SPATIAL DATA MODELS IN CURRENT COMMERCIAL RDBMS

Matthew McGranaghan

Associate Professor
University of Hawaii
Geography Department
2424 Maile Way
Honolulu, HI 96822 USA
matt@hawaii.edu

ABSTRACT

Vendors of database software are adding spatial capabilities to their products. This paper compares the spatial data types and the spatial relations which are recognized by a set of current commercial RDBMS and layered spatial database management packages. While there is convergence on the notion that databases should handle spatial information, there are both gross and subtle differences in what this means. A range of types, relations, capability, and ease of use is evident.

INTRODUCTION

Developments in database technology enable change in the practice of automated cartography and GIS, but impose limits through the data models they support. This paper describes the spatial data models that several current vendors of extended relational database management systems implement. It compares the richness of the data types, relations, and operators that each of the several vendors now offer. Minimally, these need to support an applications needs. More ambitiously, they might be hoped to support interoperability of heterogeneous spatial databases. However, the range of differences, both in substance and in nomenclature may inhibit this.

The relational model (Codd 1970) and relational databases have proven extremely useful for many applications, but, only a few years ago, many in GIS thought that spatial data were poor candidates for processing in relational database management systems (RDBMS). More recently, it has become apparent that extensible relational and object-oriented databases can meld spatial and mainstream databases (Strickland 1994, Abel 1996). van Oosterom (1993)

describes three approaches to incorporating spatial data in a relational database. He refers to them as dual architecture, layered architecture, and integrated architecture. We will not consider dual architecture further.

The layered architecture approach provides a layer of abstraction, in practice a set of relational tables containing the spatial information, between the user and the kernel of a standard RDBMS, which allows the user to think in terms of a spatial model but then translates this into data types which are native to the RDBMS. van Oosterom cites System 9, GEOVIEW, and CSIRO-DBMS (Abel 1989) as examples. ESRI's SDE is another.

An integrated architecture is one in which the types of data in the database are extended to include spatial data types. van Oosterom sites Intergraph's TIGRIS, and several research systems, including his own GEO++ (van Oosterom and Vijlgrief 1991), based on Stonebraker's Postgress. Strickland (1994) suggests that this ability to extend the relational model has allowed it to subsume object-oriented database functions, and the resulting object-relational model of extended RDBMS seems poised to carry the current vendors forward.

Mainstream commercial RDBMS vendors have also begun to offer spatial data handling capabilities. However, the claim to have "spatial capabilities" is vague. Vendors have taken different approaches, developing different data types and recognizing different relationships among spatial entities in these extended models. This paper is an attempt to make a *pima facia* comparison of the spatial data models, spatial operators, and richness of the application programmer's interface (API) provided in each of the major RDBMS' current release, based on sales literature, product documentation, published descriptions, and inquiry. It does not reflect hands-on experience or experimentation with the software. It certainly may reflect misconceptions that more familiarity would correct. The objective is to illuminate differences and similarities among the systems' spatial capabilities. System administration, management, and security issues are ignored.

DESCRIPTIONS

The following sections describe the objects, relations, and functions in each product. The descriptions are grouped by system architecture.

Layered Architecture Systems

Some information was available regarding Sybase Spatial Query Server, ESRI's SDE, and CSIRO-SDM (following SIRO-DBMS), all of which use a layered architecture.

SIRO-DBMS (Abel 1989), implements spatial objects and an extended SQL interface that recognizes spatial relations among them. This tool-box is built as a lean layer upon which developers can build custom systems. The data model is very similar to the U.S. Spatial Data Transfer Standard (SDTS) data model. It includes: point, rectangle, simple line segment, line string, link, directed link, chain, complete chain, area chain, network chain, ring, and both simple and

complex polygon types). Topology is carried in relations of entities and components. Points can be represented in relational tables. More complex features can be built up from them, or have coordinates represented directly in BLOB fields. It implements eight spatial operators as database search qualifiers: mbrint, intersects, encloses, is_enclosed_by, crosses, is_connected_to, within_buffer, and adjacent_to. Both the spatial model and operations are very well chosen for GIS applications. They are accessed via a C API of approximately two dozen functions. Quadtree-based spatial indexing is used to enhance performance, and comparisons with Oracle's OMD indicate that SDM is faster (Zhou 1995).

Sybase has spatial facilities through Spatial Query Server, a product from Vision International, a division of Autometric Inc. The Spatial Query Server (SQS) supports the definition of spatial data types spatial operators, and a spatial indexing schema, within a Sybase SQL Server. The eleven supported Spatial Data Types (SDT) are: point, rectangle, circle, ellipse, azimuth, line, polygon gpolygon (polygons nested within another polygon), voxel, polygon_set, and rectangle_set. The spatial qualifier operators are: intersect, inside, outside, beyond, and within. Spatial queries are processed against templates defining a geometric shape (i.e., rectangle, ellipse, circle, point, line, polygon, or gpolygon). A spatial index may be declared on a column of an SQS data type.

