3D Data Modelling in a Geo-DBMS

Jantien Stoter and Peter van Oosterom
Department of Geodesy, Delft University of Technology
P.O. BOX 5030, 2600 GA, Delft, the Netherlands
{j.e.stoter|oosterom}@citg.tudelft.nl

In this paper a number of alternative DBMS data models for 3D objects are presented. The models range from a non topologically structured model based on 3D geometric data types to a fully topologically structured model (with some variants in between). The alternatives are evaluated in the context of a 3D cadastre as an example application in which parcels are vertically subdivided. The evaluation also takes into account current and future possibilities to implement the models in a DBMS.

Introduction

Recent important developments in GISs are the increasing interest in 3D or even 4D as well as the growing importance of having a DBMS for maintaining both spatial and non-spatial data in one integrated environment. A relevant question is therefore to look if and how 3D geo-data (both geometry and topology) can be maintained in DBMSs.

Mainstream DBMSs have implemented spatial data types and spatial functions more or less similar to the OpenGIS Consortium Simple Features Specification for SQL. In most DBMSs z-values can be used to represent data types in 3D (points, lines, polygons), although this z-value is not used in the spatial functions (area, distance). 3D volumetric data types are however not supported in DBMSs. In this article we describe how 3D spatial objects can be maintained in currently available DBMSs.

In this research we use the DBMS of the Netherlands's Kadaster containing cadastral parcels as starting point and extend this DBMS to support 3D spatial objects. For the implementation Oracle Spatial 9i is used.

In case of constructions on top of each other, the parcels need to be extended and divided vertically in 3D. For example, we have a parcel with a railway station, a bus and tram station on top of it and a metro station below it, all owned by different owners (figure 1). These properties are located on top of each other and have to be defined in 3D.

Figure 1: Building complex in The Hague: example of a 3D situation.

To extend the parcels to 3D a table input_3D is created, that contains for every parcel the different height-levels of ownership (z_list). The z-list contains n z-values corresponding to n-1 consecutive ranges associated with the parcel. The vertical extents of the rights of the parcel containing the tram and bus, railway and metro station (parcel '13295'), are as follows:

- metro station: –25 to –1 m below surface
- railway station: –1 to 6 m
- tram/bus station: 6m to 12 m

```
SQL> select * from input_3d;

MUNICIP OSECTION     PARCEL   Z_LIST
------- --------     ------   ----------
GVH12        R       12131    Z_ARRAY(0, 12, 40)
GVH12        R       13290    Z_ARRAY(0, 12)
GVH12        R       13288    Z_ARRAY(0, 12)
GVH12        R       13289    Z_ARRAY(0, 12)
GVH12        R       13294    Z_ARRAY(0, 3, 12)
GVH12        R       13291    Z_ARRAY(0, 3, 12)
GVH12        R       13293    Z_ARRAY(0, 3, 12)
GVH12        R       13292    Z_ARRAY(0, 3, 12)
GVH12        R       13295    Z_ARRAY(-25, -1, 6, 12)
```

Implementation of 3D geo-data in the DBMS within current techniques

Based on the table input_3d and the parcels a 3D representation of the rights was generated in two ways:

- using 3D spatial data types with two variants: a. a set of records with 3D polygons and b. one record with a 3D multipolygon
- using a full topological model

The 3D representation in both ways consists conceptually of polyhedrons (body with flat faces). The parcelboundary in 2D is the footprint of the representation.

Using 3D spatial data types

A pl/sql script has been written to generate the 3D spatial objects using the 3D polygon primitive. Every 3D spatial object is defined as an object defined by a set of records representing a polyhedron with references to the (flat) faces it consists of. The faces are stored as 3D polygons. This model is partly a topological model, since the body is defined by references to the faces and the faces can be shared by neighbour-bodies. The generated table looks as follows:

```
SQL> select body_id,face_id,shape from rightobject3d where body_id=1;

   BODY_ID    FACE_ID
---------- ----------
SHAPE(SDO_GTYPE, SDO_SRID, SDO_POINT(X, Y, Z), SDO_ELEM_INFO, SDO_ORDINATES)
--------------------------------------------------------------------------------
         1          1                          --one complex object (body_id=1)
SDO_GEOMETRY(3003,                            --upper face of the body
    NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1), SDO_ORDINATE_ARRAY(
                                              --3003 indicates a 3D polygon
82220.96, 455098.11, -25, 82221.36, 455098.44, -25, 82232.88, 455106.96, -25,
82238.93, 455099.08, -25, 82242.01, 455101.41, -25, 82237.26, 455107.61, -25,
82247.5, 455115.01, -25, 82253.82, 455106.78, -25, 82256.67, 455108.99, -25,
.....
82220.96, 455098.11, -25))
         1          2                          --1st face in between
SDO_GEOMETRY(3003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1), SDO_ORDINATE_ARR
AY(82220.96, 455098.11, -25, 82220.96, 455098.11, -1, 82221.36, 455098.44, -1,
82221.36, 455098.44, -25, 82220.96, 455098.11, -25))
         1          3                          --2nd face in between
SDO_GEOMETRY(3003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1), SDO_ORDINATE_ARR
AY(82221.36, 455098.44, -25, 82221.36, 455098.44, -1, 82232.88, 455106.96, -1, 8
2232.88, 455106.96, -25, 82221.36, 455098.44, -25))
etc.....
```

