

Maps in Computers

Searching Geographic Information

Once data have been stored in the geographic information system, we will want to be able to access it. The process of retrieving geographic information from a GIS involves both standard database query operations and geographically-based search criteria. The result is the ability to organize and find information in an integrated fashion.

Querying a database involves selecting one or more records from the entire set of records in the database. We are normally interested in only a subset of the data (otherwise we really don't need to perform the query operation). The query process involves reducing the total set of records to just those in which we are interested. We do so by creating query expressions that select the records in which we have an interest and filter out those records in which we have no interest.

The query expression consists of variables, constants, and operators. The variables are names of fields from which we are drawing data values. The constants are values against which we compare the values from the specified field. Constants may be numeric values, including integers (no fractional parts, such as 1, -5, 256) and floating point (with fractional parts, such as 3.1415927). Constants may also be text, or string values, which will normally need to be enclosed in quotes ("California"). The type of data for the constants must match the type of data in the field being examined, so if the field contains text values, the search constant must be text.

We can construct a query expression that specifies a field by name (e.g., "MTBE99" representing MTBE levels for wells in 1999) and a constant value (e.g., "10", representing a value in parts per million). We then need to indicate the nature of the relationship in which we have an interest. In this case, we would specify a relational operator such as "greater than." The expression would look like this: `[MTBE99] > 10`, where the name enclosed in brackets ("[]") is the name of a field in our table. Query expressions may also compare the values in one field with values from another field. For example, we might be interested in those records in which the MTBE levels in 1999 are greater than those in 1997 (assuming we have fields with MTBE concentrations for those two years), with an expression like this: `[MTBE99] > [MTBE97]`.

The basic database query operations include relational operations (equal to, not equal to, greater than, less than, etc.) and Boolean operations (and, or, not, xor). The relational operators are standard equality and inequality operations. The Boolean operators provide connections between expressions that result in a Boolean (true/false) value. *And* is the intersection operator, indicating that both operands in the expression are true. *Or* is the union operator, indicating that one or both of the operands in the expression are true. *Not* is the negation operator, producing a result that is the opposite of the

operand's value (e.g., Not A results in a value of false if A is true, and a value of true if A is false). *XOR* is the exclusive or operator, indicating that one of the operands is true, but not both (e.g., A xor B results in a value of true if either A or B is true and a value of false if A and B are both true or both false).

We begin with fairly simple queries, such as find all wells with 1999 MTBE levels greater than 10 parts per million, and can create increasingly complex queries that combine other, simpler query operations, such as find all abandoned wells with 1999 MTBE levels greater than 10 parts per million.

Basic Query Operations

Type	Operators
Relational	=, >, <, >=, <=
Boolean	and, or, not, xor

Each query results in the selection of those records satisfying the query expression. The query essentially results in a true/false (or Boolean) answer. That is, evaluating the query expression using the data values for each data record leads to a selection of the record for a value of “true” and no selection if the value is “false”. Therefore, you can configure your query to reduce your selection to exactly the records of interest by building compound expressions.

Compound expressions incorporate two or more basic expressions linked by a Boolean operator. For example, we might be interested in wells with 1999 MTBE levels greater than 10 parts per million that show higher levels in 1999 than in 1997 (i.e., wells that showed an increase from 1997 to 1999). The query expression would look something like this: $([MTBE99] > 10)$ and $([MTBE99] > [MTBE97])$. Each part of the expression enclosed in parentheses represents a single, basic query. Each will result in either a true or false value. In order for the entire expression to be true, then both parts of the expression must be true. So only those records that show MTBE99 values greater than 10 and show an increase from 1997 to 1999 will be selected.

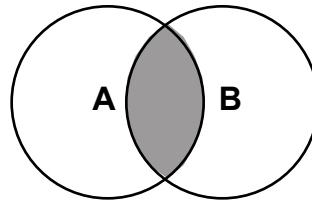
Alternatively, we could create a somewhat different expression by changing the operation. Suppose we are interested in wells with 1999 MTBE levels greater than 10 and any well that showed an increase from 1997 to 1999, no matter what its 1999 MTBE level. In this case, we would use the “or” operator in the expression: $([MTBE99] > 10)$ or $([MTBE99] > [MTBE97])$. This results in a overall value of true if either of the sub-expressions is true. We end up with a larger selection (potentially) than if we had used the “and” operator.

Results from Boolean Operations

These tables show the results of Boolean expressions for different values of A and B. A and B may be values in a database field that may take on values of either true or false, or may be the result of a relational expression, such as $MTBE99 > 10$, that produces a result of true or false. Venn diagrams, such as those shown to the right of each table, graphically depict the results of Boolean operations. The circles represent the sets of values included for A and B. The shaded areas represent the values that satisfy the Boolean expression. For example, for A and B, the shaded area includes those parts of A that are also in B.

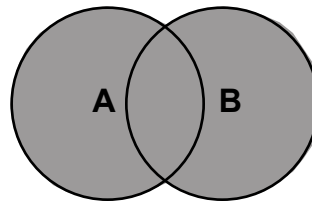
A and B

A	B	Result
T	T	T
T	F	F
F	T	F
F	F	F



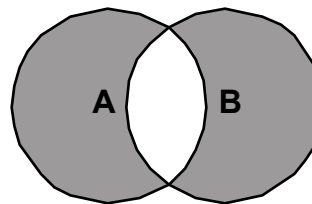
A or B

A	B	Result
T	T	T
T	F	T
F	T	T
F	F	F



A xor B

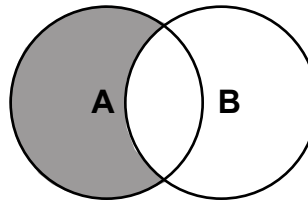
A	B	Result
T	T	F
T	F	T
F	T	T
F	F	F



An example of the negation operator (Not)

This table shows the results of Boolean expressions for different values of A and B, where we have negated B using the Not operator. The original value of B is shown, and the result of the negation (taking the opposite of B) is shown in parentheses.

