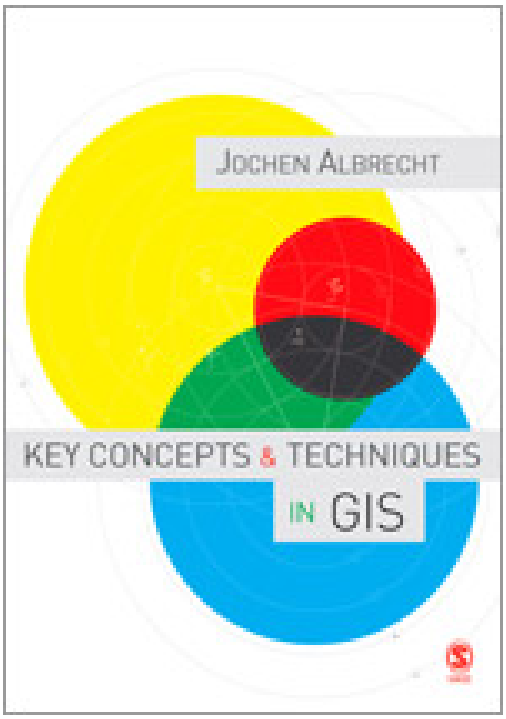


Sage Research Methods

Key Concepts and Techniques in GIS

For the most optimal reading experience we recommend using our website. A free-to-view version of this content is available by clicking on this link, which includes an easy-to-navigate-and-search-entry, and may also include videos, embedded datasets, downloadable datasets, interactive questions, audio content, and downloadable tables and resources.

Author: Jochen Albrecht

Pub. Date: 2011

Product: Sage Research Methods

DOI: https://doi.org/10.4135/9780857024442

Methods: Geographic information systems, Artificial neural networks, Databases

Keywords: literacy, elevation
Disciplines: Anthropology, Geography
Access Date: July 21, 2025
Publisher: SAGE Publications Ltd
City: London
Online ISBN: 9780857024442

Spatial Search

Among the most elementary database operations is the quest to find a data item in a database. Regular databases typically use an indexing scheme that works like a library catalog. We might search for an item alphabetically by author, by title or by subject. A modern alternative to this are the indexes built by desktop or Internet search engines, which basically are very big lookup tables for data that is physically distributed all over the place.

Spatial search works somewhat differently from that. One reason is that a spatial coordinate consists of two indices at the same time, *x* and y. This is like looking for author and title at the same time. The second reason is that most people, when they look for a location, do not refer to it by its *x/y* coordinate. We therefore have to translate between a spatial reference and the way it is stored in a GIS database. Finally, we often describe the place that we are after indirectly, such as when looking for all dry cleaners within a city to check for the use of a certain chemical.

In the following we will look at spatial queries, starting with some very basic examples and ending with rather complex queries that actually require some spatial analysis before they can be answered. This chapter does deliberately omit any discussion of special indexing methods, which would be of interest to a computer scientist but perhaps not to the intended audience of this book.

