SPATIAL QUERY LANGUAGES

Agnès Voisard
Computer Science Institute, FreieUniversität Berlin, Germany

Keywords: spatial database management system, geographic information system, data definition language, data manipulation language, SQL, abstract data type, object-oriented models

Contents

Summary

Spatial database systems store and manage data that encompass a spatial component, such as architectural data, environmental information, navigation networks, and geomarketing data. This article focuses on the interaction between end users and data stored in a spatial database, and more specifically on the type of request that end users are likely to ask the system (i.e., fetch data and perform operations on the data). It presents the current tools offered by a spatial database system to allow end users to express these various requests, namely spatial query languages. Three basic types of interaction are presented: using a textual query language (most of the time SQL, extended to the spatial component of the data), using a visual, graphical language with a drawing zone on the screen, and finally using a graphical set of predefined actions (menu-based interaction).

1. Introduction

Spatial database systems store and manage data that encompass a spatial component, such as environmental information, architectural data, geologic data, navigation networks, and geomarketing data. A spatial query language is a means for an end user to communicate with such systems: in other words to retrieve and manipulate the data that they store. The term “end user” is employed in opposition to the term “designer,” whose main function is to define the database: that is, the appropriate execution environment (modules and data structures) for particular applications. Database definition is achieved through the use of a data definition language (DDL) whereas end users are concerned with a data manipulation language (DML). This text assumes the reader is familiar with the basic notions of database management systems, data modeling, and query languages. The paper’s focus is on geospatial applications as they are a rich representative example of spatial data handling. Usually, such applications are managed by a geographic information system (GIS), whose kernel is a spatial database management system (DBMS). A spatial DBMS offers all basic functionalities of any database management system: the management of large amounts of data and of structured information, plus additional abilities to handle the spatial dimension (see *Spatio-Temporal Information Systems*).

This article hence concerns the interaction between end users and data—instances—stored in a spatial database. More specifically, its focus is on:

- the type of request that end users are likely to ask the system (i.e., fetch data and perform operations on the data), and
- the tools offered by a spatial database system to allow end users to express these various requests, namely spatial query languages.

Basic end user requests are (a) retrieving data from the database and (b) performing operations on the data. Spatial queries may concern an entire database (e.g., to retrieve all data satisfying a given condition) or be applied to a pre-selected part of this database (typically visualized in a map). Examples of queries that end users may pose include:

- From a map library that stores a set of maps, fetch and display a map of Sierra Leone.
- Show me where Bordeaux is on a map of France.
- Show me all cities with more than 700,000 inhabitants on this US map.
- On this geologic map, show me all regions with a concentration of cobalt greater than 40%.
- On a world map, what are the countries having no islands?

Note that presentation issues are not considered here, e.g. “Display highways in blue and forests in green,” or “Zoom by one factor into the map of Europe.” Those are left to graphical user interface tools whose objects of interest are (map) displays and not databases. Interface visualization and interaction concerns are discussed under *Interacting with GIS: From Paper Cartography to Virtual Environments.*

The peculiarity of geospatial information handling is based on the fact that neither standard models nor query languages exist to manipulate this kind of data because of the presence of its spatial component. Roughly speaking, a geographic database can be seen as organizing data into *themes* (sometimes called *maps*) made of *geographic entities*. A geographic entity is an entity of the real world, such as a county, a river, or an area where corn is grown. Basically, such an entity has two kinds of components: an alphanumeric component, such as the population of a county, and a spatial component (e.g., the geometry of this county). Handling the latter in a standard DBMS is a challenging task.

As with all current database query languages, present spatial query languages can roughly be classified into two categories: text based and visual spatial query languages. However, the requirements for visual languages in this context go beyond those for query languages designed for standard databases. In addition, each of the two previously cited categories can also be further refined as illustrated later in this paper.

In an orthogonal way to the previous point, query languages can be divided into two groups: imperative query languages, with queries defined step by step; and declarative query languages, with queries defined as an expression that the result should verify. The second approach is seductive but is also a challenge to implement in the context of spatial query languages.

