Drie aanbieders in een benchmark vergeleken

De stand van zaken rond het Geo-DBMS

Peter van Oosterom

In een Geografisch Informatiesysteem (GIS) worden objecten gerepresenteerd die op één of andere manier een relatie hebben met een locatie op aarde. Een GIS bevat dus objecten met een geometrisch kenmerk. De architectuur van het GIS is aan verandering onderhevig: steeds vaker worden systemen gebaseerd op een geïntegreerde architectuur, dat wil zeggen alle geometrische data wordt in een DataBase Management Systeem (DBMS) opgeslagen, gezamenlijk met administratieve data.

De eerste stap bij het realiseren van een Geo-DBMS is het beschikbaar maken van datatypen en operatoren voor de geometrische primitieven, 'simple features' in OpenGIS termen: punt, lijn en polygoon. Deze zijn inmiddels gestandaardiseerd en geïmplementeerd in verschillende (commerciële) DBMS'en. Een vereiste wanneer er met echte datasets wordt gewerkt, is dat er ook ruimtelijke indexering beschikbaar is. Een standaard benchmark voor Geo-DBMS'en ontbreekt nog, daarom presenteert dit artikel een eigen benchmark met een dataset van het Kadaster die uit meer dan 120 miljoen ruimtelijke records bestaat. De resultaten van deze benchmark zullen voor drie DBMS'en worden gegeven: Oracle, Informix en Ingres. Naast de hierboven genoemde 'basis' functionaliteit wordt er ook gewerkt aan een aantal uitbreidingen, zoals het ondersteunen van topologische structuren in het DBMS ('complex features') en 3D geometrische primitieven.

GEÏNTEGREERDE ARCHITECTUUR

Het belang van de geïntegreerde architectuur, om geometrische en administratieve attributen in één DBMS te beheren, is door de industrie onderkend. Het OpenGIS Consortium heeft de basis ruimtelijke typen en functies gestandaardiseerd, in de OpenGIS terminologie heet het de 'Simple Feature Specification' (SFS). De SQL/SFS specificatie zal ook onderdeel uitmaken van de toekomstige ISO SQL3 standaard. Op dit moment zijn er verschillende commerciële DBMS'en beschikbaar met ondersteuning voor ruimtelijke datatypen (sommigen volgens de OpenGIS standaard): Oracle, Informix, Ingres of IBM DB2.

Tevens komen er ook steeds meer commerciële GIS-pakketten die de geïntegreerde architectuur ondersteunen: ESRI ArcGIS, MapInfo Professional, Intergraph GeoMedia Professional of Bentley MicroStation GeoGraphics (zie afbeelding 1). Dit artikel gaat verder in op de volgende onderwerpen: 2D ruimtelijke datatypen, ruimtelijke indexering en clustering, benchmarking, topologie en 3D ruimtelijke datatypen.

2D RUIMTELIJKE DATATYPEN

De abstracte standaarden van OpenGIS en ISO TC211 zijn goed op elkaar afgestemd en definiëren dezelfde geometrische primitieven. In de Engelstalige standaard aangeduid als point, line, surface en solid, in het Nederlands heten ze punt, lijn, oppervlak en lichaam. Voorlopig heeft alleen OpenGIS ook een implementatiespecificatie gedefinieerd. Deze is beschikbaar voor 2D ruimtelijke typen en gespecificeerd voor verschillende platforms: OLE/COM, Corba en SQL.

In meer detail zullen nu de SQL-datatypen beschreven worden. OpenGIS SFS/SQL kent de typen Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon and GeomCollection.

AFBEELDING 1: POLYGON OPGESLAGEN IN ORACLE SPATIAL, WEERGEGEVEN MET BENTLEY MICROSTATION.

De eerste drie typen spreken voor zich. De daarop volgende drie typen omvatten één of meer voorkomens van de gerelateerde basistypen. Het laatste type, GeomCollection, kan elk basis geometrie-type (in elk aantal en combinatie) bevatten. Van de drie geteste DBMS'en is alleen Informix geheel OpenGIS compliant en kent dan ook al deze typen. Oracle ondersteunt al deze geometrie-typen ook, maar verpakt ze in één generiek type met de naam sdo_geometry.