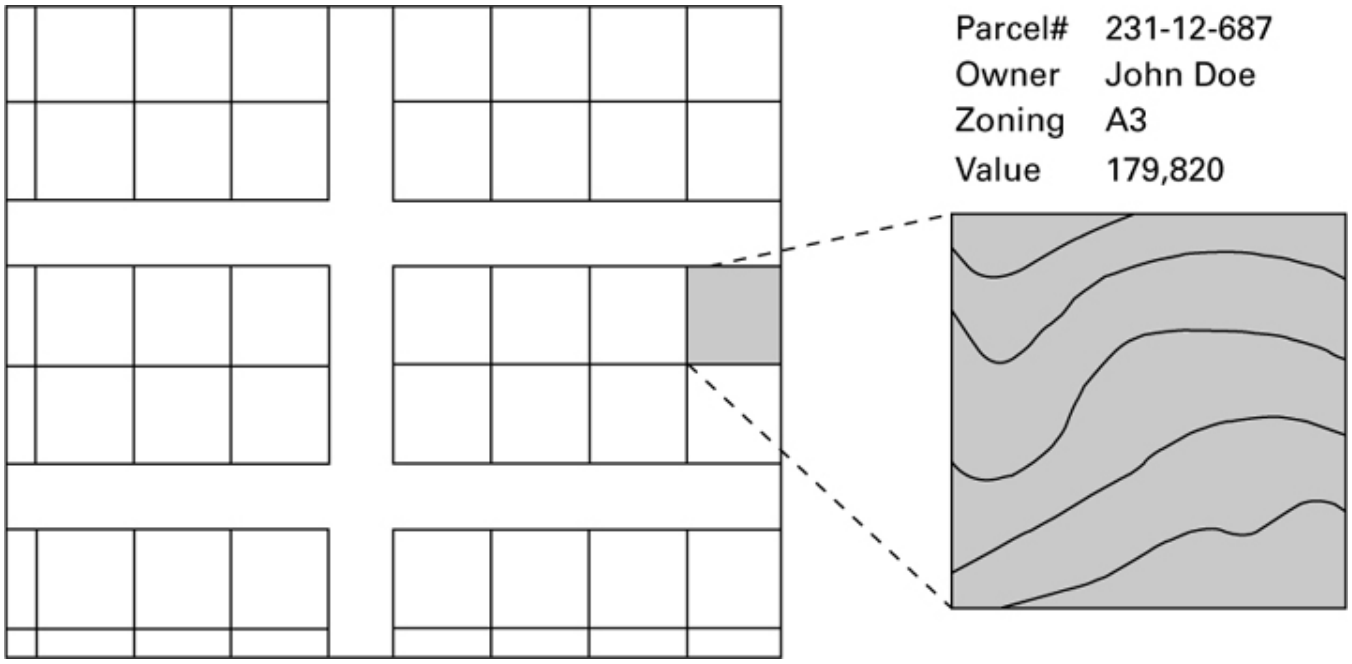
4.1 Simple spatial querying

When we open a spatial dataset in a GIS, the default view on the data is to see it displayed like a map (see Figure 8). Even the most basic systems then allow you to use a query tool to point to an individual feature and retrieve its attributes. They key word here is ‘feature’; that is, we are looking at databases that actually store features rather than field data.

If the database is raster-based, then we have different options, depending on the sophistication of the system. Let's have a more detailed look at the right part of Figure 8. What is displayed here is an elevation dataset. The visual representation suggests that we have contour lines but this does not necessarily mean that this is the way the data is actually stored and can hence be queried by. If it is indeed line data, then the current cur-

sor position would give us nothing because there is no information stored for anything in between the lines. If the data is stored as areas (each plateau of equal elevation forming one area), then we could move around between any two lines and would always get the same elevation value. Only once we cross a line would we 'jump' to the next higher or lower plateau. Finally, the data could be stored as a raster dataset, but rather than representing thousands of different elevation values by as many colors, we may make life easier for the computer as well as for us (interpreting the color values) by displaying similar elevation values with only one out of say 16 different color values. In this case, the hovering cursor could still query the underlying pixel and give us the more detailed information that we could not possibly distinguish by the hue.