This paper is organized as follows. Section 2 studies the typical interaction between end users and a spatial database management system and gives the basic vocabulary. It presents a reference database schema as well as reference queries that will be used throughout this paper. A list of requirements for a spatial query language ends this section. Section 3 presents text-based interaction. This type of interaction is usually based on the popular SQL language. Its use is illustrated in the context of both relational and object-oriented systems. Section 4 focuses on visual approaches, whereas Section 5 describes other ways of interacting with a spatial database system. Finally, Section 6 addresses concluding remarks.

2. Interacting with a Spatial Database System

In communication with a spatial information system, there is a fundamental distinction to be made between the various users of the system, namely the end users and the designers. The latter develop applications using a data definition language (DDL) and programming languages, and examine the actions that the end users are likely to perform in order to tune the system accordingly. End users, on the other hand, utilize the defined actions and interact with the system through the use of a data manipulation language (DML) in order to fetch information and perform operations on data stored in the underlying database. End users can be further divided into sophisticated end users who are able to use an elaborate query language, and naïve users who would rather consider sets of predefined ("canned") actions. For more details on user profiles, and distinct kinds of applications, see *Advanced Geographic Information Systems*.

This section explores the functionalities that a query language should provide in the context of spatial information querying. For this purpose, it uses a set of reference *schemas* as well as a list of queries that one is likely to pose against the proposed schemas. In database terminology, a (database) schema is a way to describe structure information. A schema is made of entity descriptions, consisting of entity names (for instance Person or City) and their *attributes*, each of them having a certain *type*. A basic type is for instance an integer or a string. The following expression: Person

(Firstname:string, LastName:string, SocialSecurityNumber:integer) is a schema containing one entity, Person. The first two attributes of this schema are of the string type and the third is of the integer type. A spatial type is a nonstandard type, in the sense that most standard DBMSs do not consider such types. More recent DBMSs allow application designers to define new types, thus supporting the definition of spatial types, as will be seen later in this paper. At implementation time, according to the underlying model that stores the information, one will consider particular schemas such as a relational schema (made of relations) or an object-oriented schema (made of object classes). We use the term “schema” in its broad sense here.

2.1 Reference Schema

All examples in this paper will be given in terms of a toy example, described in this section. Let us assume the existence of the three schemas given in Figure 1. They are expressed in terms of a graphical notation common in conceptual data modeling (see *Conceptual Modeling of Geographic Applications*). At implementation time, these schemas will be stored according to the data model of the spatial database management system. For instance, an entity will translate into a relation in the relational model and into a class in an object-oriented model.

