GEOGRAPHIC INFORMATION PROCESSING USING A SQL-BASED QUERY LANGUAGE

Kevin J. Ingram
William W. Phillips
Kork Systems, Inc.
6 State Street
Bangor, Maine 04401

ABSTRACT

The utility of a Land Records or Natural Resource information system is greatly enriched if its geographic (or map-based component) and associated attribute data components can be integrated. Differences in how these two classes of data are now processed, in particular, the lack of a common query language, have impeded this integration. Kork Systems' new Geographic Information System (KGIS) combines separate geographic and attribute data bases into a single, integrated system. Geographic data (polygons, lines and points) are maintained in a fully-intersected topologic data structure with direct links to their associated attributes. The query language used in KGIS is based on the Structured Query Language (SQL). Several important additions were made to SQL to incorporate spatial concepts into the query language, including location, area, length, proximity and geographic context. These additions enable the user to form rapid, on-line queries about complex spatial relationships among the data.

INTRODUCTION

The utility of a Land Records or Natural Resource information system is greatly enriched if the spatial and non-spatial attribute components associated with geographic features can be integrated. Differences in how these two classes of data are now processed, in particular, the lack of a common query language, have impeded this integration. A query language is grounded in the data model on which it is based. In discussing the query language developed for Kork Systems' new Geographic Information System (KGIS), it is necessary to

understand the data model on which KGIS is built. The remainder of this paper comprises a review of the data models available to a Geographic Information System (GIS) developer, a short overview of the data model used in KGIS and finally, a description of the query language used in KGIS.

DATA MODELS FOR GEOGRAPHIC FEATURES

The continuing development of GIS technology has involved two concurrent trends: 1) increasing sophistication of data models for both the spatial and non-spatial (attribute) information about geographic features, and 2) strengthening links between the spatial and non-spatial attribute portions of the information.

Data models for non-spatial attribute information have progressed from special purpose files designed to be accessed by specific application programs, through the hierarchical and network data models used in the first generalized data base management systems (DBMS), to the relational model which has emerged as a powerful and flexible structure for representing attribute information in tabular form.

The special characteristics of spatial data (Peuquet, 1984) have presented a challenge to GIS developers in their search for suitable data models. Early geographic data bases consisted of a catalog of files, segregated by map sheet and data layer, containing either vector or tessellated data.

Vector data usually consisted of polygon-digitized ("double-digitized") or spaghetti-digitized segments. Dueker (1985) has described the shortcomings of this data model. The topologic model, an improvement on earlier vector structures, made it possible to relate one feature to other features within a data layer. Relating data between layers, however, required an expensive polygon overlay operation.

Tessellated data, either grid-cell or raster encoded, provide for simpler computations at the cost of increased data volume and lower positional precision. The application of hierarchical tessellated data structures (quadtrees and other variants) to geographic data bases reduced the data volume burden and made spatial searches efficient.

The success of the relational model for managing non-geographic data has led to attempts (Waugh and Healey, 1985) to apply it to geographic data as well. This approach has the strength of storing the data in a sophisticated DBMS and the flexibility of the relational model. Since all information is represented by a single data model, this approach should provide a strong link between the spatial and non-spatial attribute components

of the geographic information. It generally suffers by forcing the user to manipulate the information about geographic features at a very low-level (Abel and Smith, 1985).

The more recent development of the object-oriented data model has provided a powerful tool for managing geographic data, including model constructs such as classification, generalization and aggregation. One system (Frank, 1986) based on an object-oriented data model, PANDA, is implemented on a network DBMS and provides spatial access to geographic features through a modified quadtree structure for data storage. The object-oriented approach allows considerable knowledge about the behaviour of objects to be embedded in the system. However, this inherently limits the system's flexibility with regard to changes in the schema since any modification necessitates re-programming.

In summary, there does not yet appear to be a single data model for geographic features which is superior in every aspect to all other data models. As a result, the link between the spatial and non-spatial attribute components of geographic features has usually been either weak or non-existent. This shortcoming can be reduced by using a hybrid data model (Morehouse, 1985). However, if the hybrid nature of the data model is visible externally, then the user will be forced to work in one mode or the other. This tends to segregate specific capabilities by mode which reduces the systems overall flexibility.