Figure 8 Simple query by location



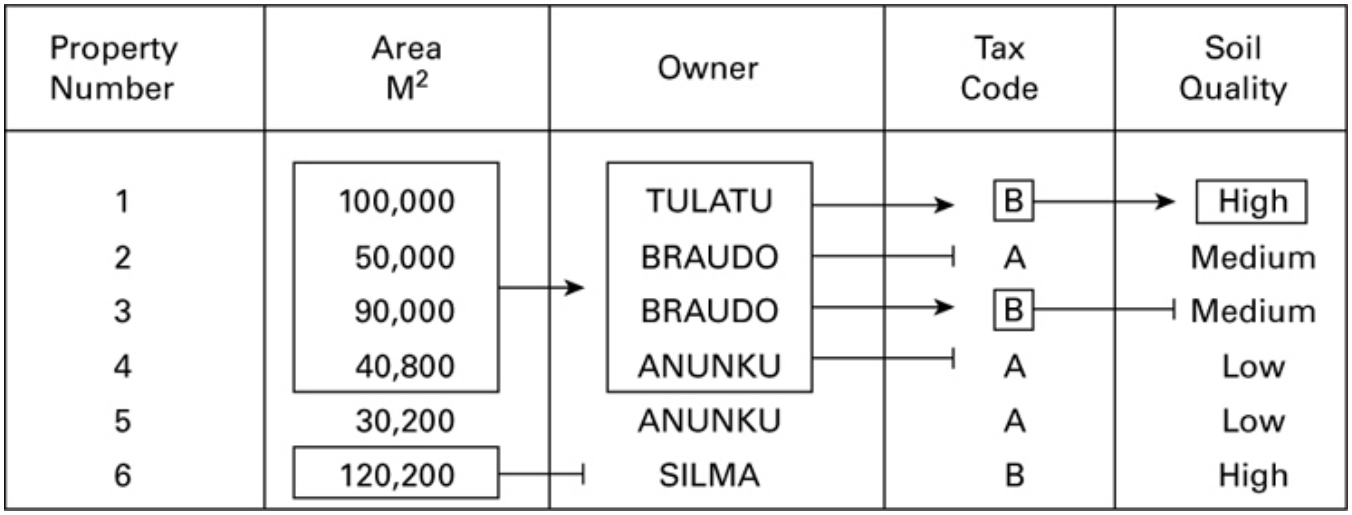
This example illustrates another crucial aspect of GIS: the way we store data has a major impact on what information can be retrieved. We will revisit this theme repeatedly throughout the book. Basically, data that is not stored, like the area between lines, cannot simply be queried. It would require rather sophisticated analytical techniques to interpolate between the lines to come up with a guesstimate for the elevation when the cursor is between the lines. If, on the other hand, the elevation is explicitly stored for every location on the screen, then the spatial query is nothing but a simple lookup.

4.2 Conditional querying

Conditional queries are just one notch up on the level of complication. Within a GIS, the condition can be either attribute-or geometry-based. To keep it simple and get the idea across, let's for now look at attributes only (see Figure 9).

Here, we have a typical excerpt from an attribute table with multiple variables. A conditional query works like a filter that initially accesses the whole database. Similar to the way we search for a URL in an Internet search engine, we now provide the system with all the criteria that have to be fulfilled for us to be interested in the final presentation of records. Basically, what we are doing is to reject ever more records until we end up with a manageable number of them. If our query is “Select the best property that is >40,000m^2, does not belong to Silma, has tax code ‘B’, and has soils of high quality”, then we first exclude record #5 because it does not fulfill the first criterion. Our selection set, after this first step, contains all records but #5. Next, we exclude record #6 because our query specified that we do not want this owner. In the third step, we reduce the number of candidates to two because only records #1 and #3 survived up to here and fulfill the third criterion. In the fourth step, we are down to just one record, which may now be presented to us either in a window listing all its attributes or by being highlighted on the map.

Figure 9 Conditional query or query by (multiple) attributes



Keep in mind that this is a pedagogical example. In a real case, we might end up with any number of final records, including zero. In that case, our query was overly restrictive. It depends on the actual application,