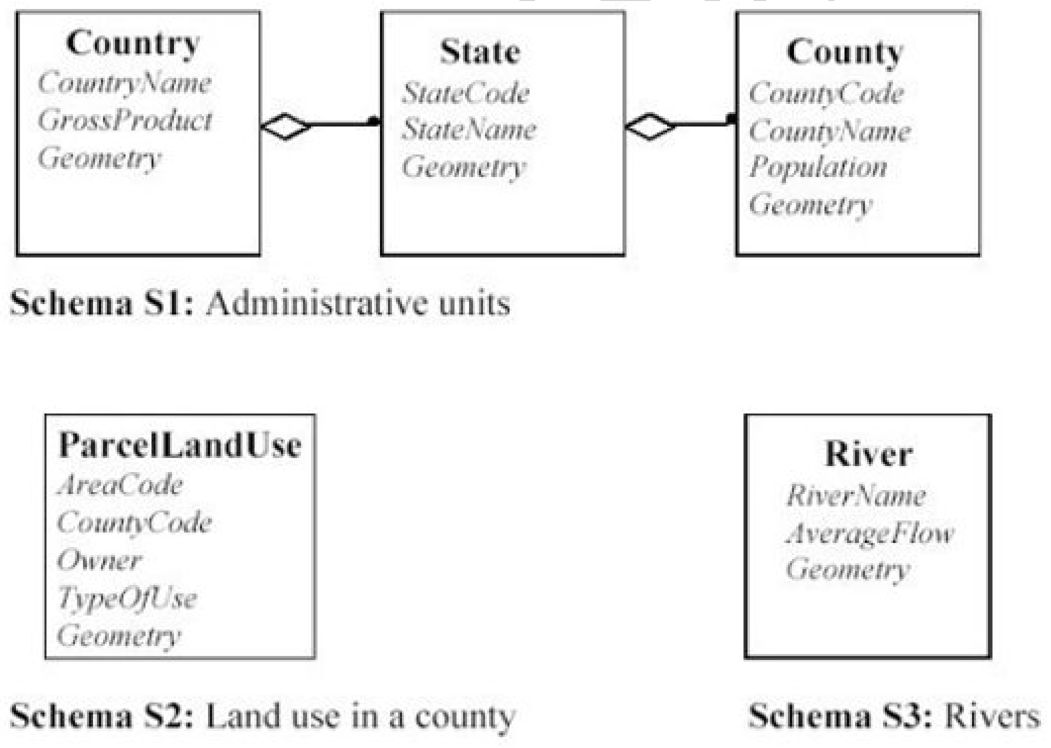

Figure 1. Three reference schemas

The toy application considers three different schemas, S1, S2, and S3. They are all part of this single application and they will sometimes be queried simultaneously if the desired information resides in more than one schema. Notice that the diagrams of Figure

1 do not encompass the type of the attributes. For instance, the name of a country (attribute *CountryName*) is a string, but, in order to keep it simple, the keyword "string" does not appear in this representation.

Schema S1 is concerned with administrative units in countries and contains three entities: Country, State, and County. Even though the vocabulary used in the description of the entity types is the one used in the United States of America, there exists a mapping between this decomposition and the one used in most other countries. For example, a State is a *Land* in Germany and a County is a *Commune* in France. An instance of entity type Country is for example "the United States," an instance of State "the State of California," and, finally, an instance of entity type County may be "Santa Clara County."

The diamond shape symbol between entity types Country and State (and that between State and County) indicates that a country is composed of states (and a state is composed of counties). This is an illustration of the *aggregation* concept. Each entity has its own alphanumeric attributes, for instance *CountyName* and *GrossProduct* in entity type Country, and one spatial attribute denoted *Geometry*. This field describes a particular (two-dimensional) part of the plane, such as a polygon or a set of non-connected polygons—for example, the geometry of the United States is a set of polygons which includes mainland areas and islands. In the following, such a two-dimensional attribute will be referred to as a *region*. Note that each of the three levels, Country, State, and County, is a partitioning of the plane (no region overlapping).

Schema S2 is concerned with a simple modeling of the land use. Only one entity is relevant to our application, ParcelLandUse, which corresponds to an area of interest (a parcel) in a county. Such an area has four alphanumeric attributes: its code, the code of the county it belongs to, its owner, and its type of use (e.g., "industrial," "rural," "forest," or "residential"). In a real application, the possible values—called the *domain* in the database vocabulary—are usually predefined. Note again that the proposed schema corresponds to a simplified version of the reality.

The geometry of such an area, materialized by attribute *Geometry*, is again a two-dimensional object, although, in contrast to S1, there may be overlapping of the geographic areas. As an example, an area of interest can be both residential and located in a forest. A true partitioning of these areas would imply a predominance of a value, for instance residential, over another one, for instance forest. Finally, Schema S3 concerns rivers and encompasses one entity only, River, with the following three attributes: *RiverName* and *AverageFlow* (both alphanumeric); and *Geometry* which is a polyline, a one-dimensional spatial object. Note that here a river is assimilated with a (one-dimensional) line, although this is realistic only at a small scale.

