately for each transportation mode and frequency class, i.e. In the case of Málaga we would have 4 different service areas. Therefore, by overlapping them 4 different indicators can be provided:

- Total share of population without access to metro in their neighborhood
- Total share of population without access to class 3 bus service
- Total share of population without access to neither class 2 nor 3 bus service
- Total share of population without access to bus service

Each one of these indicators could be further disaggregated by gender and group edges same as the green areas accessibility.

## 4.4 Conclusions and further development

This study intends to be repeatable for any other city complying with GTFS and with availability of population distribution data, which for Andalusian municipalties is collected by IECA. The authors have identified other cases, such as Sevilla, where these conditions exist, and the procedure could be run. However, the indicator has a strong bias towards urban contexts, it is not sure that applying this methodology to less urbanized areas (if data were available) would produce a meaningful result.

Eventually, this research might be extended with real or approximate transit use data, if a source of enough quality is not found. Comparing the service areas with real usage data could unveil the habits of the population and the real influence of distance and accessibility.

# 5 New buildings

**Main SDG** **Linked to:**

7 AFFORDABLE AND CLEAN ENERGY

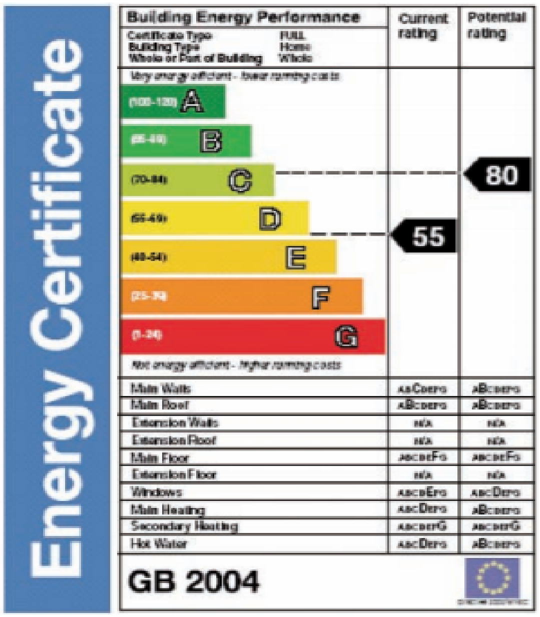
12 RESPONSIBLE CONSUMPTION AND PRODUCTION

11 SUSTAINABLE CITIES AND COMMUNITIES

## 5.1 Intro

The reference paper contains a dataset with the Percentage of residential houses, which have been built after 1980, based on the National Census and building stock statistics, updated 2011. The possibility to update it for the 785 Andalusian municipalities using cadastral registers will be explored, as this register is updated monthly.

The year 1980 has been chosen as a milestone for housing modernization and building technology improvement. Other milestones could be equally or more relevant, and so it has been explored, particularly in the Spanish context. The requirements relating to the energy certification of buildings established in Directive 2002/91/ EC of the European Parliament and of the Council, were transposed in Royal Decree 47/2007, of January 19, through which a basic procedure for the certification of energy efficiency in new buildings was approved. Since then, developers are forced to include the energy label in the elements of information, promotion, offer and contracts of all new buildings from the beginning, trying to make housing purchase decisions better informed. Besides, public buildings or institutions that serve a relevant number of people must display the energy label prominently as an exemplary measure.



| | Current rating | Potential rating |
|---|---|---|
| Main Walls | ABCDEFG | ABCDEFG |
| Main Roof | ABCDEFG | ABCDEFG |
| Extension Walls | n/a | n/a |
| Extension Roof | n/a | n/a |
| Main Floor | ABCDEFG | ABCDEFG |
| Extension Floor | n/a | n/a |
| Windows | ABCDEFG | ABCDEFG |
| Main Heating | ABCDEFG | ABCDEFG |
| Secondary Heating | ABCDEFG | ABCDEFG |
| Hot Water | ABCDEFG | ABCDEFG |

*Figure 16*

Moreover in 2006 a new Technical Building Code was approved by Royal Decree[1], representing a milestone in relation to construction requirements and also those related to the energy efficiency of buildings. 2006 Technical Building Code modifications allowed for a two-year delay, it has been considered 2008 as major tipping point in terms of energy efficiency in the construction activities, applicable only for the Spanish market. Both years will be used as relevant dates and hence, 2 indicators will be calculated.

## 5.2 Input Data

The main source of information for this indicator are the CAT files from Cadastre. The CAT format reflects all the non-confidential alphanumeric information in the Cadastre databases. It includes information on urban, rustic and protected properties. Farms, building units, premises, orchards and real estate are represented.

Based on the methodology developed by IECA to identify private households/dwellings, Cadastre Table 14 (building registers) is the basis of the procedure. There is one of such registers for each building unit in a cadastral plot.

## 5.3 Methodology

1 Table 14 is completed by adding the amount of real estate that are meant to be dwellings (code "V") grouped by the cadastral reference.

2 The reference year (1980 or 2008) is saved for later use to compare with the effective year.

3 For single attached or single dettached dwellings:

  3.1 Query for any real estate including at least one building meant for dwelling and labelled according to the reference year.

  3.2 Filters applied:

   A construction cannot be

    A.a '0123' garages

    A.b '0522' '0512' swimming pools

    A.c '1037' '1036' '1038' reservoirs

   B Total aggregated surface of buildings in the range between 24 and 700 m.

4 For apartment dwellings with only one real estate unit and at least one construction meant for dwelling.:

  4.1 Query for each aggregation of constructions meant for dwelling (code 'V') per block, stairwell and door, and labelled according to the reference year.

  4.2 Filters applied

   A Construction must be '0111' or '0112'

   B Total aggregated surface of buildings in the range between 24 and 700 m.

5 For apartment dwellings with more than one real estate unit with at least one construction meant for dwelling each.:

  5.1 Query is the same as for non-collective: any real estate including at least one building meant for dwelling, and labelled according to the reference year.

[1] https://www.boe.es/eli/es/rd/2006/03/17/314/con/20060328

5.2 Filters applied
C Construction must be '0111' or '0112'
D Total aggregated surface of buildings in the range between 24 and 700 m.

We can now merge the tables resulting from steps 3, 4 and 5 and group per municipality. The procedure should be repeated for each year of reference.

## 5.4 Conclusions and further development

This study could be repeatable for any municipality in Spain from publicly available Cadastre information.

The main indicator (houses built after 1980) has also a European reference in the 2011 dataset *Share of new Buildings* (Baranzelli 2011). The confrontation of both datasets in the following table provides some insights on the intensity of building in the largest municipalities:

| | *Baranzelli 2011* | *IECA 2021* |
|---|---|---|
| *Algeciras* | 31,53% | 58,49% |
| *Almería* | 26,75% | 53,37% |
| *Cádiz* | 22,10% | 37,88% |
| *Córdoba* | 26,88% | 49,35% |
| *Dos Hermanas* | 33,06% | 72,40% |
| *Granada* | 29,16% | 43,83% |
| *Huelva* | 31,91% | 50,50% |
| *Jaén* | 28,66% | 52,18% |
| *Jerez de la Frontera* | 27,52% | 58,03% |
| *Málaga* | 30,65% | 46,96% |
| *Sevilla* | 26,92% | 42,77% |

# 6 Built area per capita

**Main SDG** **Linked to:**

11 SUSTAINABLE CITIES AND COMMUNITIES

15 LIFE ON LAND

## 6.1 Intro

This indicator captures the amount of built-up area per capita in each urban center (Florczyk 2019) and needs to be calculated using data reporting about land use. "Built-up" is defined by the presence of buildings (roofed structures). This definition largely excludes other parts of urban environments and the human footprint such as paved surfaces (roads, parking lots), commercial and industrial sites (ports, land- fills, quarries, runways) and urban green spaces (parks, gardens).

Consequently, such built-up area may be significantly different from other urban area data that use alternative definitions. The number of inhabitants is calculated within the boundaries of the area of interest (urban centers) using updated IECA population data.

There are two different approaches for his indicator. The first one is processing by means of code for the Cadastre table of subplots; the second one is the use of spatial analysis in QGIS starting from layers obtained from the INSPIRE services provided also by the Cadastre. We are choosing the second one as the analysis of the subplots is already embedded on it.

## 6.2 Input Data

The European INSPIRE Directive (Directive 2007/2 / CE, Infrastructure for Spatial Information in Europe) establishes the mandatory general rules for the establishment of a Spatial Information Infrastructure in the European Community based on the Infrastructure of the Member States. The transposition of this Directive into the Spanish legal system is developed through Law 14/2010, of July 5, on infrastructure and geographic information services in Spain (LISIGE).

To ensure that the spatial data infrastructures of the Member States are compatible and interoperable in a community and cross-border context, the Directive requires that specific Common Implementing Rules be adopted for data, metadata and services.

Among the geographical data that the Directive requires harmonization for are the Cadastral Parcels (CP, Cadastral Parcels), the Addresses (AD, Addresses) and the Buildings (BU, Buildings). For this reason, the Cadastre has generated a set of data in accordance with INSPIRE, transforming its data according to the mandatory standards established.

According to the INSPIRE specifications document for buildings, (INSPIRE Data Specification on Buildings - Technical Guidelines 3.0), a building is considered to be any permanent construction, over the surface or underground, with the purpose of housing people, animals and things, or the production and distribution of goods or services.

The representation of buildings in the Spanish cadastre is complex. In the cartography, the construction elements are drawn according to the built volumes indicated with Roman numerals. Thus, an enclosure labeled –I + II reflects that this part of the building has one floor below ground and two above ground.