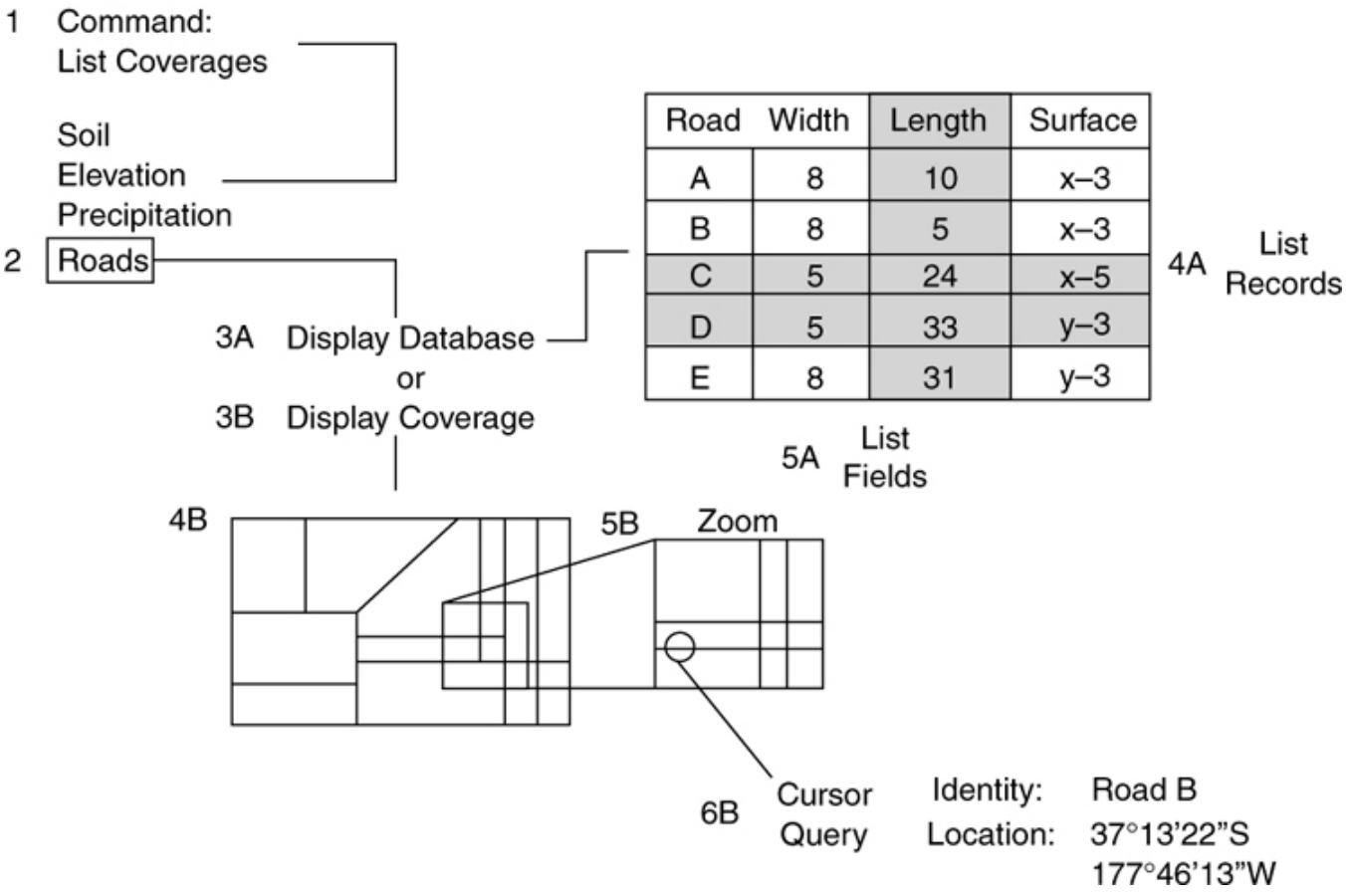
whether this is something we can live with or not, and therefore whether we should alter the query. Also, this conditional query is fairly elementary in the way it is phrased. If the GIS database is more than just a simple table, then the appropriate way to query the database may be to use one dialect or another of the structured query language **SQL.**

4.3 The query process

One of the true benefits of a GIS is that we have a choice whether we want to use a tabular or a map interface for our query. We can even mix and match as part of the query process. As this book is process-oriented, let's have a look at the individual steps. This is particularly important as we are dealing increasingly often with Internet GIS user interfaces, which are difficult to navigate if the sequence and the various options on the way are not well understood (see Figure 10).

First, we have to make sure that the data we want to query is actually available. Usually, there is some table of contents window or a legend that tells us about the data layers currently loaded. Then, depending on the system, we may have to select the one data layer we want to query. If we want to find out about soil conditions and the ‘roads’ layer is *active* (the terminology may vary a little bit), then our query result will be empty. Now we have to decide whether we want to use the map or the tabular interface. In the first instance, we pan around the map and use the identify tool to learn about different restaurants around the hotel we are staying at. In the second case, we may want to specify *‘Thai cuisine under $40’* to filter the display. Finally, we may follow the second approach and then make our final decision based on the visual display of what other features of interest are near the two or three restaurants depicted.