Het voordeel hiervan is dat het type dan gesloten is onder allerlei ruimtelijke operaties, zoals bijvoorbeeld de intersectie. Het nadeel is dat er minder 'scherp' gemodelleerd kan worden. De Ingres ruimtelijk typen zijn van voor het OpenGIS-tijdperk en deze zijn dan ook het meest beperkt (geen gaten in polygonen, geen cirkelbogen, beperkt aantal punten per object, geen geodetische coördinaten). Maar ze hebben als voordeel dat er gekozen kan worden voor representatie van de coördinaten op basis van 8 byte floating point of 4 byte integer-getallen. Hieronder volgt een OpenGIS compliant voorbeeld om een tabel te definiëren (een Kadastrale perceelsgrens) met een LineString geometrie-attribuut en hier vervolgens een record in te voegen.

```
create table xfio_boundary(
 object_id  integer       not null, – Line object
                          Id
 slc        integer       not null, – slc code
 classif    integer       not null, – Object class
                          code
 shape      st_linestring not null, – Line
                          coordinates (2D)
 accu_cd    integer       not null, – Accuracy
                          code
 object_dt  integer       not null, – Measurement,
                          user time
 l_municip  char(5)       , – Left Municipality
                          code
 r_municip  char(5)       , – Right Municipality
                          code
...);

insert into xfio_boundary values (...,
   ST_LineFromText('linestring(((103654574
   460970880,\
   104323607 460885924, 104769627 460885924,
   105523616 461013359,\
   105544856 461395663, 105061624 461741343,
   104089976 461777967,\
   103474041 461639912, 103474041 461162032)',
   128992) ,...);
```

Naast de ruimtelijke typen zelf zijn er ook operatoren op deze typen gedefinieerd. Deze zijn in een aantal categorieën onder te brengen. De eerste categorie operatoren kan een *deel van een ruimtelijk* attribuut selecteren; bijvoorbeeld het beginpunt van een LineString (StartPoint) of informatie over de betreffende attribuutwaarde leveren, zoals het aantal punten in een Polygon (NumPoints) of het ruimtelijk referentiesysteem (SRID). Hiervan definieert de OpenGIS SFS/SQL-specificatie er zo'n kleine 20 die allemaal door Informix worden ondersteund. Ingres biedt hier vijf functies en Oracle slechts twee.

De tweede categorie operatoren bestaat uit functies die de *topologische relatie* ('touch', 'inside', 'overlaps', enzovoort) tussen twee geometrische objecten bepalen. Zowel Oracle als Informix bieden

Een standaard benchmark voor Geo-DBMS'en ontbreekt nog

hier alle negen OpenGIS gedefinieerde operatoren. Ingres heeft ongeveer de helft van de functies beschikbaar.

De volgende categorie operatoren omvat *geometrische berekeningen* op basis van één operand ('area', 'length', 'buffer', 'centroid', enzovoort) of twee operanden (bijvoorbeeld 'distance', 'union', difference'). Ook hier bieden Oracle en Informix de meeste functionaliteit (en Informix dan weer volgens de specificatie van OpenGIS) en Ingres biedt slechts een bescheiden aantal functies. Hieronder staan enkele SQL-voorbeelden, welke de ruimtelijke operatoren gebruiken (volgens OpenGIS-standaard).

```
/* bereken de lengte van de grens met object_id
12345 */
select object_id, ST_Length(shape)
from xfio_boundary
where object_id = 12345;

/* bereken een buffer rond alle grenzen binnenin
zoekgebied */
select ST_buffer(shape) as buffered_shape
from xfio_boundary
where ST_Within(shape, ST_PolyFromText(
'polygon ((78550000 457900000, 78550000
458025000,\
78650000 458025000, 78650000 457900000, 78550000
457900000))',
28992));

/* verzamel statistieken over het aantal punten
per grens */
select count(*), ST_NumPoints(shape)
from xfio_boundary
group by ST_NumPoints(shape);

/* clip (intersection, knip) de grenzen met het
zoek polygon */
select ST_Intersection(shape, ST_PolyFromText(
'polygon ((78550000 457900000, 78550000
458025000,\
```

```
78650000 458025000, 78650000 457900000, 78550000
457900000))',
28992)) as clipped_shape
from xfio_boundary
where ST_Intersects (shape, ST_PolyFromText(
'polygon ((78550000 457900000, 78550000
458025000,\
78650000 458025000, 78650000 457900000, 78550000
457900000))',
28992));
```