KGIS OVERVIEW

In designing KGIS, we defined three functional requirements. First, the system should handle spatial and non-spatial attributes in a single context to give the user the maximum amount of flexibility in formulating queries. Second, it should have the ability to relate geographic features in one layer, e.g. soils, with those in other layers, e.g. bedrock geology or roads. Third, the system should not impose a fragmentation of the spatial data upon the user, but should maintain them as a seamless whole while permitting the user to define an arbitrary subset. To accomplish these requirements, we decided to distinguish between the external and internal views of the data, designing each view to support a different data model.

The external view supports the relational model, giving the user access to all information in the data base including both spatial and non-spatial attributes of geographic features. The user views geographic features at a high level, manipulating them in the same context as other information. Each thematic layer of geographic features is represented in a separate relational table.

Internally, KGIS is implemented on a hybrid data model (Keating et al, forthcoming). Non-spatial attribute data are maintained in a relational DBMS. The relational model provides the necessary flexibility in the schema. Spatial data are maintained in an object-oriented DBMS, PANDA. The object-oriented approach provides the high-level view of geographic features by hiding inner complexity of the data structure in the lower levels of the system. The data model for spatial attributes contains elements of several other data models as well. The schema defined for the spatial data stores them in a fully intersected topologic data structure, an improvement on the layer-by-layer topologic model. This schema permits a geographic feature in one layer to be related to features in other layers as easily as in the same layer. Finally, the PANDA DBMS provides a hierarchical tessellated data structure for storing and accessing spatial data. Therefore while no artificial fragmentation of the spatial data base is necessary, this capability enables rapid spatial searches of any arbitrary user-defined subset of the data base.

KGIS QUERY LANGUAGE FEATURES

SQL (Chamberlin et al., 1976) will soon be an official standard, query language for relational DBMSs. For this reason, it was chosen as the basis of the KGIS query language. However, SQL suffers from the same shortcomings as other commercial DBMS query languages when applied to geographic data management (Abel and Smith, 1985). Missing but necessary facilities include 1) a high level view of geographic features, 2) support for the graphic display of geographic features and 3) an ability to express spatial relationships as selection criteria. In addition, an adequate geographic query language (Frank, 1982) must provide support for 4) geographic context specification, 5) graphical context specification and 6) graphical input.

To provide these facilities, we have made several additions to SQL. Since maintaining compatability with the SQL standard is a priority, we have retained the overall syntax whenever possible, adding extensions as new commands or as functions for use with existing commands. The resulting language provides facilities in all of the areas described above.

High Level View Of Geographic Features

Geographic features are distinguished from other data base entities, which have only non-spatial attributes, by having spatial attributes and spatial relationships to other geographic features as well. By treating them as objects, the KGIS query language provides a high level view of geographic features, relieving the user from manipulating the complex internal representation of geographic information directly. Geographic features

appear as entire entities, such as 'Parcel 123' or 'Route 1', which have both spatial and non-spatial attributes.

Graphic Display Of Geographic Features

The graphic display of geographic features involves two issues: 1) specifying the information to be displayed and 2) specifying the format in which to display it.

All geographic features have associated positional information which, when taken in its entirety, can be expressed graphically as a map. The map for a geographic feature, therefore, may be treated as one of its spatial attributes. To view the map of a feature, the user simply retrieves it like any other attribute. The following query

```
SELECT MAP FROM PARCELS WHERE VALUATION > 100000;
```

displays, on the graphics screen, a map of each parcel whose value is greater than $100000.

On an ordinary paper map, the legend describes graphic symbology used to represent the mapped information. Analogously in KGIS, a dynamic legend describes the symbology used to render the information displayed in map form on the graphics screen. This legend provides a mechanism for specifying how query results are graphically displayed. A default display format is maintained for each layer, e.g. parcels or soils, in the data base. This format provides a complete description of how a feature is to be displayed and it is referred to whenever a query involving graphics is executed. The legend is dynamic because, unlike a paper map, information can be added to or removed from the display and as this happens, the descriptions in the legend change correspondingly.

Spatial Criteria For Data Retrieval

A geographic feature differs from other data base entities in having spatial attributes such as size, shape and location, which depend only on the individual feature, and spatial relationships to other geographic features, such as proximity, adjacency and direction.