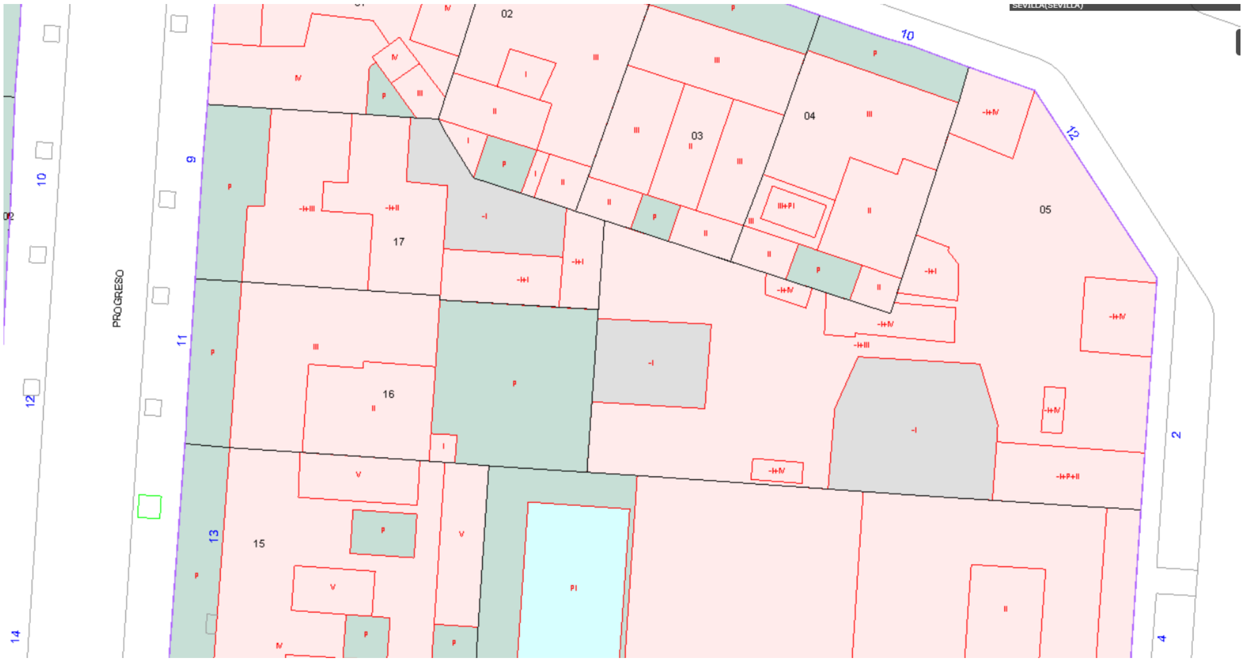
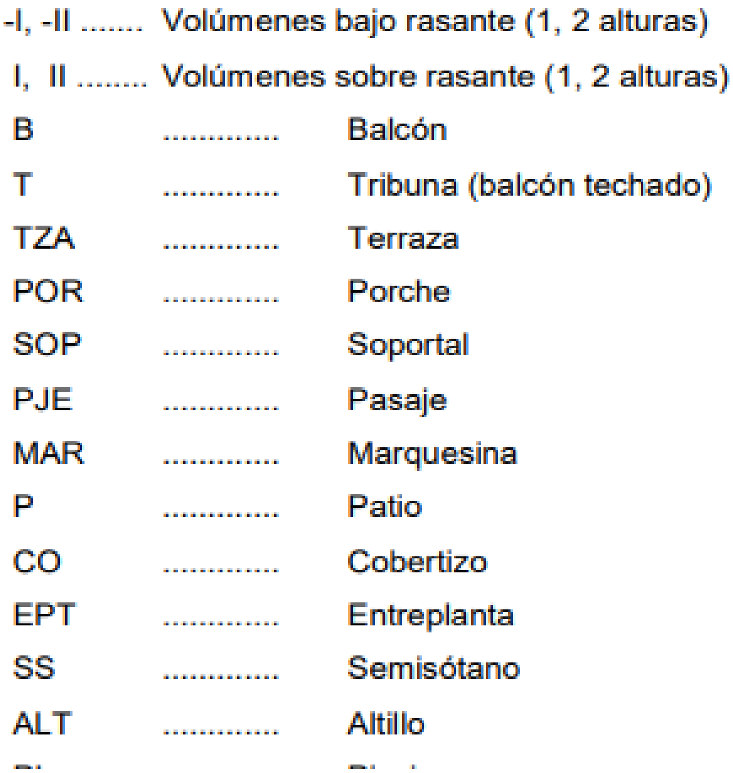


*Figure 17: Cadastral cartography showing the labelling of construction elements*



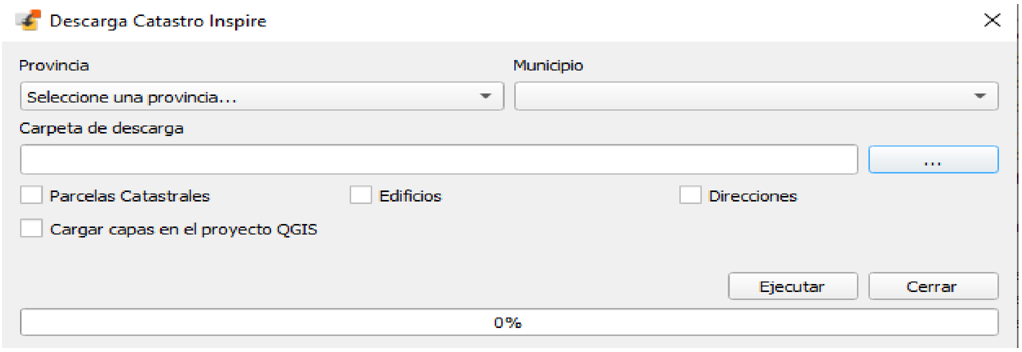
| | |
|---|---|
| -I, -II ....... | Volúmenes bajo rasante (1, 2 alturas) |
| I, II ........ | Volúmenes sobre rasante (1, 2 alturas) |
| B ............. | Balcón |
| T ............. | Tribuna (balcón techado) |
| TZA ............. | Terraza |
| POR ............. | Porche |
| SOP ............. | Soportal |
| PJE ............. | Pasaje |
| MAR ............. | Marquesina |
| P ............. | Patio |
| CO ............. | Cobertizo |
| EPT ............. | Entreplanta |
| SS ............. | Semisótano |
| ALT ............. | Altillo |

The building, as INSPIRE considers it, is not a graphic object that exists in the GIS of the Spanish cadastre and we have to build it from the union of the graphic objects of the layer that represents all the rooms with volumetry built above ground.

For this reason, the geometry of the building in the INSPIRE model is defined as a multi-enclosure that represents the enveloping line of all buildings with volumetry above ground of each cadastral parcel, excluding cantilevers and terraces or balconies.

## 6.3 Methodology

Data is obtained from the INSPIRE download service (GML-type layers, easily convertible to SHP) and therefore conforming to the European standards. The data has been downloaded using the QGIS INSPIRE Spanish Catastral Information Downloader Plugin.



Descarga Catastro Inspire

Provincia: Seleccione una provincia...

Municipio:

Carpeta de descarga:

□ Parcelas Catastrales □ Edificios □ Direcciones

□ Cargar capas en el proyecto QGIS

*Figure 18*

The layer of interest in this case is the one referring to Buildings, which consists only of the built-up area, from where the entire built-up area has been obtained. With this layer, a simple spatial sum of the area represented by the built area is made and divided by the total population.

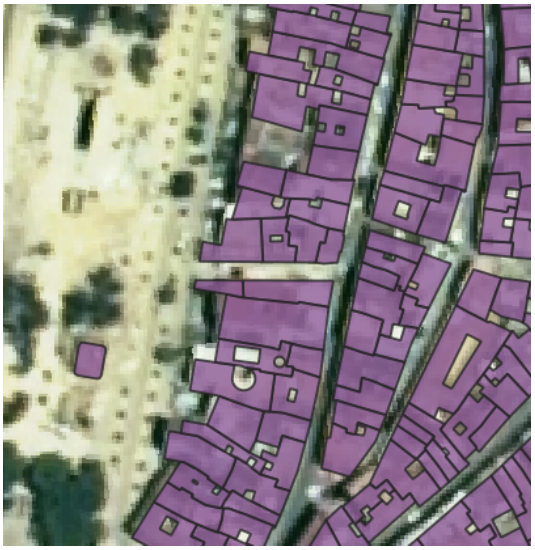


*Figure 19: Building layer from INSPIRE services*

A problem was detected in the Building layer from INSPIRE, where reservoirs attached to dams are mistakenly identified as buildings. A simple selection of the buildings with more than 1ha allowed us to identify these errors and remove them from the layer.

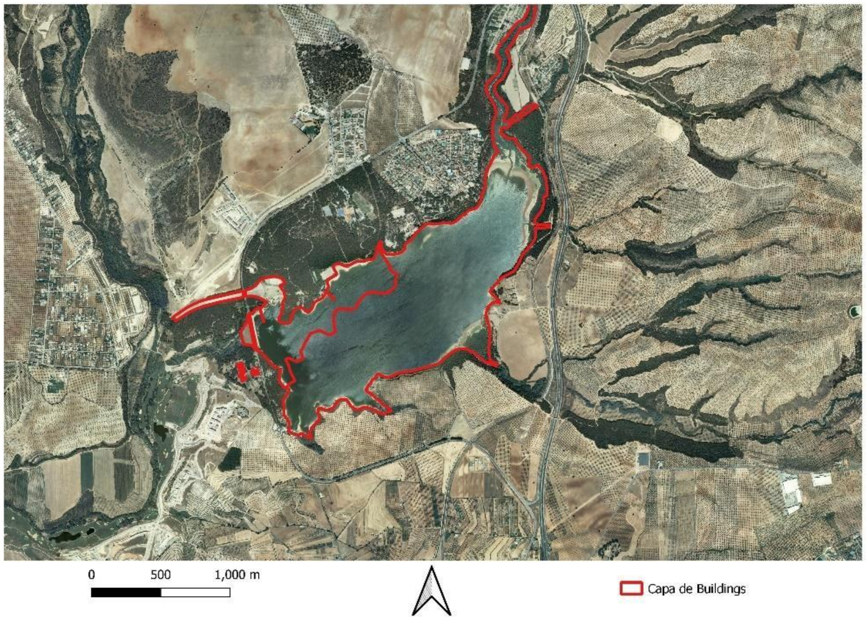


*Figure 20: Mistake in INSPIRE data, reservoir identified as building structure*

## 6.4 Conclusions and further development

This study could be repeatable for any municipality in Spain from publicly available Cadastre information.