Oracle biedt als enige nog een vierde categorie operatoren, *de ruimtelijke aggregaten*. Dit zijn functies die in een SQL aggregatie-constructie gebruikt kunnen worden (net zoals 'count', 'sum', 'min') om een geometrische operatie met alle betreffende records uit te kunnen voeren; bijvoorbeeld de convex hull van alle objecten of de geaggregeerde geometrie van alle objecten. Hieronder staat een Oracle-voorbeeld met een geometrische aggregatiefunctie om op basis van perceelgrenzen, de gemeentegrenzen (grenzen waarvan de gemeentecode links en rechts verschillend zijn) af te leiden.

```
select sdo_aggr_union(mdsys.sdoaggrtype(shape,
0.1)) as g_shape,
r_municip, l_municip
from xfio_boundary
where r_municip <> l_municip
group by r_municip, l_municip;
```

RUIMTELIJKE INDEXERING EN CLUSTERING

De beschikbaarheid van ruimtelijke datatypen en operatoren is een deel van de DBMS-diensten die voor een GIS noodzakelijk zijn.

Voorlopig heeft alleen OpenGIS ook een implementatiespecificatie gedefinieerd

Een ander belangrijk aspect is de toegang tot de ruimtelijke data; de ruimtelijke indexering en clustering. Bekende ruimtelijke indexstructuren zijn de *quadtree* en de *r-tree*. Indexstructuren en de SQL statements om deze te definiëren zijn geen onderdeel van de SQL (OpenGIS) standaard, zoals blijkt uit de fragmenten hieronder.

```
/* Oracle */
/* eerst tabel user_sdo_geometry_metadata vullen
*/
insert into user_sdo_geom_metadata
values ('xfio_boundary', 'shape',
mdsys.sdo_dim_array(
mdsys.sdo_dim_element('X', -25000000, 325000000,
0.5),
mdsys.sdo_dim_element('Y', 275000000, 625000000,
0.5)), NULL);
/* daarna spatial index definiëren (default is
rtree) */
create index xfiox_boundary_0
on xfio_boundary (shape)
indextype is mdsys.spatial_index
parameters ('sdo_fanout=64 sdo_rtr_pctfree=10');

/* Informix */
create index xfiox_boundary_0
on xfio_boundary (shape st_geometry_ops)
using rtree (NO_SORT='FALSE', FILL_FACTOR='90',
BOUNDING_BOX_INDEX='NO');

/* Ingres */
create index xfiox_boundary_0
on xfio_boundary(shape)
with structure=rtree,
range=((-25000000, 275000000), (325000000,
625000000));
```

Naast indexeren is ruimtelijk clusteren van belang om te zorgen dat objecten die in de werkelijkheid dicht bij elkaar liggen, ook op (RAID) disk dicht bij elkaar worden opgeslagen. Omdat in veel ruimtelijke selecties meestal enige honderden tot duizenden objecten (die dan ook vaak dicht bij elkaar liggen) gevonden worden, zou dit zonder clustering nog veel tijd kunnen kosten. De ruimtelijke index weet snel de adressen van de geselecteerde objecten te vinden. Maar als voor het ophalen van de objecten zelf evenzoveel aparte disk-leesacties nodig zouden zijn, gaat dit toch traag. Het mooiste zou zijn indien de data (in de tabel zelf) volgens de ruimtelijke index geclusterd zouden kunnen worden. Geen van de geteste systemen kan dat op dit moment. Wel kunnen ze clusteren op een niet-ruimtelijk attribuut (zoals postcode of de alfanumerieke 'Spatial Location Code', SLC), waar dan wel een ruimtelijke locatie in zit 'verborgen'. Een Ingres voorbeeld:

```
modify xfio_boundary to btree on slc;
```