Figure 10 The relationship between spatial and attribute query



4.4 Selection

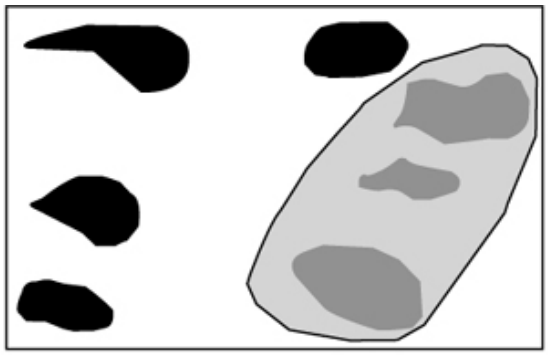
Most of the above examples ended with us selecting one or more records for subsequent manipulation or analysis. This is where we move from simple mapping systems to true GIS. Even the selection process, though, comes at different levels of sophistication. Let's look at Figure 11 for an easy and a complicated example.

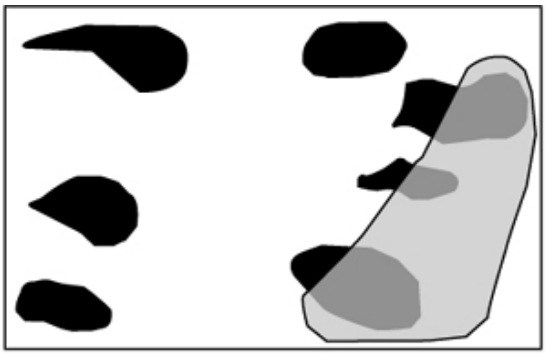
In the left part of the figure, our graphical selection neatly encompasses three features. In this case, there is no ambiguity the records for the three features are displayed and we can embark on performing our calculations with respect to combined purchase price or whatever. On the right, our selection area overlaps only partly with two of the features. The question now is: do we treat the two features as if they got fully selected or do we work with only those parts that fall within our search area? If it is the latter, then we have to perform

some additional calculations that we will encounter in the following two chapters.

One aspect that we have glanced over in the above example is that we actually used one geometry to select some other geometries. Figure 12 is a further illustration of the principle. Here, we use a subset of areas (e.g. census areas) to select a subset of point features such as hospitals. What looks fairly simple on the screen actually requires quite a number of calculations beneath the surface. We will revisit the topic in the next chapter.

Figure 11 Partial and complete selection of features





Stand	Name	Area	Species
A–3	North	20	Pine
C–2	East–1	10	Pine
C–2	East–2	40	Mix

Stand	Name	Area	Species
A–3	North	10	Pine
C–2	East–1	5	Pine
C–2	East–2	30	Mix

4.5 Background material: Boolean logic

This topic is not GIS-specific but is necessary background for the next two chapters. Those who know **Boolean logic** may merrily jump to the next chapter, the others should have a sincere look at the following.

Boolean logic was invented by English mathematician George Bool (181564) and underlies almost all our work with computers. Most of us have encountered Boolean logic in queries using Internet search engines. In

essence, his logic can be described as the kind of mathematics that we can do if we have nothing but zeros and ones. What made him so famous (after he died) was the simplicity of the rules to combine those zeros and ones and their powerfulness once they are combined. The basic three operators in Boolean logic are NOT, OR and AND.

Figure 13 illustrates the effect of the three operators. Let's assume we have two GIS layers, one depicting income and the other depicting literacy. Also assume that the two variables can be in one of two states only, high or low. Then each location can be a combination of high or low income with high or low literacy. Now we can look at Figure 13. On the left side we have one particular spatial configuration not all that realistic because it's not usual to have population data in equally sized spatial units, but it makes it a lot easier to understand the principle. For each area, we can read the values of the two variables.