ESRI's Spatial Data Engine (SDE) sits atop an Oracle (or other) RDBMS but builds its own spatial layer eschewing Oracle's spatial data facilities. Reputedly, for better performance. SDE's spatial types include: point, point-cluster, spaghetti line, non-self-crossing line string, ring, polygon, and donut polygon. Polygon nesting beyond a "doughnut hole" requires the definition of separate (perhaps also "doughnut") polygons on down to the least enclosed polygon. Spatial indexing is through a three-tiered partition of the user coordinate system set up at the time of database creation. The API consists of 138 functions to enter, manage, edit, query, and annotate spatial and related attribute data. The SE_RELATION() function generates a bit-mask indicating which of the following eight spatial relations hold among two objects (line intersection, point in common, common boundary in same order, common boundary in reversed order, spatially identical features, area intersect (at least one feature is an area and the other is at least partially inside it), primary feature contained by secondary feature, and secondary feature contained by the primary feature).

Integrated Architecture Systems

Oracle's Spatial Data Option, CA-OpenIngress' Object Management Extension (OME) and Spatial Object Library (SOL), and Informix's Illustra 2D and 3D Spatial DataBlades are examples of commercial integrated architecture spatial relational databases. Some information was available for each of these.

Oracle's Spatial Data Option (formerly Oracle7 MultiDimension) claims support for three types of objects (point, line, polygon) and three spatial relations in promotional literature but the reference manual describes a more complex two

tiered system of access. First, bins of data on retrieved based on five relations (enclose, enclosed-by, overlap, equal_to, outside) between bins and three types of query window (range, proximity, and polygon). Second, data points stand in one of three relations (inside, outside, on_the_border) of the query window, while line segments can also overlap the query window (Oracle 1996).

Oracle's basic spatial model is that of a point in an up-to-32-dimensional space. The HHCODE represents a linearly-indexed bounded cell in the multi-dimensional data space. The range query retrieves HHCODES falling in an n-dimensional minimum bounding rectangle. The proximity query returns HHCODES within a radius of a point, with the assumption that each dimension is scaled the same. While one suspects what the polygon query should return, it is not really clear. The opaqueness of "a polygon window is defined by specifying a start and end point for each node, in two dimensions, up to a maximum of 124 nodes" (Oracle 1996, p. 3-7) is daunting. Predictably, the API is more database- than space-oriented.

CA-OpenIngress with the Object Management Extension (OME) and Spatial Object Library (SOL) implements geographic data types and geometric SQL functions. SOL comes from a partnership with Mosaix Technologies Ltd, and "provides a rich set of library elements for application development. Data involving spatial relationships can be handled by the database in the same manner as the more traditional data types of characters and numbers. Using these spatial shapes and functions, location data can be integrated easily into business applications." Details on the relations and entities, however, are not provided in the literature that has been examined to date.

Illustra Information Technologies Inc. (since December 1995 a subsidiary of Informix Software Inc.) calls its Illustra Server a "dynamic content management system". The 2D Spatial DataBlade Module supports ten spatial types: circle, directed graph, ellipse, line segment, path, point, polygon, polygon set, quadrangle, and square/rectangle (Informix 1994, p. 2-1). Coordinates are double precision. (Paths are allowed to be self-crossing. Polygon sets allow nested and disjoint polygons via explicit parent-child enclosure declarations.) Four functions (insert, update, copy, and micopy) convert external (string) representations to internal C structure representations. The select function returns a string representations of objects. While promotional literature indicates that 2D Spatial DataBlade provides "over 200 functions" to create, compare, manipulate, and query spatial objects, this count is generated by overloading fifty-seven operators to handle multiple object-types as function arguments. For instance, the Boolean operator *overlap*, can be called to compare objects of any type(s). The operators return the range of standard as well as spatial data types.

The 3D Spatial DataBlade Module has eighteen 3D data types, (including point, box3d, quadrangle, circle, ellipse, line segment, path, polygon, polygon set, vector, unit vector, circular arc, rectangle, polyline/polyarc, polycurve, polygon mesh, polygon surface, and polyface mesh). While promotional materials claim "over 1,000" functions, these are in fact arranged as sixty-four

overloaded function calls. Key relations (location, distance, overlap) are incorporated in the database and accessible via SQL and a C programming API. R-TREE spatial indexing is used as are "smart objects" in which the system decides whether to store coordinates for small objects in a relation or as a "large object".