The indicator could also be calculated using 2015 data from the *Global Human Settlement Layer Urban Centres Database* (GHS-UCDB) (Florczyk 2019). Some of the most populated Andalusian cities are present in this database and compared with our data in the following table:

| | *GHS-UCDB 2015* | *IECA 2021* |
|---|---|---|
| | *($m^2$ / capita)* | *($m^2$ / capita)* |
| *Algeciras* | 124,82 | 33,87 |
| *Almería* | 114,30 | 32,32 |
| *Cádiz* | 77,50 | 19,98 |
| *Córdoba* | 82,19 | 42,28 |
| *Dos Hermanas* | 152,57 | 50,07 |
| *Granada* | 68,12 | 27,17 |
| *Huelva* | 107,99 | 28,06 |
| *Jaén* | 93,84 | 40,73 |
| *Jerez de la Frontera* | 179,49 | 43,51 |
| *Málaga* | 108,13 | 33,96 |
| *Sevilla* | 86,29 | 25,31 |

# 7 Inputs for the statistical corporate scorecard and visualization tool.

In accordance with the IECA structure, these indicators could be included under the georeferenced data area in its portal. However, and using as a reference former projects visualizing SDG data, some widely-used SDG-centered approaches have been considered that might contribute to increasing users' commitment and the general knowledge of the 2030 Agenda, both geographically structured and chart oriented.

Open SDG is an open source, free-to-reuse platform for managing and publishing data and statistics related to the UN Sustainable Development Goals (SDGs). It is built exclusively with open-source libraries and tools and can be hosted and maintained using free services.

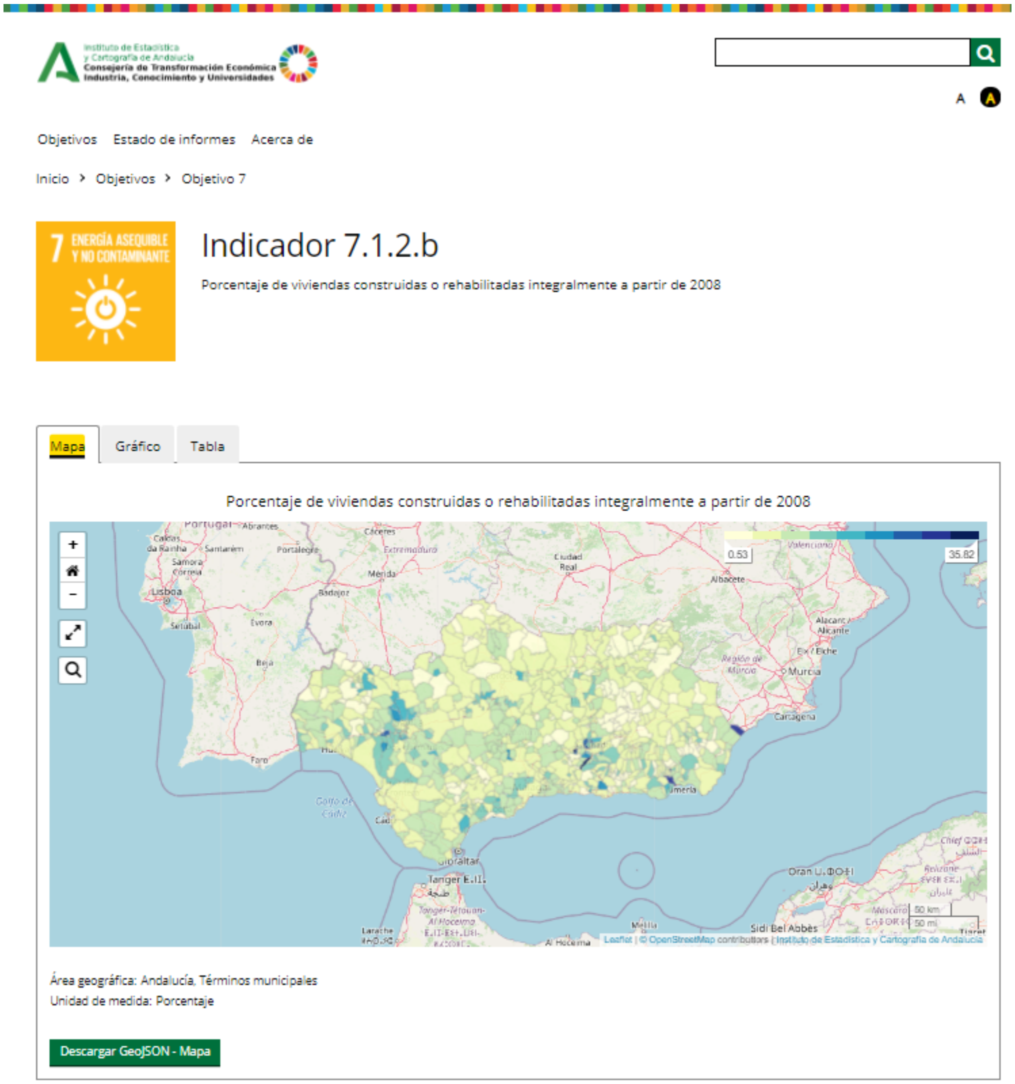
Open SDG is the result of collaboration between the UK Office for National Statistics (ONS), US government, the nonprofit Center for Open Data Enterprise (CODE) and members of the Open SDG community. It is already widely used, both by Spanish Administrations (ISTAC in the Canary Islands, SADEI in Asturias, ICANE in Cantabria) and abroad (DESTATIS in Germany, the Los Angeles City Hall…).

Open SDG platforms are very customizable and there are a variety of optional features which can be easily configured:

- Multi Language support
- Traffic Monitoring
- Site search
- Support for several types of charts
- Support for Maps: By default, data uploaded to an Open SDG platform is displayed on a chart and a table, but can also be configured to show geographic data.
- Embedded content: Another way of showing data/information on an indicator page is by embedding content from other websites/applications.
- Targets on goal pages
- Reporting status options
- Display data for national indicators as well as global indicators
- Filter by disaggregation: Open SDG platforms allow data to be displayed in a way in which it can be filtered by disaggregation. This allows the user to compare different breakdowns for a particular indicator.
- News, posts, and categories

OpenSDG has been chosen as the quick solution to rapidly publish the results of the research in a seamless way. Three of the resulting indicators (all but the public transit accessibility indicator) have been formatted according to the specifications required by Open SDG and the website is available online in a temporary website (https://ods-municipios-andalucia.github.io/ods-municipios-andalucia/).

In the short term, IECA might choose to upgrade its Open SDG site with further indicators.


Instituto de Estadística y Cartografía de Andalucía
Consejería de Transformación Económica Industria, Conocimiento y Universidades
Objetivos
Estado de informes
Acerca de
Inicio › Objetivos › Objetivo 7
7 ENERGÍA ASEQUIBLE Y NO CONTAMINANTE
Mapa
Gráfico
Tabla
0.53
35.82
Descargar GeoJSON - Mapa

Indicador 7.1.2.b

Porcentaje de viviendas construidas o rehabilitadas integralmente a partir de 2008

Porcentaje de viviendas construidas o rehabilitadas integralmente a partir de 2008

Área geográfica: Andalucía, Términos municipales
Unidad de medida: Porcentaje

*Figure 21: IECA Open SDG sample website*

In the mid and long terms, though, IECA is researching other options that might result in more customizable and insightful visualizations. In particular, D3.js visualization framework and visx (a suite of several low-level standalone packages for building visual interfaces with react) components are suitable for more engaging applications.

# 8 Contributions oriented to knowledge exchange and capacity-building

## 8.1 Workshop

In early september 2021, a workshop will be organized by the Andalusian Agency for International Development Cooperation (AACID), the Andalusian Institute for Statistics and Cartography (IECA) and the United Nations Development Program (UNDP) to disclose the results of the project and share the acquired knowledge with other Statistics Agencies, Researchers and Institutions related with Sustainable Development and Public Policies.

The format and scope of the workshop is still to be defined, but it is likely there will be two different approaches, one focused on high-level applicability and the second one with a technical and scientific focus.

## 8.2 Scientific Publications

Between November 15 and 19, 2021, the XXI edition of the JECAS (Statistics Conference of the Autonomous Communities) will be organized by the Canarian Institute of Statistics and take place entirely online, under the slogan Communities that count. Mobilizing data for sustainable development. The goal of the event is to address the role of public statistics in the Autonomous Communities as a fundamental basis for the analysis, evaluation and decision-making that improve the quality of life in our society, with special attention to the measurement of sustainable development.

Within this event, and as a result of this project, two different scientific communications have been admitted:

- Pgrouting for the calculation of accessibility indicators: This communication will present the methodology and software used to calculate the indicator relative to the percentage of the population that does not have green areas in their neighborhood, proposed in the JRC European Manual of Voluntary Local Reports. The indicator aims to measure the amount of population that can easily reach a green area in their neighborhood. The methodology used is based on that presented in the document "A walk to the park?" Assessing access to green areas in Europe's cities. Based on the information available in Andalucía on the georeferenced population, at the portal level, the communication proposes the calculation of service areas at 10 minutes and later their intersection with the layer of green areas available for Andalucía, thus identifying the green areas accessible to less 10-minute walk from each inhabited portal.
- Public Statistics on the follow-up and monitoring of the SDGs at the local level in Andalucía: This communication will present the results of the first milestone of the partnership of IECA, the Andalusian Agency for International Development Cooperation (AACID) and the United Nations Development Program (UNDP), with the aim of reinforcing the institutional role of Public Statistics in the follow-up and monitoring of the SDGs at the local level in Andalucía: the calculation and dissemination of three indicators selected from those proposed in the European Manual of Voluntary Local Reports for all municipalities in Andalucía.

# 9 References

1 Poelman H, (2018) *A Walk To The Park? Assessing Access To Green Areas In Europe's Cities Update Using Completed Copernicus Urban Atlas Data*, WP 01/2018, https://ec.europa.eu/regional_policy/en/information/publications/working-papers/2018/a-walk-to-the-park-assessing-access-to-green-areas-in-europe-s-cities

2 Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2001). "*Section 24.3: Dijkstra's algorithm". Introduction to Algorithms (Second ed.).* MIT Press and McGraw–Hill.