Figure 12 Using one set of features to select another set

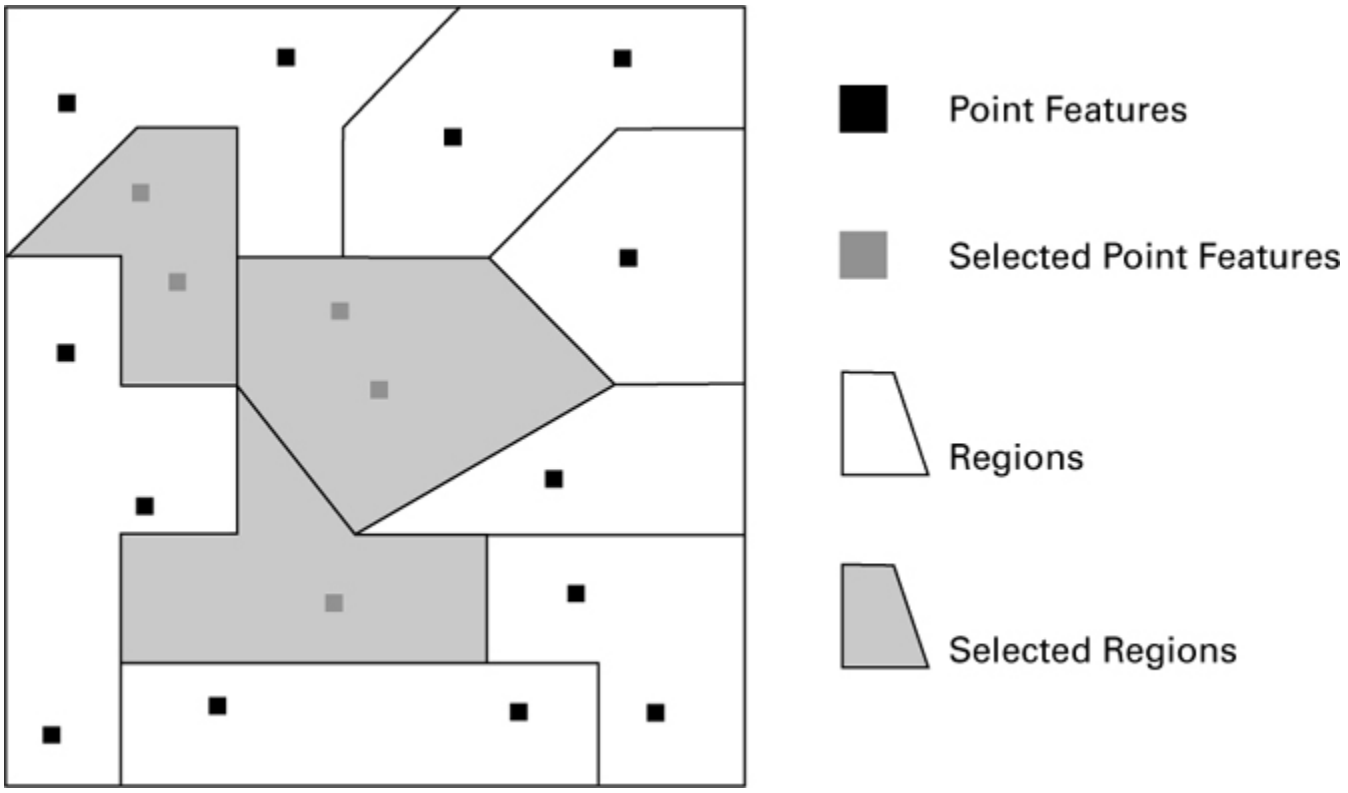
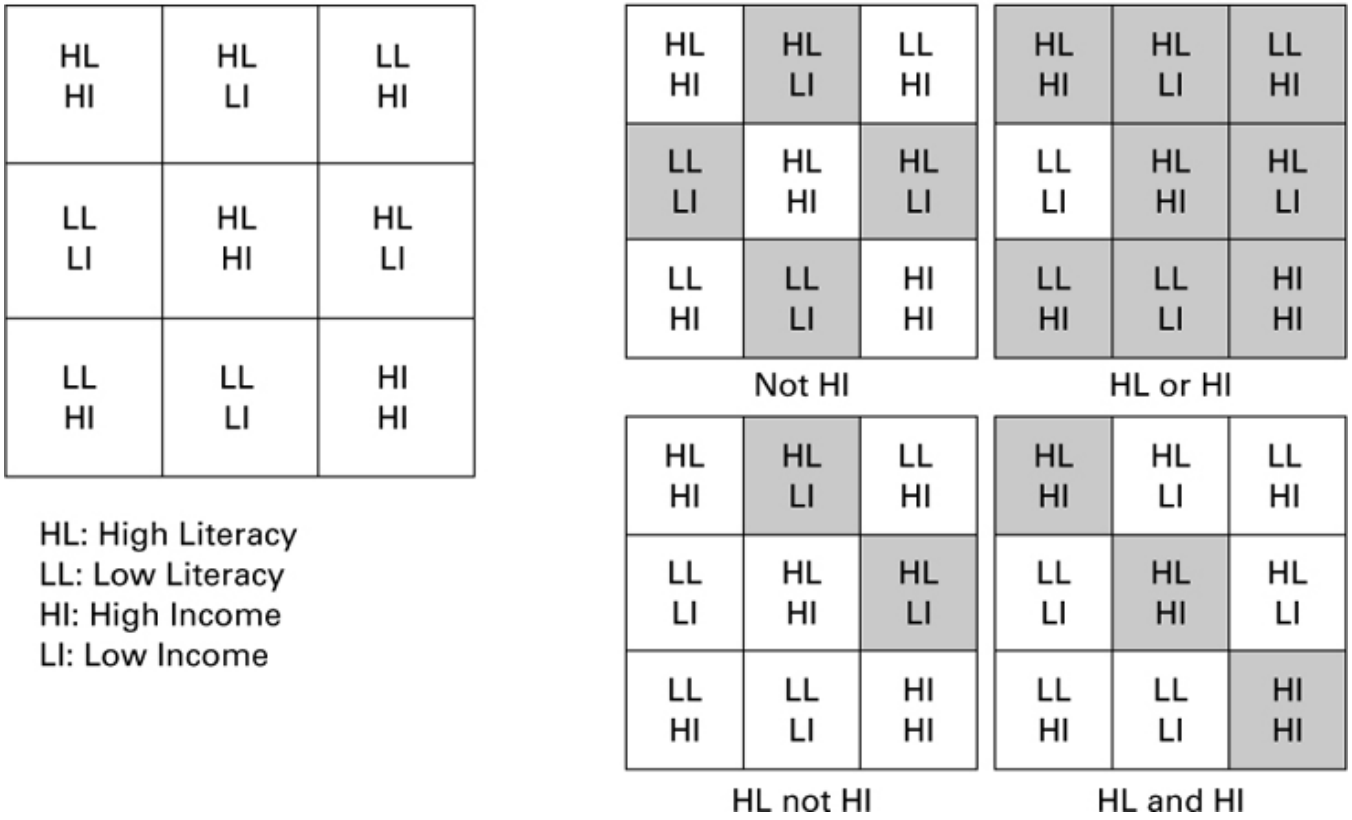