In addition to MAP, spatial attributes that are currently supported include AREA, PERIMETER and LENGTH. They appear to the user to be stored explicitly in the data base. In actuality, because of their dependency on a feature's positional information, these attributes are computed when they are requested. They are treated in the same syntactic context as other attributes. They may be retrieved along with a list of other attributes or used in selection constraints as the following example shows:

```
SELECT ID,MAP,PERIMETER,OWNERNAME FROM PARCELS WHERE
```

```
AREA > 10;
```

Spatial relationships between geographic features are much less tractable, often involving fuzzy or application-dependent definitions. Peuquet (1985) has stated that all spatial relationships appear to be derivable from three primitives: boolean set operations, distance and direction. Of these, direction is the least useful because a model for direction, free from dependency on human interpretation, has not been developed. As a result, we have focused our efforts to date on spatial relationships that can be derived from boolean operations and distance. Each of these imply, in a sense, a relational join operation, i.e. a spatial join. These spatial relationships relate two separate groups of features, or themes, over a shared domain, namely location is space. In certain cases the join criteria could be made explicit in terms of shared topology. We decided that this would overburden the user and opted instead to implement spatial relationships as high-level functions to better express a users intuitive understanding. All the spatial relationships we currently support fall into two classes: those that act like attributes and those that act like predicates. Attribute-like relationships include DISTANCE and OVERLAP. Predicate-like relationships include OVERLAY and ADJACENT.

DISTANCE is implemented as a scalar function of two themes. It expresses the minimum distance from a feature in the first theme to one in the second. It can be used either as an attribute or a selection criteria as follows:

```
SELECT class, depth, map, distance ( soils, roads )
FROM soils,roads
WHERE distance ( soils, roads ) < 500:meters and
      roads.surface = 'Paved';
```

This query returns several soil attributes, including the map and distance to the nearest road, for soils which occur within 500 meters of a paved road.

OVERLAY is a boolean function of two themes. Stated as a predicate, it expresses the spatial intersection of a feature in the first theme with one in the second, e.g. polygon-polygon, line-line, point-in-polygon, line-in-polygon, etc. It is used as a selection criteria as follows:

```
SELECT class, cec, permeability, soils.map
FROM soils, parcels
WHERE valuation > 60000 and overlay(soils, parcels);
```

This query returns information, including graphics, about soils which occur on parcels valued above $60,000 dollars. The topologic data structures, used in KGIS to represent geographic features, are built at the time the data are added to the data base and alleviate the need to perform polygon intersection computations at query

time. Instead, the OVERLAY operation is reduced to identifying shared topology.

Often, with queries involving the OVERLAY relationship, information relative to the overlapping portion is required. Queries of this kind can be expressed using the OVERLAP modifier. OVERLAP is a scalar function of spatial attributes, used in a query involving the OVERLAY relationship to express attributes of the overlap. For example,

```
SELECT r.id, overlap(r.length), overlap(r.map)
FROM townships t, roads r
WHERE t.name = 'Hampden' and maintenance = 'State'
      and overlay (roads,townships);
```

returns the id, surfacing material, length and map of State-maintained roads that pass through the township of Hampden. The OVERLAP function returns only that portion of the length and map which falls within Hampden.

ADJACENT is a boolean function of two themes. Stated as a predicate, it expresses whether the boundaries of two geographic features share a topologic 1-cell, referred to in KGIS as an edge. This relationship can exist between two polygons or between a line and a polygon. It is used as a selection criteria as follows:

```
SELECT id, address, valuation
FROM parcels,roads
WHERE adjacent( roads, parcels) and
      roads.id = 'Elm Street';
```

This query returns information about parcels which are adjacent to Elm Street. As with the OVERLAY function, execution of the ADJACENT function consists of identifying shared topology between pairs of features.

Geographic Context

The locational data for geographic features is maintained in a single, seamless data base, not partitioned into pre-defined map sheets or their equivalent. If a user does not wish to query against the full geographic extent of the data base, a geographic context may be established. In keeping with the relational model, we refer to this geographic context as a GEOVIEW. A GEOVIEW effectively partitions the data base spatially so that only those geographic features that fall within the specified area are considered for retrieval. A GEOVIEW may use a geographic feature or an arbitrary ground window. A GEOVIEW is specified using a SET command as follows:

```
SET geoview WHERE counties.id = 'PENOBSCOT'
                or
SET geoview WHERE lowerleft = utm(500000,3894000) and
                  upperright = utm(520000,3900000)
```

Once established, a GEOVIEW remains in effect for subsequent queries until changed or reset.