Another way to use the currently available 3D spatial data types is to make use of a multipolygon. This has also been implemented and the resulting table looks as follow:

```
SQL> select body_id,shape from rightobject3d_multipol where body_id=1;

BODY_ID    SHAPE(SDO_GTYPE, SDO_SRID, SDO_POINT(X, Y, ),SDO_ELEM_INFO,SDO_ORDINATES)
--------   ------------------------------------------------------------
         1
SDO_GEOMETRY(3007,                                    --3007 indicates a 3D multipolygon
NULL, NULL,
SDO_ELEM_INFO_ARRAY(1, 1003, 1, 109, 1003, 1, 124  --multipolygon consists of several simple
polygons (1003,1) ; for every simple polygon the offset in the coordinate array is given
, 1003, 1, 139, 1003, 1, 154, 1003, 1, 169, 1003, 1, 184, 1003, 1, 199, 1003, 1,
.....
574, 1003, 1, 589, 1003, 1, 604, 1003, 1, 619, 1003, 1, 634, 1003, 1),
SDO_ORDINATE_ARRAY(
82220.96, 455098.11, -25, 82221.36, 455098.44, -25, 82232.88, 455106.96, -25,
82238.93, 455099.08, -25, 82242.01, 455101.41, -25, 82237.26, 455107.61, -25,
82247.5, 455115.01, -25, 82253.82, 455106.78, -25, 82256.67, 455108.99, -25,
.....
82086.74, 455275.92, -25, 82117.4, 455236.1, -25, 82115.65, 455234.76, -25,
82136.77, 455207.36, -25, 82178.4, 455153.55, -25, 82220.96, 455098.11, -25,
                                                      --end of 1st polygon
82220.96, 455098.11, -25, 82220.96, 455098.11, -1, 82221.36, 455098.44, -1,
82221.36,455098.44, -25, 82220.96, 455098.11, -25, --end of 2nd polygon
82221.36, 455098.44, -25, 82221.36,455098.44, -1, 82232.88, 455106.96, -1, 82232.88,
455106.96, -25, 82221.36, 455098.44, -25,          --end of 3rd polygon
82232.88, 455106.96, -25, 82232.88, 455106.96, -1, 82238.93, 455099.08, -1,
82238.93, 455099.08, -25, 82232.88, 455106.96, -25, etc.....
```

With this solution it is possible to retrieve 2D topological relationships between (the projection of) 3D objects and surface parcels. Once 3D spatial functions are available, also 3D relationships can be retrieved. Another advantage is that 3D (multi)polygons are recognised by GIS/CAD applications that can make a database connection, by which it is possible to visualise (and edit) the data in a GIS or CAD (figure 2).