COMPARISON

The following table summarizes the counts of data types and spatial relations described above, attempting to penetrate marketing rhetoric. It appears that the number of spatial data types ranges from 2 up to 18. This would seem to reflect an ultimate foundation of point-based vector representation, coupled with some differences in the level of abstraction and integration with which the system is designed to work. The number of relations among objects in the data base appears to range between 3 and 8 (or ?). There is much more variation in the API sizes, which range from on the order of three up to 138. This range reflects more the intended uses of products and the number of spatial relations recognized in queries, and the number of functions in the application programmers interface (API) as indicators of richness or expressiveness of the representation and the system. The size of the API may also indicate how complex it is to use. The table shows that there are differences, but masks what they are.

System	Data Types	Relations	API size
Sybase SQS	11	5	?
ESRI SDE	7	8	138
SIRO-DBMS	12	8	~30
Oracle SDO	2	3	3
CA SOL	?	?	?
Informix 2D SDB	10	4	57
Informix 3D SDB	18	4	64

The following sections attempt to more precisely compare the features of these systems. The effort is made more difficult by differences in nomenclature and the amounts of information that were available for each system.

Data Types

The following table indicates, for comparison, the spatial data types in each system. It raises several difficulties with the information available from vendors. These include differences in nomenclature, such as differentiating among several meanings of "complex polygon", and differences in levels of generality. For instance, identifying a "quadrilateral", or for that matter a "rectangle" type in addition to the more general "polygon" in literature describing a system, without also indicating how and whether there really is specialization, obfuscates rather than clarifies system capabilities --- one might as usefully list pentagons,

hexagons, etc. as types. Because of the limited treatment of 3D data in most of these systems, and for the sake of brevity, the table leaves out many of the 3D types in 3D Spatial DataBlade. Exclusion of a type from this list does not mean that it can not be implemented in a given system, only that it is not mentioned as a type in the system's data model. In any event, entries in the table are my best guess at native types from the information I have available.

Types	SQS	SIRO-DBMS	SDE	SDO	2D-SDB	3D-SDB
Point	Y	Y	Y	Y	Y	Y
Point Cluster	N	N	Y	N	N	N
Line Segment	N	Y	Y	Y	Y	Y
polyline	N	Y	N	Y	Y	Y
Ring	N	Y	N	Y	Y	Y
Topological Arc	N	Y	N	N	N	N
Simple Polygon	Y	Y	Y	Y	Y	Y
Complex Polygon	Y	N	N	N	Y	Y
Donut Polygon	N	Y	Y	N	Y	Y
Nested Polygon	Y	N	N	N	Y	Y
Circle	Y	N	N	Y	Y	Y
Ellipse	Y	N	N	N	Y	Y
Rectangle	Y	Y	Y	N	Y	Y
Rectangle Set	Y	N	N	N	N	N
Quadrilateral	Y	N	N	N	Y	N
Graph Network	Y	N	Y	N	Y	N
Layer	N	N	Y	N	N	Y
Azimuth	Y	N	N	N	Y	Y
Voxel	Y	N	N	N	N	Y

Only SIRO-DBMS seems to consciously support the topological arc notion that is at the core of the US spatial data infrastructure. It is not clear whether this signals anything. Perhaps it signals a general pulling back from data models that have supported much analytic cartography and GIS. Perhaps it signals realization that similar information can be recovered in reasonable time from other types. Or perhaps it signals unfamiliarity with existing spatial data processing techniques and / or a continuing model of separation of database from spatial data processing.

Relations

Comparing these systems on the relations that they understand also proved elusive. All of the systems provide some ability to extract data based on spatial relations, essentially allowing spatial relations among objects to become part of the qualifying predicate in a database selection operation.

The table below indicates the relations that each system recognizes. It appears that these systems are very well matched in this regard, however, from the system documentation, one once again gets the sense that there are

differences in meaning among interpretation of these concepts. One difference is in the types of data returned by queries on a given relation.

Operation	SQS	SIRO-DBMS	SDE	SDO	2D-SDB	3D-SDB
MBR Intersect	N	Y	N	N	N	N
Object Intersect	Y	Y	Y	Y	Y	Y
Enclosure	Y	Y	Y	Y	Y	Y
Proximity/Buffer	N	Y	N	N	N	N
Contiguity	N	Y	Y	N	N	N
Spatial Equality	N	N	Y	N	Y	Y
Exclosure	Y	N	N	Y	N	N

The APIs and Operations

Comparing the number of operations that each system can perform on data to produce new information reveals a bit about system orientation. Several of the systems allow what one might consider more traditional GIS, or other application, capabilities than simple data retrieval so that in addition to treating relations as spatial qualifiers on retrieval, they can also return newly computed spatial objects or measurements that result from performing a spatial operation on qualifying objects. This capability makes a system seem more like a GIS programming environment than a database interface, and indicates further migration from a dual model of separating spatial and attribute data. Drawing a distinction between what is retrieval and what requires computation of new spatial objects or information in not easy. The following table attempts to characterize each system by the number of functions it offers, in categories structured after Roger Tomlinson's list of seventy-two GIS functions.