Ook in Oracle kan een dergelijke primaire opslagstructuur worden gedefinieerd (zodat ruimtelijke clustering wordt afgedwongen), maar helaas kan er dan geen ruimtelijke index meer gebruikt worden. De oplossing is dan om de data voor het laden ruimtelijk te clusteren (buiten de DBMS). Een alternatief bij Oracle is het werken met partities. Deze voorkomen dat de data totaal willekeurig worden verdeeld, zoals dit binnen één grote tabel zou kunnen gebeuren. Elke partitie heeft namelijk een eigen fysieke opslag. Nadeel is dat de gebruiker zelf moet bepalen wat redelijke 'deel'-tabellen zijn. Uit testen met niet geclusterde data bleek dat query's tot 30 keer langzamer waren dan dezelfde query's op geclusterde data (op een tabel met een omvang van ongeveer éénvijfde van de data die in tabel 1 zijn weergegeven).

	Laadtijd	Database omvang
Ingres 2.5	96 uur	24.296 Mb
Oracle 9.0.1 (9.2.0)	77 uur	48.151 Mb
Informix 9.21	94 uur	49.746 Mb

Tabel 1: Database laadtijden (onzuiver) en database omvang (zeven tabellen).

	Index creatie tijd	Index omvang
Ingres 2.5 (alleen bbox)	5 uur	4.589 Mb
Oracle 9.0.1 (alleen bbox)	75 uur	10.486 Mb
Informix 9.21 (ook shape)	18 uur	16.542 Mb

Tabel 2: Index creatie tijd (onzuiver) en omvang (totaal acht representatieve indexen).

BENCHMARKING

In de inleiding is reeds aangegeven, dat er geen standaard benchmark voor geo-DBMS'en bestaat (zoals deze wel voor 'administratieve' DBMS'en bestaat). Daarom is er zelf een test ontwikkeld. De test-dataset bestaat uit de geometrische data van het Kadaster (van 1 juni 2001) met ruim 120 miljoen ruimtelijke records verspreid over 7 tabellen. De testconfiguratie bestaat uit een Sun Enterprise E3500 server van het GDMC (Geo-Database Management Center) binnen de TU Delft. Deze computer heeft twee 400 MHz UltraSPARC CPUs, 2 Gb hoofdgeheugen en 600 Gb diskruimte, op verschillende RAID0- en RAID5-sets.

Hoewel het laden van de data in de verschillende geo-DBMS'en moeilijk met elkaar te vergelijken is, omdat de ruimtelijke datatypen niet helemaal gelijk zijn voor wat betreft hun nauwkeurigheid (en dus omvang van opslag) en de verschillende DBMS'en meer of minder tools bieden voor efficiënte bulkloading, geeft tabel 1 een aardig overzicht van het laden van de dataset in de drie geteste systemen Ingres 2.5, Oracle 9.2.0 (na laden in 9.0.1, maar er wordt geen verschil in laadtijd verwacht) en Informix Dynamic Server 2000 version 9.21 (met Spatial Datablade version 8.11). Wel moet worden opgemerkt dat dit onzuivere laadtijden zijn, omdat het systeem ook met andere taken bezig kon zijn. Bij Informix moet bovendien worden opgemerkt dat er ook een 'fast load' optie bestaat, die wegens tijdgebrek nog niet is getest. Alle DBMS'en hebben zeer veel parameters voor het tunen van het systeem.

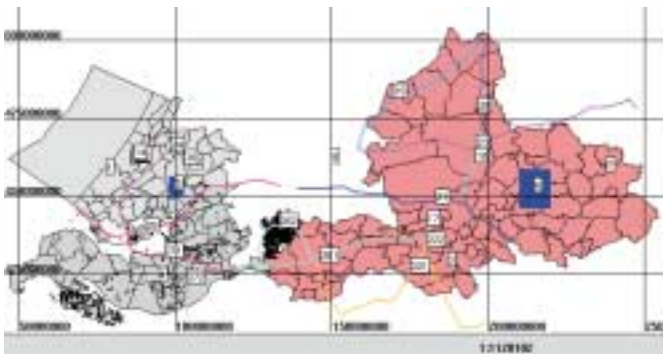
Gedurende de benchmark is ongeveer evenveel tijd besteed aan het tunen van de verschillende systemen, dit in samenwerking met de verschillende betrokken leveranciers. Naast de drie geteste DBMS'en, staan ook IBM DB2 (ook OpenGIS compliant) en enige niet commerciële DBMS'en (Postgres/PostGIS en MySQL) op de planning bij het GDMC, om getest te worden.