3 Copernicus Urban Atlas, https://land.copernicus.eu/local/urban-atlas

4 Datos Espaciales de Referencia de Andalucía (DERA), https://www.juntadeandalucia.es/institutodeestadisticaycartografia/DERA/

5 Dirección General del Catastro, http://www.catastro.minhap.es/documentos/preguntas_frecuentes_formato_cat.pdf

6 Conferencia Hidrográfica del Guadalquivir, 2020, *¿Qué es el Dominio Público Hidráulico*?, https://www.chguadalquivir.es/blogprensa/-/blogs/-que-es-el-dominio-publico-hidraulico-

7 Nestar, Noelia, 2015, Evaluación de soluciones para el cálculo de rutas óptimas con bases de datos, TFG Universidad de Valladolid, https://core.ac.uk/download/pdf/211098806.pdf

8 Poelman H, Dijkstra L, (2015) *A Measuring access to public transport in European cities* WP 01/2015, https://ec.europa.eu/regional_policy/sources/docgener/work/2015_01_publ_transp.pdf

9 General Transit Feed Specification (GTFS) https://developers.google.com/transit/gtfs/reference

10 Baranzelli, Claudia; Ronchi, Silvia (2011): *UDP - Share of new buildings, 2011 (JRC LUISA Reference Scenario 2016).* European Commission, Joint Research Centre (JRC). https://data.jrc.ec.europa.eu/dataset/jrc-luisa-udp-sharenewbuild-reference-2016

11 Florczyk A.J., Corbane C., Ehrlich D., Freire S., Kemper T., Maffenini L., Melchiorri M., Pesaresi M., Politis P., Schiavina M., Sabo F., Zanchetta L., (2019) *GHSL Data Package 2019*, JRC Publications Repository, https://publications.jrc.ec.europa.eu/repository/bitstream/JRC117104/ghsl_data_package_2019_1.pdf

12 MITECO, Certificación Energétca en Edificios , https://energia.gob.es/desarrollo/EficienciaEnergetica/CertificacionEnergetica/Paginas/certificacion.aspx

13 Dirección General del Catastro (2011), *Fichero informático de remisión de catastro (bienes inmuebles urbanos, rústicos y de características especiales)* http://www.catastro.minhap.es/documentos/formatos_intercambio/catastro_fin_cat_2006.pdf :

14 Dirección General del Catastro (2016), Conjunto de Datos INSPIRE, http://www.catastro.meh.es/webinspire/documentos/Conjuntos%20de%20datos.pdf

# 10 Annex 1: Green Areas Accessibility Code

## 10.1 Postgis Code: green areas layer creation

```
--Analizamos los campos leisure, landuse y access de Open Street Map

select leisure,count(*),sum(ST_Area(geom))/10000 AS hectareas

from zonas_verdes.seleccion_zonas_verdes

group by leisure;

select landuse,count(*),sum(ST_Area(geom))/10000 AS hectareas

from zonas_verdes.seleccion_zonas_verdes

group by landuse;

select access,count(*),sum(ST_Area(geom))/10000 AS hectareas

from zonas_verdes.seleccion_zonas_verdes

group by access;

-- Seleccionamos los polígonos que vamos a eliminar con campos leisure, landuse
y access que no nos interesa

create table zonas_verdes.seleccion_zonas_verdes_filtro as

select id,
geom,full_id,osm_id,osm_type,landuse,"natural",leisure,"name","access"

from zonas_verdes.seleccion_zonas_verdes

where (leisure in
('bandstand','fairground','golf_course','pitch','sports_centre','swimming_pool')

or landuse in
('allotments','basin','brownfield','commercial','construction','farmland','farmy
ard','industrial',

'meadow','orchard','quarry','reservoir','residential','traffic_island')

or access in ('customers','designated','no','private','prohibited'));

--Eliminamos los polígonos seleccionados anteriormente

create table zonas_verdes.seleccion_zonas_verdes_02 as

select id,
geom,full_id,osm_id,osm_type,landuse,"natural",leisure,"name","access"

from zonas_verdes.seleccion_zonas_verdes

where id not in (select id from zonas_verdes.seleccion_zonas_verdes_filtro);

--Comprobamos que se ha hecho correctamente
```

```
select leisure,count(*),sum(ST_Area(geom))/10000 AS hectareas
from zonas_verdes.seleccion_zonas_verdes_02
group by leisure;
select landuse,count(*),sum(ST_Area(geom))/10000 AS hectareas
from zonas_verdes.seleccion_zonas_verdes_02
group by landuse;
select access,count(*),sum(ST_Area(geom))/10000 AS hectareas
from zonas_verdes.seleccion_zonas_verdes_02
group by access;
--Realizamos un buffer de 4.5 metros sobre las zonas verdes
create table zonas_verdes.seleccion_zonas_verdes_03 as
select id, st_buffer(geom,+4.5) as geom,
full_id,osm_id,osm_type,landuse,"natural",leisure,"name"
from zonas_verdes.seleccion_zonas_verdes_02;
select count(*) from zonas_verdes.seleccion_zonas_verdes_03; --48.205
--Creamos un índice paa que vaya más rápido
create index seleccion_zonas_verdes_03g on
zonas_verdes.seleccion_zonas_verdes_03 using gist(geom);
--Unimos los polígono en uno sólo
create table zonas_verdes.seleccion_zonas_verdes_04 as
select ST_Union(geom) as geom from zonas_verdes.seleccion_zonas_verdes_03;
--Dividimos los polígonos
create table zonas_verdes.seleccion_zonas_verdes_05 as
select (ST_Dump(geom)).geom as geom from zonas_verdes.seleccion_zonas_verdes_04;

--Deshacemos el buffer de 4.5 metros
create table zonas_verdes.seleccion_zonas_verdes_06 as
select ST_buffer(geom,-4.5) as geom from zonas_verdes.seleccion_zonas_verdes_05;

--Calculamos el área
alter table zonas_verdes.seleccion_zonas_verdes_06 add hectareas double
precision;
```

```
UPDATE zonas_verdes.seleccion_zonas_verdes_06 SET hectareas=ST_AREA(geom)/10000;

--Eliminamos los polígonos de menos de 0.25 hectáreas
create table zonas_verdes.seleccion_zonas_verdes_def as
select *
from zonas_verdes.seleccion_zonas_verdes_06
where hectareas>=0.25;

--Añadimos id
ALTER TABLE zonas_verdes.seleccion_zonas_verdes_def ADD COLUMN id SERIAL;
ALTER TABLE zonas_verdes.seleccion_zonas_verdes_def ADD PRIMARY KEY (id);

--------------------------------------------------------------------------------
--Tratamos de depurar la capa a partir de Catastro. Esta parte del script se lanza sobre
--la BD de Catastro, tabla 46
--Para ello seleccionamos las parcelas cuyos todos los BIM son de titularidad no pública,
--es decir, no comiencen ni por P, ni por Q, ni por S. Pueden haber otras empresas públicas
--que no comiencen por P,Q,S. Es el caso de ESTADIO LA CARTUJA DE SEVILLA S.A. (nif a41770207)
--Además, que una parcela sea de titularidad privada no quiere decir que sea de acceso privado
--------------------------------------------------------------------------------


--Contamos el número de titularidades según si son públicas o privadas de cada parcela EN LA TABLA 46
create table temp_met.parcelas_privadas01 as
select deleg,cmuntrib_dgc,parcela,
COUNT(CASE WHEN substr(nif_titular,1,1) not in ('P','Q','S') and substr(nif_titular,1,1) is not null THEN 1 END) as priv,
COUNT(CASE WHEN substr(nif_titular,1,1) is null THEN 1 END) as nulo,
COUNT(CASE WHEN substr(nif_titular,1,1) in ('P','Q','S') THEN 1 END) as publ
from cat2019.r46_2019
group by deleg,cmuntrib_dgc,parcela;

--Filtramos por las parcelas CON TODOS LOS BIM de titularidad no pública,
anexándole la geometría
create table temp_met.parcelas_privadas as
select distinct a.*,b.refcat_d,b.geom
from temp_met.parcelas_privadas01 a
left join grafica.parcela_2019 b on
(a.deleg||a.cmuntrib_dgc||a.parcela=b.refcat_d)
where (priv>0 and publ=0);

--Creamos un índice para que vaya más rápido
create index parcelas_privadasg on temp_met.parcelas_privadas using gist(geom);
--------------------------------------------------------------------------------

--Realizamos la intersección entre temp_met.parcelas_privadas y
zonas_verdes.seleccion_zonas_verdes_def
CREATE TABLE zonas_verdes.intersection_zv_parcelas AS (SELECT zv.id as id_zv,
ST_INTERSECTION(zv.geom, parcelas.geom) AS geom
                      FROM temp_met.parcelas_privadas AS parcelas,
zonas_verdes.seleccion_zonas_verdes_def AS zv
WHERE ST_INTERSECTS(zv.geom, parcelas.geom)
);
---Query returned successfully in 11 min 24 secs.

--En algunos polígonos muy grandes compuestos por varias parcelas catastrales,
--se observa que las parcelas están separadas algunos metros, posiblemente
--separadas por pequeños senderos, ríos, etc.
--Vamos a realizar por tanto a la intersección un buffer de 4.5 metros,
create table zonas_verdes.intersection_zv_parcelas2 as
```

```
select id_zv,st_buffer(geom,+4.5) as geom
from zonas_verdes.intersection_zv_parcelas;

--Unimos los polígonos de la intersección agrupando por id de la zona verde
create table zonas_verdes.intersection_zv_parcelas3 as
select id_zv,ST_Union(geom) as geom from zonas_verdes.intersection_zv_parcelas2
group by id_zv;
---Query returned successfully in 38 secs 629 msec.

--Deshacemos el buffer
create table zonas_verdes.intersection_zv_parcelas4 as
select id_zv,st_buffer(geom,-4.5) as geom
from zonas_verdes.intersection_zv_parcelas3;

--Realizamos la diferencia entre las zonas verdes y la intersección zonas
verdes-parcelas privadas

--En primer lugar, creamos índices para ambas tablas
create index intersection_zv_parcelas4g on
zonas_verdes.intersection_zv_parcelas4 using gist(geom);
create index seleccion_zonas_verdes_defg on
zonas_verdes.seleccion_zonas_verdes_def using gist(geom);

--Realizamos la diferencia
CREATE TABLE zonas_verdes.seleccion_zonas_verdes_sinpriv AS
SELECT zv.id,ST_CollectionExtract(COALESCE(ST_Difference(zv.geom,
parcelas.geom), zv.geom),3) As geom
FROM zonas_verdes.seleccion_zonas_verdes_def AS zv
LEFT JOIN zonas_verdes.intersection_zv_parcelas4 AS parcelas ON
(zv.id=parcelas.id_zv);
---Query returned successfully in 17 secs 695 msec.
```

```
--Calculamos el área
alter table zonas_verdes.seleccion_zonas_verdes_sinpriv add hectareas double
precision;
UPDATE zonas_verdes.seleccion_zonas_verdes_sinpriv SET
hectareas=ST_AREA(geom)/10000;
--Eliminamos los polígonos de menos de 0.25 hectáreas
create table zonas_verdes.seleccion_zonas_verdes_sinpriv2 as
select *
from zonas_verdes.seleccion_zonas_verdes_sinpriv
where hectareas>=0.25;

--Comparamos el área de la zona verde antes de quitar las parcelas privadas y
después
create table zonas_verdes.comparativa_superficie as
select a.id,a.hectareas,b.hectareas as
hectareas_sinpriv,100*b.hectareas/a.hectareas as porc,
b.geom,
st_perimeter(b.geom) as perimeter_sinpriv,
st_perimeter(b.geom)/b.hectareas as cociente
from zonas_verdes.seleccion_zonas_verdes_def as a
left join zonas_verdes.seleccion_zonas_verdes_sinpriv2 as b on (a.id=b.id)
order by porc;

--Vamos a eliminar las de porcentaje de superficie inferior al 10% y
perímetro/área>1000
create table zonas_verdes.seleccion_zonas_verdes_sinpriv_def as
select id,hectareas_sinpriv,geom
from zonas_verdes.comparativa_superficie
where (porc>=10 or cociente<=1000);

--Vemos los que se han eliminado
create table zonas_verdes.superficies_raras_eliminadas as
```

```
select id,hectareas_sinpriv,geom

from zonas_verdes.comparativa_superficie

where ((porc<10 and cociente>1000) or porc is null or cociente is null);



--------------------------------------------------------------------------------

-- Recuentos de las tablas

--Zonas verdes

select count(*),sum(ST_Area(geom))/10000 AS hectareas from
zonas_verdes.seleccion_zonas_verdes_def;

--12.226, 1.594.346 hectáreas


--Parcelas privadas

select count(*),sum(ST_Area(geom))/10000 AS hectareas from
zonas_verdes.intersection_zv_parcelas4;

--6.909     659.808 hectáreas


--Zonas verdes menos parecelas privadas

select count(*),sum(ST_Area(geom))/10000 from
zonas_verdes.seleccion_zonas_verdes_sinpriv;

--12.226    934.538 hectáreas

select 1594346- 659808 --934538 (ok)


--Zonas verdes menos parecelas privadas de más de 0.25 ha.

select count(*),sum(ST_Area(geom))/10000 from
zonas_verdes.seleccion_zonas_verdes_sinpriv2;

--10.293    934.440 hectáreas


--Superficies extrañas eliminadas

select count(*),sum(ST_Area(geom))/10000 from
zonas_verdes.superficies_raras_eliminadas;

--2.709     2.209 hectáreas
```

```
--Capa definitiva sin superficies extrañas

select                  count(*),sum(ST_Area(geom))/10000                  from
zonas_verdes.seleccion_zonas_verdes_sinpriv_def;

--9.517     932.230 hectáreas

select 2209+ 932230 --934439 (ok)

select 9517+ 2709 --12.226 (ok)
```