One fact reflected in this example is that geographic applications must frequently combine data collected for different purposes—and thus stored in distinct database schemas. The factor that allows the uniting of all these databases is the geographic location of the phenomena they portray. Thus, the underlying assumption for the queries discussed here is that the data concern the same geographic region. If this were not true, queries across these databases would be meaningless.

-
-
-

Bibliography

Egenhofer M. (1992). Why not SQL! *International Journal of Geographical Information Systems* **6**(2), 71–85. [This paper includes a discussion on the (non-)adequacy of SQL in a spatial context.]

Egenhofer M. and Frank A.U. (1988). Towards a spatial query language: user interface considerations. *Proceedings of the International Conference on Very Large Databases (VLDB)*, pp. 124–133. [This paper sets an important basis for spatial query languages from a graphical user interface viewpoint.]

El Masri R. and Navathe, S. (1994). *Fundamentals of Database Systems*, 873 pp. Redwood City, CA: Benjamin Cummings. [This text book is a good reference for a description of the relational model and its extension as well as object-oriented models and query languages.]

Frank A.U. (1982) MAPQUERY: database query language for retrieval of geometric data and its graphical representation. *ACM SIG Computer Graphics* **16**(3), 199–207. [This is a pioneering study in terms of map querying.]

Güting R.H. (1988). Geo-relational algebra: a model and query language for geometric database systems. *Proceedings of the European Conference on Database Theory (EDBT)*, pp. 506–527. [This is one of the most solid references on spatial querying in an extended-relational context.]

Ooi B.-C. (1990). *Efficient Query Processing in Geographic Information Systems, Lecture Notes in Computer Science No.471*, 208 pp. Berlin/Heidelberg/New York: Springer-Verlag. [A comprehensive introduction to the field of spatial querying and query processing.]

Mainguenaud, M. and Portier-Aufaure, M.-A. (1991). Cigales: a graphical query language for geographical information systems. *Proceedings of the 4th International Conference on Spatial Data Handling (SDH)*, pp. 393–404. [One of the first attempts to define a visual spatial query language.]

Rigaux, P., Scholl, M., and Voisard, A. (2001). *Spatial Databases—With Application to GIS*. 432 pp. San Francisco: Morgan Kaufmann. [This book devotes four papers to spatial data modeling and querying as well as to query processing.]

Zloof, M. (1975). Query-by-example: the invocation and definition of tables and forms. *Proceedings of the International Conference on Very Large Databases (VLDB)*, pp.1-24. [The first graphical query language for databases]

Biographical Sketch

Agnès Voisard received her Master's and Ph.D. degrees in computer science from the University of Paris at Orsay (Paris XI) and INRIA (French Institut National Recherche en Informatique et en Automatique) in 1989 and 1992, respectively. The focus of her Ph.D. work was the design of both a model and a graphical user interfaces for an object-oriented spatial database. During the academic year 1991–1992 she was a research assistant in the database group of the Conservatoire National des Arts et Métiers (CNAM) in Paris. In 1992–1993 she was an INRIA postdoctoral fellow at Ludwig Maximilian University in

Munich. In 1993, she was appointed Assistant Professor of Computer Science at the Freie Universität Berlin, where she obtained her Habilitation in 1999. In 2001 she joined Kivera, Inc. (Oakland, California) as a system architect. Her areas of expertise center around data modeling, geographic information systems, human–computer interaction, and interoperability in information systems. More precisely, her research work in the past few years has been focusing on geospatial data modeling and query languages, navigation systems, object-oriented modeling, graphical user interfaces for multimedia applications, and integration and management of heterogeneous documentation. She has over 30 publications in these domains. She has participated in several program committees and was general chair of the 5th International Symposium on Spatial Databases (SSD'97). She is a member of the Editorial Board of *GeoInformatica* and of the Steering committee of the I-20 initiative on GIS interoperability, US National Center of Geographic Information and Analysis (NCGIA). With Philippe Rigaux and Michel Scholl she co-authored *Spatial Databases—With Application to GIS* (Morgan Kaufmann, 2001). She was program co-chair of the 2002 ACM GIS symposium.