Function Class	SQS	SIRO-DBMS	SDE	SDO	2D-SDB	3D-SDB
Input, Edit, Convert	?	1	2	0	2	2
Overlay	?	1	3	1	3	3
Buffer/Corridor	?	1	1	0	0	0
Display	?	0	0	0	0	0
Elevation modeling	?	0	0	0	0	3
Other	?	7	14	5	14	13

From this point of view, SDE is very GIS-like at the outset and the 2D and 3D Spatial DataBlades have considerable analytic geometric capability. Oracle SDO and SIRO-DBMS both seem more oriented toward accessing data for use by an application program.

CONCLUSIONS

From the comparison, a few general conclusions can be drawn. Vendors have recognized that spatial data are worth supporting, and their attention to this is paves the way for more GIS and cartographic use of RDBMS. Data access will be increasingly easy for even very large databases.

The distinction between GIS and database software is blurred by the spatial capabilities of current RDBMS. These range from bare data item retrieval to fairly full analytic geometric manipulation of spatial data. Products with large libraries seem to provide many GIS facilities. Expansion in this direction seems likely. It is not yet clear what this will mean for GIS development. The potential for GIS to be subsumed within database technology in the not too distant future seems real, but systems to date lack much in the way of data input and spatial co-registration.

The ranges of variation among the data types and operations are noteworthy. While progress has been made toward standardization in the spatial data community, and there is considerable convergence among the models in these systems, the differences are impediments to adoption and to interoperability. It is not yet clear which of these sets of spatial types and relations will prove to be "most adequate". Public benchmarks showing performance on a suite of common GIS tasks under each data model would be interesting. So would demonstrations of the effort required to move between these data models.

The problems in trying to make information about these systems commensurate, and the concomitant limitations of the present paper should be recognized. Differences in nomenclature and nuance make description-based comparison suspect. This work was conducted without side-by-side access to test these products experimentally. Additionally, firms have considerable financial interest in the reputations of their products, and some consider detailed descriptions of their capabilities to be proprietary information. The cooperation of several firms in providing demonstrations and brief access, or even simply being willing to sell documentation beyond sales literature to non-licensees of their software is appreciated.

REFERENCES

Abel, D.J. (1989) SIRO-DBMS: a Database Tool Kit for Geographical Information Systems. *International Journal of Geographical Information Systems*, v3, n2, p103-116, 1989.

Abel, D.J. (1996) What's Special about Spatial?, *Proceedings of the 7th Australasian Database Conference*, Melbourne, Australia, January 29-30, 1996.

CA-OpenIngres, Object Management Extension (OME) and Spatial Object Library (SOL) sales literature.

Codd, E. (1970) A Relational Model for Large Shared Data Banks. *Communications of the Association for Computing Machinery*, v 13, n 6, p 377-387.

Informix (1994) *Illustra 2D Spatial DataBlade Guide (Release 1.3),* Illustra Information Technologies, Inc., Oakland CA, October 1994.

Informix (1994) *Illustra 3D Spatial DataBlade Guide (Release 1.2),* Illustra Information Technologies, Inc., Oakland CA, October 1994.

Milne, P., S. Milton, and J.L. Smith (1993) Geographical object-oriented databases --- a case study. *International Journal of Geographical Information Systems*, v. 7, n. 1, p. 39-55

Oracle (1996) *Oracle7 Spatial Data Option Reference and Administrator's Guide (Version 7.3.2),* Oracle Corporation, Redwood City, CA, April 1996.

ESRI (1995) *SDE The Spatial Data Engine User's Guide, version 2.0.* Environmental Systems Research Institute, Redlands, CA.

Strickland, T.M. (1994) Intersection of Relational and Object, *AM/FM International Proceedings.* p. 69-75.

van Oosterom, P.J.M. (1993) *Reactive Data Structures for Geographic Information Systems.* Oxford University Press, New York.

Viljlbrief, T. and P van Oosterom, The GEO++ System: an Extensible GIS, *Proceedings Fifth International Symposium on Spatial Data Handling*, p40-50, Charleston, South Carolina, August 1992.

Zhou, X. (1995) A Comparison of Oracle7 MultiDimension and ARC SDM for Point Data Retrieval.