## 10.2 Postgis Code: indicator calculation

```
--****ANÁLISIS DE RED ZONAS VERDES****

--Variables:

--     distancia para acercar los límites del parque al viario: 50m

--     distancia andando: 833m

--     intervalo entre puntos perimetrales de zonas verdes: 10m


-- 1) GENERACIÓN DE PUNTOS EN EL PERÍMETRO DE ZONAS VERDES

--Partimos de:

-- capa de Zonas verdes: polígonos o líneas "ZonasVerdes"

-- el proceso consiste en dividir las líneas de perímetro de las zonas verdes

-- en segmentos, y luego generar puntos en esos segmentos

--Resultado:

-- tabla "ZonasVerdes_puntos" (puede generarse en QGIS)

-- Si la generamos en QGIS, normalmente necesitamos regenerar los ids antes de
importarla

--
https://www.google.com/amp/s/anitagraser.com/2010/10/27/adding-a-unique-id-column-to
-layers-in-qgis/amp/


-- 2) AJUSTE DE LOS PORTALES AL VIARIO

--Partimos de:

-- capa de portales: Portales

--Resultado:

-- tabla PortalesAjustados (puede generarse en QGIS)


-- 3) CREACIÓN TOPOLOGÍA
```

```
--Partimos de:
-- capa de viario: Viario
-- generada como línea (forzamos en QGIS que no sea multilínea)
--Resultado: tablas viario_simple_noded y viario_simple_noded_vertices_pgr
-- tablas con los edges y nodos de la topología generada


-- Forzamos pasar a tipo de dato linea la geometria de nuestro viario recien
cargado, que tendra geometria multilinea
DROP TABLE IF EXISTS viario_simple;
DROP TABLE IF EXISTS viario_simple_vertices_pgr;
CREATE TABLE viario_simple AS (SELECT id, ST_LineMerge(geom) AS geom FROM
public."Viario");


-- Generamos la topología de nuestro viario tras cargar la capa, añadiendo antes dos
campos llamados source y target (se generara una capa de nodos)
ALTER TABLE viario_simple ADD COLUMN source INTEGER;
ALTER TABLE viario_simple ADD COLUMN target INTEGER;
SELECT pgr_createTopology('viario_simple', 0.0001, 'geom', 'id', 'source',
'target');


-- Analizamos cuantos nodos no se han generado mediante la funcion de analyzegraph,
será el número de intersecciones detectadas:
SELECT pgr_analyzeGraph('viario_simple', 0.0001, 'geom', 'id', 'source', 'target');


--Corregimos la topologia y se generara una nueva tabla con el viario corregido
(viario_simple_noded)
SELECT pgr_nodeNetwork('viario_simple', 0.0001, 'id', 'geom');


-- Una vez corregido el viario ejecutamos de nuevo la herramienta de createTopology
para la nueva tabla:
SELECT pgr_createTopology('viario_simple_noded', 0.0001, 'geom', 'id');


-- 4) PREPARACIÓN PUNTOS PARA pgRouting
--Para preparar la capa de puntos para ejecutar la herramienta de PGRouting
WithPoints y sus variantes.
```

```
--Partimos de:
--     Capa de viario con la topología ya preparada: viario_simple_noded, viario_simple_noded_vertices_pgr
--     Capa de perímetro de zonas verdes "ZonasVerdes_puntos":
--Resultado:
-- tabla fractions


--Por performance, vamos a limitar el número de puntos de zonas verdes
DROP TABLE IF EXISTS puntos_1;
CREATE TABLE puntos_1 AS
SELECT DISTINCT puntos.*
FROM public."PuntosVerdes" puntos, public."Viario" edges
WHERE ST_DWITHIN(puntos.geom, edges.geom, 0.1)
ORDER BY id ASC;


--Creamos una tabla llamada fractions con la entrada adecuada para la familia withpoints:
-- edge dentro de la distancia límite
-- fraction: valor de 0 a 1 que indica en qué punto del edge se sitúa (0 un extremo, 1 el otro)
-- guardamos para más adelante la distancia inicial (entre la situación original del punto y la nueva)
DROP TABLE IF EXISTS fractions;
CREATE TABLE fractions
(
    origin_id numeric,
    st_transform geometry,
    fraction double precision,
    initial_distance double precision,
    id_via bigint
);

DO $$
```

```
DECLARE
       points BIGINT;
       points_batch BIGINT := 10000;
       points_min BIGINT := 0;
       points_max BIGINT := 0;
       points_sup BIGINT := 0;
BEGIN
       SELECT COUNT(*) INTO points FROM puntos_1;
       SELECT MIN(id) INTO points_min FROM puntos_1;
       points_max := points_min + points;
       RAISE NOTICE 'Se van a analizar % puntos', points;

LOOP
       EXIT WHEN points_min > points_max;
       points_sup := points_min + points_batch;

       DROP TABLE IF EXISTS fractions_1;
       CREATE TABLE fractions_1 AS
       SELECT
              puntos_1.id AS origin_id,
              ST_Transform(puntos_1.geom, 25830),
              ST_LineLocatePoint(viario_simple_noded.geom,
ST_Transform(puntos_1.geom, 25830)) AS fraction,
              ST_Distance(viario_simple_noded.geom, ST_Transform(puntos_1.geom,
25830)) as initial_distance,
              viario_simple_noded.id AS id_via
       FROM
              puntos_1, viario_simple_noded
       WHERE
              ST_distance(viario_simple_noded.geom, ST_Transform(puntos_1.geom,
25830)) < 0.1
              AND puntos_1.id > points_min
              AND puntos_1.id <= points_sup;
```

```
    --Vamos añadiendo a una tabla
    DROP TABLE IF EXISTS fractions_parcial;
    CREATE TABLE fractions_parcial AS SELECT * FROM fractions;

    DROP TABLE IF EXISTS fractions_total;
    CREATE TABLE fractions_total AS
    SELECT * FROM fractions_parcial
    UNION
    SELECT * FROM fractions_1;

    DROP TABLE IF EXISTS fractions;
    CREATE TABLE fractions AS SELECT * FROM fractions_total;

    points_min := points_sup;
    RAISE NOTICE 'Puntos completados %', points_sup;

END LOOP;
END; $$

ALTER TABLE fractions
ADD COLUMN id SERIAL PRIMARY KEY;

--reducir fractions agrupando combinaciones de edge + fraction
DROP TABLE IF EXISTS fractions_2;
CREATE TABLE fractions_2 AS (
    SELECT id_via, fraction, min(initial_distance)
    FROM fractions
    GROUP BY id_via, fraction);
ALTER TABLE fractions_2
    ADD COLUMN id SERIAL PRIMARY KEY;
```

```
-- 5) ANÁLISIS DE RED CON PGROUTING
--Partimos de:
-- Tabla viario_simple_noded, viario_simple_noded_vertices_pgr
-- Tabla fractions

DROP TABLE IF EXISTS network_edges;
CREATE TABLE network_edges
(
    id bigint,
    geom geometry(LineString,25830)
);

DROP TABLE IF EXISTS network_nodes;
CREATE TABLE network_nodes
(
    id bigint,
    the_geom geometry(Point,25830),
    min double precision
);

DO $$
DECLARE
       fractions BIGINT;
       fractions_batch BIGINT := 10000;
       fractions_min BIGINT = 0;
       fractions_sup BIGINT;
BEGIN
       SELECT COUNT(*) INTO fractions FROM fractions_2;
       RAISE NOTICE 'Se van a analizar % fracciones', fractions;

LOOP
       EXIT WHEN fractions_min > fractions;
```

```
        fractions_sup := fractions_min + fractions_batch;


        DROP TABLE IF EXISTS fractions_3;
        CREATE TABLE fractions_3 AS
        SELECT *
        FROM fractions_2
        WHERE id > fractions_min
        AND id <= fractions_sup
        ORDER BY id ASC;


        --Llamada a la función pgr_withpointsdd
        --pgr_withpointsdd: Modify the graph to include points and using Dijkstra
algorithm,
        --extracts all the nodes and points that have costs less than or equal to the
value distance from the starting point.
        --The edges extracted will conform the corresponding spanning tree.
        --Parámetros:
        --      edges_sql: consulta que proporciona el viario (extraído de la
topología)
        --      points_sql: consulta que proporciona los puntos y vértices de los que
saldrán los puntos de partida
        --      start_vid: array de los puntos y/o vértices que se van a analizar.
Según el signo se trata de unos u otros (se pueden combinar)
        --              x: vértices (nodos). No los estamos utilizando
        --              -x: puntos (combinación de edge + fraction)
        --      distancia
        --      directed: TRUE, los edges pueden recorrerse en ambos sentidos
        --      details: TRUE, el resultado incluye los puntos intermedios
        DROP TABLE IF EXISTS network;
        CREATE TABLE network AS (
                SELECT * FROM pgr_withpointsdd(
                        'SELECT id, source, target, st_length(geom) as
cost,-st_length(geom) as reverse_cost FROM viario_simple_noded',
                        'SELECT id AS pid, id_via AS edge_id, fraction from fractions_3',
                        (SELECT array_agg(-fractions_3.id) FROM fractions_3),
                        833,
                        directed => false,
                        details => true,
                        equicost => false
                )
        );

        --edges que han sido alcanzados
        --agrupo para que no estén repetidos (cada edge puede alcanzarse varias veces, por distintos caminos)
        --esto optimizará las próximas queries
        DROP TABLE IF EXISTS network_edges_1;
        CREATE TABLE network_edges_1 AS (
                SELECT viario_simple_noded.id, viario_simple_noded.geom
                FROM network
                LEFT JOIN viario_simple_noded
                ON network.edge = viario_simple_noded.id
                GROUP BY viario_simple_noded.id, viario_simple_noded.geom
        );

        --nodos que han sido alcanzados
        --agrupo para que no estén repetidos (optimizará las próximas queries)
        --y guardo la distancia mínima para llegar a cada uno
        DROP TABLE IF EXISTS network_nodes_1;
        CREATE TABLE network_nodes_1 AS (
                SELECT viario_simple_noded_vertices_pgr.id, viario_simple_noded_vertices_pgr.the_geom, MIN(network.agg_cost)
                FROM network
                LEFT JOIN viario_simple_noded_vertices_pgr
                ON network.node = viario_simple_noded_vertices_pgr.id
```

```
                GROUP BY viario_simple_noded_vertices_pgr.id,
viario_simple_noded_vertices_pgr.the_geom
        );

        --Vamos añadiendo a dos tablas incrementales
        DROP TABLE IF EXISTS network_edges_parcial;--edges
        CREATE TABLE network_edges_parcial AS SELECT * FROM network_edges;
        DROP TABLE IF EXISTS network_nodes_parcial;--nodes
        CREATE TABLE network_nodes_parcial AS SELECT * FROM network_nodes;

        DROP TABLE IF EXISTS network_edges_total;--edges
        CREATE TABLE network_edges_total AS
        SELECT * FROM network_edges_parcial
        UNION
        SELECT * FROM network_edges_1;
        DROP TABLE IF EXISTS network_nodes_total;--nodes
        CREATE TABLE network_nodes_total AS
        SELECT * FROM network_nodes_parcial
        UNION
        SELECT * FROM network_nodes_1;

        DROP TABLE IF EXISTS network_edges;--edges
        CREATE TABLE network_edges AS SELECT * FROM network_edges_total;
        DROP TABLE IF EXISTS network_nodes;--nodes
        CREATE TABLE network_nodes AS SELECT * FROM network_nodes_total;

        fractions_min := fractions_sup;
        RAISE NOTICE 'Áreas calculadas %', fractions_sup;

END LOOP;
END; $$
```

```
--Podemos volver a reducir el número de nodes quedándonos la distancia mínima
DROP TABLE IF EXISTS network_nodes_total_1;
CREATE TABLE network_nodes_total_1 AS (
        SELECT id, the_geom, MIN(min)
        FROM network_nodes_total
        GROUP BY id, the_geom
);

-- 6) BÚSQUEDA DEL RESTO DE EDGES
--pgRouting no utiliza todos los edges para alcanzar los nodos
--buscamos esos edges que no han sido necesarios para alcanzar los nodos

--primero buscamos los edges que no se han incluido pero cuyos dos nodos extremos si
están, junto con su longitud
DROP TABLE IF EXISTS missingnetwork_1;
CREATE TABLE missingnetwork_1 AS (
        SELECT *, st_length(geom) as distancia
        FROM viario_simple_noded
        WHERE
                id NOT IN (SELECT id FROM network_edges_total WHERE id IS NOT null) AND
                source IN (SELECT id FROM network_nodes_total_1 WHERE id IS NOT null)
AND
                target IN (SELECT id FROM network_nodes_total_1 WHERE id IS NOT null)
);

--después hallamos la distancia mínima necesaria para llegar a esos nodos
DROP TABLE IF EXISTS missingnetwork_2;
CREATE TABLE missingnetwork_2 AS (
        SELECT missingnetwork_1.*, sources.min AS min_source, targets.min AS
min_target
        FROM missingnetwork_1
        LEFT JOIN network_nodes_total_1 sources
                ON missingnetwork_1.source = sources.id
```

```
       LEFT JOIN network_nodes_total_1 targets
              ON missingnetwork_1.target = targets.id
);
--por último, comprobamos que la longitud del edge sea menor que lo que quedaba hasta la distancia límite desde cualquiera de los dos nodos
DROP TABLE IF EXISTS missingnetwork_3;
CREATE TABLE missingnetwork_3 AS (
       SELECT id, geom
       FROM missingnetwork_2
       WHERE
              min_source + distancia < 833 OR
              min_target + distancia < 833
);
-- 7) ÁREA DE SERVICIO EN VIARIO
--Primero encontramos la unión de ambos resultados (procedente de pgrouting y de la búsqueda de edgen no usados)
DROP TABLE IF EXISTS area;
CREATE TABLE area as (
       SELECT * FROM network_edges_total
       UNION
       SELECT * FROM missingnetwork_3
);

-- 8) NODOS SOBRE LOS QUE HAY QUE HACER LA ÚLTIMA BÚSQUEDA
--Buscamos los últimos portales:
--     Están en un edge en el que al menos uno de los nodos extremos se ha alcanzado
--     Intersecan con un edge que sale de ese nodo
--     Están dentro de un buffer con la distancia necesaria para alcanzar ese nodo

--Edges no alcanzados pero donde, al menos, uno de los nodos sí lo ha sido
DROP TABLE IF EXISTS missingnetwork_4;
CREATE TABLE missingnetwork_4 AS (
       SELECT *, st_length(geom) as distancia
```

```
        FROM viario_simple_noded
        WHERE
                id NOT IN (SELECT id FROM area WHERE id IS NOT null) AND
                (source IN (SELECT id FROM network_nodes_total_1 WHERE id IS NOT null)
OR
                target IN (SELECT id FROM network_nodes_total_1 WHERE id IS NOT null))
);

--Nodos alcanzados de estos edges, con su geometría y su distancia
DROP TABLE IF EXISTS missingnetwork_5;
CREATE TABLE missingnetwork_5 AS (
        SELECT un.id, un.id_edge, un.geom_edge, network_nodes_total_1.min,
network_nodes_total_1.the_geom AS geom_node
        FROM (
                SELECT source AS id, id AS id_edge, geom as geom_edge
                FROM missingnetwork_4
                UNION
                SELECT target AS id, id AS id_edge, geom as geom_edge
                FROM missingnetwork_4) un
        LEFT JOIN network_nodes_total_1
        ON
                un.id = network_nodes_total_1.id
        WHERE un.id IN (SELECT id FROM network_nodes_total_1 WHERE id IS NOT null)
);

-- 9) INTERSECCIÓN CON PORTALES
--9.1)Portales dentro de un buffer mínimo del viario total alcanzado
DROP TABLE IF EXISTS XXXXXXX.PortalesArea;
CREATE TABLE XXXXXXX.PortalesArea as (
        SELECT DISTINCT portales.id, portales.geom
        FROM XXXXXXX."PortalesAjustados" portales, XXXXXXX.area edges
        WHERE ST_DWITHIN(portales.geom, edges.geom, 0.1)
);
```

```
--9.2)Código en Python para buscar los portales a partir de los nodos extremos.
SELECT * FROM XXXXXXX.missingnetwork_6

-- 10) CÁLCULO DEL INDICADOR
--missingnetwork_6 creada desde Python
drop table if exists XXXXXXX.missingnetwork_7;
create table XXXXXXX.missingnetwork_7 as
select distinct XXXXXXX."PortalesAjustados".id, XXXXXXX."PortalesAjustados".geom
from XXXXXXX.missingnetwork_6
join XXXXXXX."PortalesAjustados"
on XXXXXXX.missingnetwork_6.id = XXXXXXX."PortalesAjustados".id;

--Sumamos los portales de missingnetwork_7 con los de portalesarea
--A esta unión, que son los portales que tienen acceso a zona verde, le anexamos la
población y el municipio
--Obtenemos para cada municipio la poblacióon con acceso a zonas verdes
create table XXXXXXX.portales_conacceso as
select id from XXXXXXX.portalesarea
union
select id from XXXXXXX.missingnetwork_7;

select id,count(id)
from XXXXXXX.portales_conacceso
group by id
order by count desc;
--having count(*)>1;

create table XXXXXXX.pob_muni_concacceso as
select cumun,nmun,
sum(pob_tot) as pob_tot,
sum(pob_h) as pob_h,
sum(pob_m) as pob_m,
sum(edad0015) as edad0015,
sum(edad1664) as edad1664,
sum(edad65_) as edad65_
from (select d.pob_tot,
d.pob_h,
d.pob_m,
d.edad0015,
d.edad1664,
d.edad65_,
e.cumun,
e.nmun
from XXXXXXX.portales_conacceso a
left join XXXXXXX."PortalesAjustados" b on (a.id=b.id)
left join gridp.nedif19_met c on (b.nedificio=c.nedificio)
left join gridp.ind19 d on c.nedificio=d.nedificio
left join dera.da08_mun2019 e on (ST_Contains(e.geom,c.geom)))t1
group by cumun,nmun
order by cumun;

--Total poblacion
/*
create table zonas_verdes.pob_muni as
select cumun,nmun,
sum(pob_tot) as pob_tot,
sum(pob_h) as pob_h,
sum(pob_m) as pob_m,
sum(edad0015) as edad0015,
sum(edad1664) as edad1664,
sum(edad65_) as edad65_
from (select a.pob_tot,
a.pob_h,
a.pob_m,
a.edad0015,
a.edad1664,
a.edad65_,
c.cumun,
c.nmun
from gridp.ind19 a
left join gridp.nedif19_met b on (a.nedificio=b.nedificio)
left join dera.da08_mun2019 c on (ST_Contains(c.geom,b.geom)))t1
group by cumun,nmun
order by cumun;
*/
select sum(pob_tot) from XXXXXXX.pob_muni;--8465219

--Cruzamos ambas tablas
create table XXXXXXX.indicador_cadiz as
select a.*,
b.pob_tot as pob_tot_ac,
b.pob_h as pob_h_ac,
b.pob_m as pob_m_ac,
b.edad0015 as edad0015_ac,
b.edad1664 as edad1664_ac,
b.edad65_ as edad65_ac
from XXXXXXX.pob_muni a
left join XXXXXXX.pob_muni_concacceso b on (a.cumun=b.cumun)
where substr(a.cumun,1,2)='11';

--Actualizamos a 0 los nulos
UPDATE XXXXXXX.indicador_cadiz
SET pob_tot_ac=coalesce(pob_tot_ac,0),
pob_h_ac=coalesce(pob_h_ac,0),
pob_m_ac=coalesce(pob_m_ac,0),
edad0015_ac=coalesce(edad0015_ac,0),
edad1664_ac=coalesce(edad1664_ac,0),

edad65_ac=coalesce(edad65_ac,0)

;


--Calculamos el indicador: (poblacion total - población con acceso a zona verde / 
población total)

ALTER TABLE XXXXXXX.indicador_cadiz

ADD porc_sinac decimal,

ADD porc_sinac_h double precision,

ADD porc_sinac_m double precision,

ADD porc_sinac_edad0015 double precision,

ADD porc_sinac_edad1664 double precision,

ADD porc_sinac_edad65 double precision;


UPDATE XXXXXXX.indicador_cadiz

SET porc_sinac=100*(cast(pob_tot as float)-cast(pob_tot_ac as float))/cast(pob_tot 
as float),

porc_sinac_h=100*(cast(pob_h as float)-cast(pob_h_ac as float))/cast(pob_h as 
float),

porc_sinac_m=100*(cast(pob_m as float)-cast(pob_m_ac as float))/cast(pob_m as 
float),

porc_sinac_edad0015=100*(cast(edad0015 as float)-cast(edad0015_ac as 
float))/cast(edad0015 as float),

porc_sinac_edad1664=100*(cast(edad1664 as float)-cast(edad1664_ac as 
float))/cast(edad1664 as float),

porc_sinac_edad65=100*(cast(edad65_ as float)-cast(edad65_ac as float))/cast(edad65_
as float)

;
```