A and (not B)		
A	B	Result
T	T (F)	F
T	F (T)	T
F	T (F)	F
F	F (T)	F



Spatial Search and Retrieval

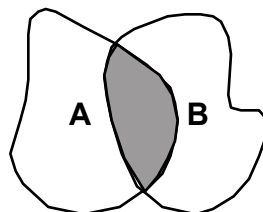
Finding and retrieving information using geographic, or spatial operations is the primary function for a geographic information system. Relational queries provide the basis for selecting data from a database, and a GIS could not function without them. However, what distinguishes a GIS from other information systems is that information can be accessed spatially, in addition to relationally. This provides the ability to analyze geographic distributions of any kind and makes the GIS a very powerful tool.

Spatial searches and retrievals involve many of the same types of operations we perform with other database queries. We are interested in selecting some of the features (records) from our database based on some established criteria. Spatial searches differ, however, in the way in which these searches are performed.

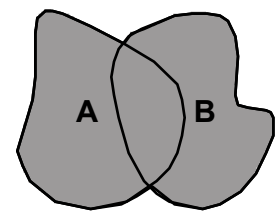
The major feature of spatial searches is that we use geographical features to perform part or all of the search request. This means searching by points, lines, or areas. We compare one set of features against another set of features. We can restrict and extend the selection set through the use of geographical operations we call “overlay.”

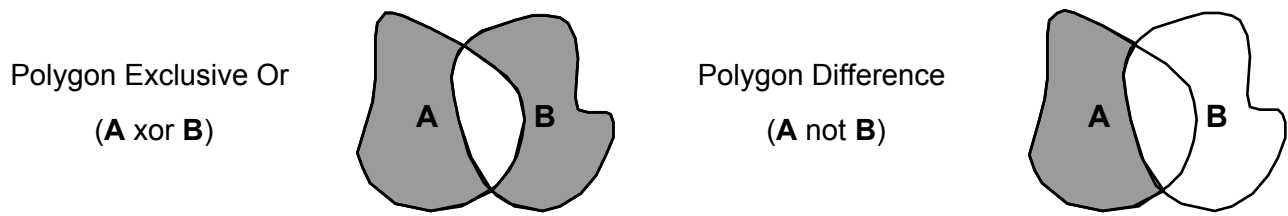
The overlay operators include intersection, union, and difference. These operations produce the geographical equivalents of the results of Venn diagrams, particularly for polygons. For example, the intersection operation produces the subset of the two input polygons that overlap.

Polygon Intersection
(A and B)

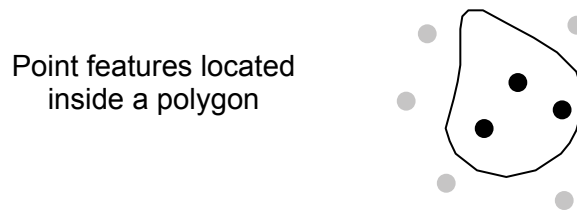


Polygon Union
(A or B)



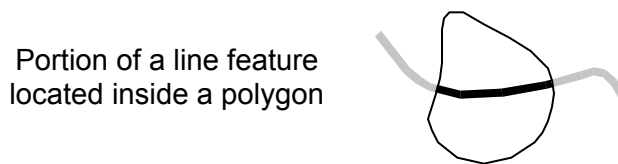


Ordinarily, we are interested in the intersection operation. That is, we want to find where one set of geographical features overlaps another. For example, we might want to know what schools are in a fire station’s response zone. In this situation, we would be looking at the schools as points and we would like to find only those in the specified response zone.



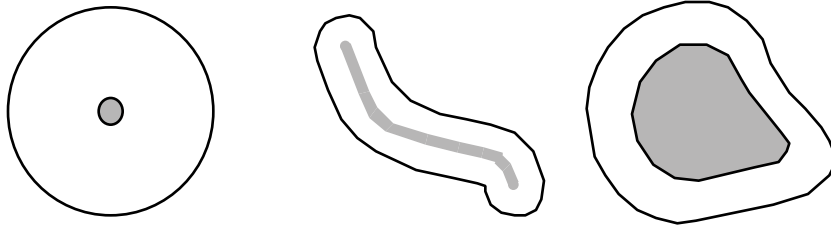
Point features located inside a polygon

Or we might be interested in the portions of a road running over a particular zone, such as an area with an unusually high water table.



Portion of a line feature located inside a polygon

Comparing points and lines to one another generally involves specifying a zone called a “buffer” around the feature that can be used in the search. A buffer is a polygon defined relative to the feature, usually as a distance zone around the feature. For example, we might be interested in locating all elementary schools located within 0.25 miles of a major road or highway that was in service before the introduction of unleaded gasoline for automobiles. These would be areas that might have higher than acceptable levels of lead in the soil. We would first identify roads that satisfied the relational query ([year opened]<1970). Then we would construct a 0.25 mile buffer around the selected roads. Finally, we would select elementary schools and compare them to the buffer to identify those within the buffer zone.



Buffers may be constructed around all geographical feature types.

Buffer zones may also be defined for nodal features and areal features. For example, we might be interested in potential customers living within 1 mile of our store. Or we might be interested in the numbers of people located within different distance zones around an airport who may be affected by changes in airport operating procedures. Finally, buffer zones may be defined by criteria other than distance, such as travel time. So our analysis of potential customers might be based on the how many potential customers live within 15 minutes total travel time of our store.