Graphical Context

To visually interpret query results displayed on a graphics screen, it is often necessary to supplement those results with background information. A base map provides a graphical context from which to interpret thematic information. KGIS provides facilities for defining a base map, displayed on the graphics screen, which persists from query to query until modified or removed. A base map is defined with the DISPLAY command as follows:

```
DISPLAY roads, parcels, lakes, streams
```

This command displays a map of all roads, parcels, lakes and streams using the default display formats within the currently defined GEOVIEW. A base map can be modified or reset using the REMOVE command:

```
REMOVE parcels
    or
REMOVE *
```

These commands remove just the parcels or the entire base map, respectively. The contents of the base map are recorded in the dynamic legend along with other graphic query results.

Graphical Input

A special graphical input facility is available for specifying a spatial constraint. The user may specify a feature by pointing at a location on the graphic screen with a pointing device, such as a mouse. The feature specified need not be displayed at the time. This facility is employed as follows:

```
SELECT * FROM parcels WHERE location = mouse;
```

When this query is submitted, the graphic cursor appears on the graphics screen. The user may move the cursor, via the mouse, to a desired location on the screen and indicate a selection by pressing the left mouse button. The user can repeat this until all the selections have been made. The process is terminated by pressing the right mouse button. The query processing then proceeds using the indicated set of features.

CONCLUSIONS

I have presented here the basis of a query language for managing geographic information. The language treats spatial and non-spatial attributes in a single context, providing a high-level relational view of geographic

features. Facilities are provided for the graphical display and input of data, the specification of both graphical and geographical context and the use of spatial attributes and relationships as selection constraints. No claims of completeness are made, however. The language currently provides facilities for some of the more common spatial relationships. As facilities for other relationships are added, the language will continue to evolve.

REFERENCES

Abel, D.J. and Smith, J.L.,1985, A RELATIONAL GIS DATABASE ACCOMMODATING INDEPENDENT PARTITIONINGS OF THE REGION, International Symposium on Spatial Data Handling, Seattle, WA, pp 213-224.

Chamberlain, D.D., M.M. Astrahan, K.P. Eswaran, P.P. Griffiths, R.A. Lorie, J.W. Mehl, P. Reisner and B.W. Wade, 1976. "SEQUEL 2: A Unified Approach to Data Definition, Manipulation, and Control", IBM Journal of Research and Development, 20(6), pp. 560-575.

Dueker, K.J., 1985, GEOGRAPHIC INFORMATION SYSTEMS: TOWARD A GEO-RELATIONAL STRUCTURE, Auto-Carto 7 Proceedings, Washington,D.C., pp 172-175

Frank, A.F., 1982, MAPQUERY: DATA BASE QUERY LANGUAGE FOR RETRIEVAL OF GEOMETRIC DATA AND THEIR GRAPHICAL REPRESENTATION, SIGGRAPH Conference, Boston, MA, Computer Graphics Vol. 16, No. 3, p 199.

_____, 1984, REQUIREMENTS FOR DATABASE SYSTEMS SUITABLE TO MANAGE LARGE SPATIAL DATABASES, International Symposium on Spatial Data Handling Proceedings, Zurich, Switzerland

_____, 1986, PANDA: AN OBJECT-ORIENTED PASCAL NETWORK DATABASE MANAGEMENT SYSTEM, Report No. 57, Department of Civil Engineering, University of Maine, 103 Boardman Hall, Orono, Maine 04469

Keating, T., Phillips, W. and Ingram, K.J., in press 1987, AN INTEGERATED TOPOLOGIC DATABASE DESIGN FOR GEOGRAPHIC INFORMATION SYSTEMS, Photogrammetric Engineering and Remote Sensing

Morehouse, S., 1985, ARC/INFO: A GEO-RELATIONAL MODEL FOR SPATIAL INFORMATION, Auto-Carto 7 Proceedings, Washington,D.C., pp 388-397

Peuquet, D.J., 1984, DATA STRUCTURES FOR A KNOWLEDGE-BASED GEOGRAPHIC INFORMATION SYSTEM, Proceedings, First International Symposium on Spatial Data Handling, Zurich, Switzerland, Geographical Institute, University of Zurich, pp 372-391

_____, 1985, THE USE OF SPATIAL RELATIONSHIPS TO AID DATABASE RETRIEVAL, International Symposium on Spatial Data Handling, Seattle, WA, pp 459-471.

Waugh, T.C. and Healey, R.G., 1985, THE GEOVIEW DESIGN: A RELATIONAL DATABASE APPROACH TO GEOGRAPHICAL DATA HANDLING, International Symposium on Spatial Data Handling, Seattle, WA, pp 193-212