## 10.3 Python Code

```
import psycopg2;

conn = psycopg2.connect("dbname=xxxx user=xxxx password=xxxx? host=xxxx")

# Open a cursor to perform database operations

cur = conn.cursor()
```

```
# Execute a command: this creates a new table
cur.execute("SELECT ST_AsText(geom_node), min, id FROM
zonasverdes_cadiz.missingnetwork_5;")
missingnodes = cur.fetchall();
newportals = 0;
cur.execute("DROP TABLE IF EXISTS zonasverdes_cadiz.missingnetwork_6;")
cur.execute("CREATE TABLE zonasverdes_cadiz.missingnetwork_6 (id INT);")
for row in missingnodes:
    geom = row[0];
    distance = str(833 - row[1]);
    query = "SELECT DISTINCT id FROM zonas_verdes.\"PortalesAjustados\" WHERE
ST_DWITHIN(geom, ST_GeomFromText('" + geom + "', 25830), " + distance + ") AND id
NOT IN (SELECT id FROM zonasverdes_cadiz.portalesarea WHERE id IS NOT null);"
    cur.execute(query);
    newpoints = cur.fetchall();
    newportals += len(newpoints);

    for point in newpoints:
        query = "INSERT INTO zonasverdes_cadiz.missingnetwork_6 (id) VALUES (" +
str(point[0]) + ");";
        cur.execute(query);

print("inserted " + str(newportals));
conn.commit();
cur.close();
conn.close();
```