Figure 13 Simple Boolean logic operations



Now we can query our database and, depending on our use of Boolean operators, we gain very different insights. In the right half of the figure, we see the results of four different queries (we get to even more than four different possible outcomes by combining two or more operations). In the first instance, we don't query about literacy at all. All we want to make sure is that we reject areas of high income, which leaves us with the four highlighted areas. The NOT operator is a unary operator it affects only the descriptor directly after the operand, in this first instance the income layer.

Next, look at the OR operand. Translated into plain English, OR means ‘one or the other, I don't care which one’. This is in effect an easy-going operand, where only one of the two conditions needs to be fulfilled, and if both are true then the better. So, no matter whether we look at income or literacy, as long as either one (or both) is high, the area gets selected. OR operations always result in a maximum number of items to be selected.

Somewhat contrary to the way the word is used in everyday English, AND does not give us the combination of

two criteria but only those records that fulfill *both* conditions. So in our case, only those areas that have both high literacy and high income at the same time are selected. In effect, the AND operand acts like a strong filter. We saw this above in the section on conditional queries, where all conditions had to be fulfilled.

The last example illustrates that we can combine Boolean operations. Here we look for all areas that have a high literacy rate but not high income. It is a combination of our first example (NOT *HI)* with the AND operand. The result becomes clear if we rearrange the query to state NOT *HI* AND *HL.* We say that AND and OR are binary operands, which means they require one descriptor on the left and one on the right side. As in regular algebra, parentheses () can be used to specify the sequence in which the statement should be interpreted. If there are no parentheses, then NOT precedes (overrides) the other two.

https://doi.org/10.4135/9780857024442