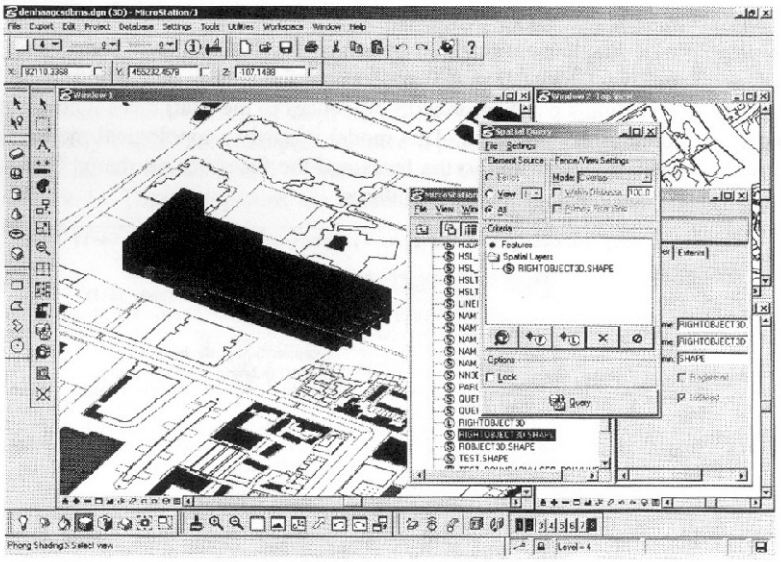

Figure 2: Visualising 3D spatial objects stored in Oracle with MicroStation Geographics. It is the 3D representation of figure 1. The largest parcel is the parcel with the metro, train and tram/bus station.

An additional advantage of the 3D multipolygon approach is the one-to-one correspondence between a record and an object. Disadvantage of these approaches is that the topology structure between objects cannot be used, which implies that there is redundant storage of edges (and in the 3D multipolygon solution also of faces).

Using a full topological model

DBMSs do not (yet) support topology (2D nor 3D). Therefore, a full topological model has to be defined in a DBMS by means of user-defined references. In this research we use the Simplified Spatial Model of Zlatanova. A 3D geometry object is therein defined as a polyhedron consisting of nodes and faces. The model consists of 3 tables: BODY, FACE and NODE. Each table contains references to other tables: the BODY-table contains references to the faces and the FACE-table contains references to the nodes (with their co-ordinates).

A pl/sql script was written to generate the BODY, FACE and NODE table:

```
SQL> select * from body order by bid,seqf;

    BID         FID        SEQF
---------- ---------- ----------
     1          1          1
     1          2          2
     1          3          3
     1          4          4
     1          5          5
     1          6          6
     1          7          7
     1          8          8
     1          9          9
     1         10         10
     1         11         11
     1         12         12
     1         13         13
     2         13          1
     2         14          2
     2         15          3
 etc.....

SQL> select * from face order by fid,seqn;

FID         NID        SEQN
---------- ---------- ----------
     1          1          1
     1          2          2
     1          3          3
     1          4          4
     1          5          5
     1          6          6
     1          7          7
     1          8          8
     1          9          9
     1         10         10
     1         11         11
     1          1         12
     2          1          1
     2         12          2
     2         13          3
     2          2          4
     2          1          5
     3          2          1
     3         13          2
     3         14          3
     3          3          4
     3          2          5
etc.....

SQL> select * from node;

NID           XYZ_LIST
------------  ----------------
      1              XYZ_LIST_T(82220.98, 455098.11, -25)
      2              XYZ_LIST_T(82221.36, 455098.44, -25)
      3              XYZ_LIST_T(82232.88, 455106.96, -25)
.....etc
```

Note that SEQF (sequence face) has no use (at all) and also note that faces are shared between bodies as it should be in a full topological model (e.g. face 13 is shared between body 1 and body 2) and nodes are shared between faces (e.g. node 2 is shared between face 1, 2 and 3).

Boundary faces within one parcel are stored once (horizontal planes); the vertical faces should also be stored non-redundantly. This is currently not the situation but could be solved by some post processing after the parcel-per-parcel based conversion from the 2D to the 3D model. Advantage of this approach is that topology structure management can be used in the storage and retrieval of the data. For example, by means of the shared (horizontal) faces one can easily find the upper and lower neighbours of a spatial object.
Disadvantages of using a topological model are:

- the data model needs three tables instead of just one (as in the 3D multipolygon case);
- since the DBMS does not recognise topology, the consistency of the data has to be checked by other software;
- querying can be very difficult at SQL level (topology is not recognised by DBMSs). For geometric queries it is always required to generate a realisation of the object, instead of being able to use the spatial queries available in the DBMS directly. Therefore, a function has been written to realise the geometry of topologically stored 3D spatial objects.

Conclusions

In this paper several 3D data model extensions in a geo-DBMS are presented for a 3D cadastre with more or less topology. Also in this paper we showed the implementations of the presented data models. The main differences in the models are the amount of topology used with the (known) advantages and drawbacks. All models are based on the same conceptual approach of vertical division of a parcel-column. The advantage of this approach is that it fits very well to the current way of thinking within the cadastre with the 2D parcel on the surface as a basis. However, a drawback of this approach is that objects above or below several current 2D parcels (e.g. a tunnel) have to be subdivided in as many 3D bodies as there are parcels on the surface. Future research will focus on this issue.