## 10.4

# 11 Annex 2: Transport Accessibility Code

```
--*************INDICADOR ACCESIBILIDAD TRANSPORTE******************
--Datos Google Transit:
--http://datosabiertos.malaga.eu/recursos/transporte/EMT/lineasYHorarios/google_tran
sit.zip
--https://developers.google.com/transit/gtfs/reference


--los ficheros csv de Open Data Málaga están modificados, por problemas de formato
en los retornos de carro.
--se han abierto con Excel y se han guardado, con separador ";"


--TODO: importación directa


-- 1) IMPORTACIÓN TABLAS
--Importación tabla paradas
--https://datosabiertos.malaga.eu/dataset/lineas-y-horarios-bus-paradas
DROP TABLE IF EXISTS ma_stops;
CREATE TABLE ma_stops (
	stop_id INTEGER,
	stop_code INTEGER,
	stop_name VARCHAR(255),
	stop_lat REAL,
	stop_lon REAL,
	PRIMARY KEY (stop_id)
);


COPY ma_stops(stop_id, stop_code, stop_name, stop_lat, stop_lon)
FROM 'C:\Users\Usuario\Documents\IECA\MunicipioMalaga\stops2.csv'
DELIMITER ';'
CSV HEADER;


--Importación tabla servicios
```

```
--Contiene la hora a la que pasa cada servicio por cada parada
--Sustituímos 24: por 00:
--TODO: en principio, parece que son únicos, pero hay algunas duplicidades
(servicios que empiezan por 3)
--https://datosabiertos.malaga.eu/dataset/lineas-y-horarios-bus-horario
DROP TABLE IF EXISTS ma_stoptimes;
CREATE TABLE ma_stoptimes (
       trip_id VARCHAR(10),
       arrival_time TIME,
       departure_time TIME,
       stop_id INTEGER,
       stop_sequence INTEGER
);

COPY ma_stoptimes(trip_id,arrival_time, departure_time, stop_id,stop_sequence)
FROM 'C:\Users\Usuario\Documents\IECA\MunicipioMalaga\stop_times2.csv'
DELIMITER ';'
CSV HEADER;

--2) PROCESO DE GENERACIÓN  DE FRECUENCIAS POR PARADA
--Creación served_stops
--Creamos una tabla con la cantidad de veces que pasa un autobús por cada parada
--Tomamos sólo las salidas entre las 6:00 y las 20:00
--https://ec.europa.eu/regional_policy/sources/docgener/work/2015_01_publ_transp.pdf
DROP TABLE IF EXISTS ma_served_stops;
CREATE TABLE ma_served_stops AS (
       SELECT stop_id, COUNT(departure_time) AS services
       FROM ma_stoptimes
       WHERE (departure_time >= '06:00:00') AND (departure_time <= '20:00:00')
       GROUP BY stop_id);

--añadimos una columna frecuencia con la cantidad media de salidas por hora
```

```
-- (14 horas)
ALTER TABLE ma_served_stops
ADD COLUMN frequency REAL;

UPDATE ma_served_stops
SET frequency = services/14;

--Creación ma_stops_frequency_class
--Creamos una tabla con la información de cada parada, incluyendo la frecuencia de
autobuses
-- y su clase de frecuencia (0-3):
--     3: > 10 servicios por hora
--     2: 4-10 servicios por hora
--     1: < 3
CREATE TABLE ma_stops_frequency_class AS (
       SELECT ma_stops.*, ma_served_stops.frequency,
             CASE WHEN ma_served_stops.frequency > 10 THEN 3
                     WHEN (ma_served_stops.frequency <= 10) AND
(ma_served_stops.frequency >= 4) THEN 2
                     WHEN ma_served_stops.frequency < 4 THEN 1
                     ELSE 0 END
                     AS frequency_class
       FROM ma_served_stops
       RIGHT JOIN ma_stops
       ON ma_served_stops.stop_id = ma_stops.stop_id
);

--Número de paradas por clase de frecuencia
SELECT frequency_class, count (stop_id)
FROM ma_stops_frequency_class
GROUP BY frequency_class
ORDER BY frequency_class;
```

# 12 Annex 3: New Buildings Code

```
--En primer lugar, añadimos a la tabla 14 el número de bienes inmuebles con destino
--vivienda agrupando por refcat_d


ALTER TABLE cat2019.r14_2019
ADD COLUMN n_bi_viv numeric;


UPDATE cat2019.r14_2019
   SET  n_bi_viv= t.count
   from (SELECT count (distinct nordenbifis), refcat_d
  FROM cat2019.r14_2019 where trim (destino_dgc)='V' group by refcat_d ) t
 WHERE r14_2019.refcat_d=t.refcat_d;


--Definimos la variable anno, para realizar el recuento
--de las viviendas construidas antes de anno y después


DO $$
DECLARE anno integer;
BEGIN


SELECT 1980 INTO anno;


---------------------------------------------------------------------------------
--La información se elabora a partir de la tabla 14 (construcciones).
--Se hace un tratamiento diferenciado para las viviendas colectivas.
---------------------------------------------------------------------------------
--A) Todas las viviendas excepto las colectivas
---------------------------------------------------
--Entendemos como vivienda cualquier bien inmueble con alguna construcción con destino vivienda
```

```
--Filtros:

--Los tipos de construcción no pueden ser piscinas, garajes...

--Además, la agregación de construcciones debe estar entre 24 y 700 metros


create table temp_met.proceso4_nocolectivas as

select cpro_ine,cmuntrib_ine,

COUNT(CASE WHEN CAST(a_ant_bim AS INT) >= anno THEN 1 END) as nuevos,

COUNT(CASE WHEN CAST(a_ant_bim AS INT) < anno THEN 1 END) as antiguos,

COUNT(CASE WHEN a_ant_bim is null THEN 1 END) as nulo,

COUNT(*) as total

from (

SELECT sum(cast(t1.stotalocal as numeric)) as stotalocal,
t2.cpro_ine,t2.cmuntrib_ine,t2.a_ant_bim, COUNT (*) n_constru, t1.deleg,
t1.cmuntrib_dgc, t1.parcela, t1.nordenbifis

from cat2019.r14_2019 t1 left join

      cat2019.r15_2019 t2 on
(t1.deleg||t1.cmuntrib_dgc||t1.parcela||t1.nordenbifis=t2.deleg||t2.cmuntrib_dgc
||t2.parcela||t2.nsecubi)

      where trim(t1.destino_dgc)='V' and t1.nordenbifis is not null and
substring (t1.tipoconstru,1,3)<>'011'

              AND t1.tip_const3d<>'011' AND t1.tip_const4d<>'0123' AND
t1.tip_const4d<>'0522' AND t1.tip_const4d<>'0512'

              AND t1.tip_const4d<>'1037' AND t1.tip_const4d<>'1036' AND
t1.tip_const4d<>'1038'

      group by t1.deleg, t1.cmuntrib_dgc, t1.parcela,
t1.nordenbifis,t2.a_ant_bim,t2.cpro_ine,t2.cmuntrib_ine

) a

where (stotalocal <=700 and stotalocal >=24)

group by cpro_ine,cmuntrib_ine;


---------------------------------------------------

--B) Viviendas colectivas

--------------------------------------------------
```

```
--B1) Vivienda colectiva en aquellas parcelas con un solo bien inmueble con al
menos
--una construcción con destino residencial. En este caso se contabiliza como una
--vivienda cada agregación de construcciones con destino V por bloque, escalera,
planta y puerta
--Filtros:
--Los tipos de construcción sólo pueden ser 0111 o 0112 (colectiva edificación
abierta o en manzana cerrada)
--Además, la agregación de construcciones debe estar entre 24 y 700 metros
--------------------------------------------------


create table temp_met.proceso4_colectivas_unicas as
select cpro_ine,cmuntrib_ine,
COUNT(CASE WHEN CAST(a_ant_bim AS INT) >= anno THEN 1 END) as nuevos,
COUNT(CASE WHEN CAST(a_ant_bim AS INT) < anno THEN 1 END) as antiguos,
COUNT(CASE WHEN a_ant_bim is null THEN 1 END) as nulo,
COUNT(*) as total
from (
SELECT sum(cast(t1.stotalocal as numeric)) as
stotalocal,t2.cpro_ine,t2.cmuntrib_ine,t2.a_ant_bim, t1.deleg, t1.cmuntrib_dgc,
t1.parcela, t1.nordenbifis, t1.bloque, t1.escalera, t1.planta, t1.puerta
FROM cat2019.r14_2019 t1 left join
      cat2019.r15_2019 t2 on
(t1.deleg||t1.cmuntrib_dgc||t1.parcela||t1.nordenbifis=t2.deleg||t2.cmuntrib_dgc
||t2.parcela||t2.nsecubi)
      where trim(t1.destino_dgc)='V' and t1.nordenbifis is not null and
substring (t1.tipoconstru,1,3)='011' AND t1.n_bi_viv=1
      AND (t1.tip_const4d='0111' or t1.tip_const4d='0112')
      GROUP BY t1.deleg, t1.cmuntrib_dgc, t1.parcela,t1.nordenbifis, t1.bloque,
t1.escalera, t1.planta, t1.puerta,t2.a_ant_bim,
      t2.cpro_ine,t2.cmuntrib_ine
) a
where (stotalocal <=700 and stotalocal >=24)
group by cpro_ine,cmuntrib_ine;
```

```
--------------------------------------------------
--B2) Vivienda colectiva en aquellas parcelas con más de un bien inmueble con al
--menos una construcción con destino residencial. En este caso es igual que para
las no colectivas.
--La vivienda se contabiliza por bien inmueble
--Filtros:
--Los tipos de construcción sólo pueden ser 0111 o 0112 (colectiva edificación
abierta o en manzana cerrada)
--Además, la agregación de construcciones debe estar entre 24 y 700 metros
--------------------------------------------------

create table temp_met.proceso4_colectivas_nounicas as
select cpro_ine,cmuntrib_ine,
COUNT(CASE WHEN CAST(a_ant_bim AS INT) >= anno THEN 1 END) as nuevos,
COUNT(CASE WHEN CAST(a_ant_bim AS INT) < anno THEN 1 END) as antiguos,
COUNT(CASE WHEN a_ant_bim is null THEN 1 END) as nulo,
COUNT(*) as total
from (
SELECT sum(cast(t1.stotalocal as numeric)) as
stotalocal,t2.cpro_ine,t2.cmuntrib_ine,t2.a_ant_bim, COUNT (*) n_constru,
t1.deleg, t1.cmuntrib_dgc, t1.parcela, t1.nordenbifis
from cat2019.r14_2019 t1 left join
      cat2019.r15_2019 t2 on
(t1.deleg||t1.cmuntrib_dgc||t1.parcela||t1.nordenbifis=t2.deleg||t2.cmuntrib_dgc
||t2.parcela||t2.nsecubi)
      where trim(t1.destino_dgc)='V' and t1.nordenbifis is not null and
substring (t1.tipoconstru,1,3)='011' and n_bi_viv<>1
      AND (t1.tip_const4d='0111' or t1.tip_const4d='0112')
      group by t1.deleg, t1.cmuntrib_dgc, t1.parcela,
t1.nordenbifis,t2.a_ant_bim,t2.cpro_ine,t2.cmuntrib_ine
) a
where (stotalocal <=700 and stotalocal >=24)
```

```
group by cpro_ine,cmuntrib_ine;

END $$;


--------------------------------------------------------------------------------

--Recuentos

select sum(nuevos),sum(antiguos),sum(nulo),sum(total) from
temp_met.proceso4_nocolectivas;

----1163110  868369 163   2031642

select sum(nuevos),sum(antiguos),sum(nulo),sum(total) from
temp_met.proceso4_colectivas_unicas;

----25119   23252  29    48400

select sum(nuevos),sum(antiguos),sum(nulo),sum(total) from
temp_met.proceso4_colectivas_nounicas;

----1499009  924517 9     2423535

--Total de viviendas del proceso 4:

--No colectivas + Colectivas únicas en parcela + Colectivas no únicas en parcela

select 2031642+48400+2423535;

----4.503.577

--------------------------------------------------------------------------------

--Unimos las 3 tablas del Proceso 4

create table temp_met.proceso4_AUX as

select coalesce(t1.cpro_ine,t2.cpro_ine) as
cpro_ine,coalesce(t1.cmuntrib_ine,t2.cmuntrib_ine) as cmuntrib_ine,

t1.nuevos_nocolec,t1.antiguos_nocolec,t1.nulo_nocolec,t1.total_nocolec,

t1.nuevos_colec_uni,t1.antiguos_colec_uni,t1.nulo_colec_uni,t1.total_colec_uni,

t2.nuevos as nuevos_colec_nouni,t2.antiguos as antiguos_colec_nouni,t2.nulo as
nulo_colec_nouni,t2.total as total_colec_nouni

from ((select coalesce(a.cpro_ine,b.cpro_ine) as
cpro_ine,coalesce(a.cmuntrib_ine,b.cmuntrib_ine) as cmuntrib_ine,

a.nuevos as nuevos_nocolec,a.antiguos as antiguos_nocolec,a.nulo as
nulo_nocolec,a.total as total_nocolec,

b.nuevos as nuevos_colec_uni,b.antiguos as antiguos_colec_uni,b.nulo as
nulo_colec_uni,b.total as total_colec_uni
from temp_met.proceso4_nocolectivas a
full join temp_met.proceso4_colectivas_unicas b on (a.cpro_ine=b.cpro_ine and a.cmuntrib_ine=b.cmuntrib_ine)) t1
full join temp_met.proceso4_colectivas_nounicas t2 on (t1.cpro_ine=t2.cpro_ine and t1.cmuntrib_ine=t2.cmuntrib_ine));


select sum(total_nocolec)+sum(total_colec_uni)+sum(total_colec_nouni) from temp_met.proceso4_AUX;--ok


--Actualizamos a 0 los nulos
UPDATE temp_met.proceso4_AUX
SET cpro_ine=coalesce(cpro_ine,'00'),
cmuntrib_ine=coalesce(cmuntrib_ine,'000'),
nuevos_nocolec=coalesce(nuevos_nocolec,0),
antiguos_nocolec=coalesce(antiguos_nocolec,0),
nulo_nocolec=coalesce(nulo_nocolec,0),
total_nocolec=coalesce(total_nocolec,0),
nuevos_colec_uni=coalesce(nuevos_colec_uni,0),
antiguos_colec_uni=coalesce(antiguos_colec_uni,0),
nulo_colec_uni=coalesce(nulo_colec_uni,0),
total_colec_uni=coalesce(total_colec_uni,0),
nuevos_colec_nouni=coalesce(nuevos_colec_nouni,0),
antiguos_colec_nouni=coalesce(antiguos_colec_nouni,0),
nulo_colec_nouni=coalesce(nulo_colec_nouni,0),
total_colec_nouni=coalesce(total_colec_nouni,0);


--Realizamos la suma de los 3 tipos de viviena
ALTER TABLE temp_met.proceso4_AUX
ADD nuevos int,
ADD antiguos int,
ADD nulo int,
ADD total int;
```

```
UPDATE temp_met.proceso4_AUX
SET nuevos=nuevos_nocolec+nuevos_colec_uni+nuevos_colec_nouni,
antiguos=antiguos_nocolec+antiguos_colec_uni+antiguos_colec_nouni,
nulo=nulo_nocolec+nulo_colec_uni+nulo_colec_nouni,
total=total_nocolec+total_colec_uni+total_colec_nouni;

--Volvemos a agrupar por municipio
create table temp_met.proceso4 as
select cpro_ine,cmuntrib_ine,
sum(nuevos_nocolec) as nuevos_nocolec,
sum(antiguos_nocolec) as antiguos_nocolec,
sum(nulo_nocolec) as nulo_nocolec,
sum(total_nocolec) as total_nocolec,
sum(nuevos_colec_uni) as nuevos_colec_uni,
sum(antiguos_colec_uni) as antiguos_colec_uni,
sum(nulo_colec_uni) as nulo_colec_uni,
sum(total_colec_uni) as total_colec_uni,
sum(nuevos_colec_nouni) as nuevos_colec_nouni,
sum(antiguos_colec_nouni) as antiguos_colec_nouni,
sum(nulo_colec_nouni) as nulo_colec_nouni,
sum(total_colec_nouni) as total_colec_nouni,
sum(nuevos) as nuevos,
sum(antiguos) as antiguos,
sum(nulo) as nulo,
sum(total) as total
from temp_met.proceso4_AUX
group by cpro_ine,cmuntrib_ine;
```