Na het laden van de data, zijn het ruimtelijk indexeren van de data en het analyseren van de data (voor latere query-optimalisatie) de vervolgstappen. Ook hier is het vergelijk weer lastig omdat de DBMS'en onderling nogal wat verschillen. In Tabel 2 zijn voor acht representatieve indexen, waarvan vier ruimtelijke, de creatietijd en index-omvang aangegeven. Alle ruimtelijke indexen zijn rtree's (Oracle biedt ook de opties grid en quadtree). Bij Ingres en Oracle bevat de index alleen de bounding box, bij Informix bevat de index ook (nog eens) de complete geometrie (deze optie bieden Ingres en Oracle niet). Op basis van een beperkt aantal testen, wordt geschat dat de Informix spatial indexen met alleen bbox ongeveer 20% kleiner zullen zijn. Bij Oracle moet worden opgemerkt dat de indexen weer in versie 9.0.1 zijn aangemaakt, maar dat uit een beperkt aantal testen bleek dat het aanmaken van spatial indexen flink sneller is geworden (bijna gehalveerd). Op basis van de mix spatial en gewone indexen, is een schatting van de aanmaaktijd van deze 'index' mix in Oracle 9.2.0 zo'n 43 uur.

Nu al het voorbereidende werk is uitgevoerd kan het bevragen van de data beginnen. Er zijn vele typen vragen uitgevoerd. Hieronder worden de resultaten van een (representatief) type vraag getoond. De tabel met perceelgrenzen (ongeveer 46 miljoen records) wordt bevraagd op basis van het attribuut met de geometrie (shape), waarbij het een SQL 'select count *' query betreft. Er wordt met een aantal verschillende geometrie-typen gezocht: rechthoeken (bbox, dit komt in de praktijk het meest vaak voor, want een beeldscherm is ook een rechthoek), willekeurige polygonen (overlap zoeken met specifiek zoekgebied) en een aantal lange polylines (representatief voor selectie van objecten door opgegeven tracé), zie afbeelding 2. Voorbeeld van een selectie op basis van overlap met een gegeven zoek polygon:

```
select count(*) from xfio_boundary
where ST_Intersects (shape, ST_PolyFromText(
'polygon ((78550000 457900000, 78550000
458025000,\
78650000 458025000, 78650000 457900000, 78550000
457900000))',
28992));
```

Tabel 3 toont de resultaten van deze geselecteerde query's. Deze metingen zijn wel zuiver (het systeem was niet belast met andere



Afbeelding 2: Overzicht van de query-objecten (vooral in Zuid-Holland en Gelderland).

Query	#objecten in bbox	# gevonden objecten	Ingres 2.5	Oracle 9.2.0	Informix 9.21
1.kleine rechthoek	546	415	1	1	0
4.midden rechthoek	964	804	0	1	0
6.grote rechthoek	175144	175098	35	23	29
10.polygon met 8 punten	2264	1704	1	1	1
14.polygon met >50 punten	1215	550	1	1	1
16.polygon met 2 gaten	31247	22402	6	12	6
20.lange verticale polyline	137389	1357	17	6	2
24.lange diagonale polyline	2500267	2561	473	43	5
26.lange multi-polyline	1981833	4514	3424	63	6

Tabel 3: Query-resultaten in tijden (zuivere metingen afgerond naar hele seconden).

taken) en bovendien zijn de query's herhaald uitgevoerd om de reproduceerbaarheid en de betrouwbaarheid van de querytijden te controleren. Opgemerkt moet worden dat de indexering en clustering goed werkt (behalve dat Ingres niet efficiënt omgaat met 'lange polyline' query's). De gevonden aantallen records zijn per vraag voor de verschillende geo-DBMS'en gelijk.

TOPOLOGISCHE STRUCTUREN

Een topologische structuur ('complex feature' in OpenGIS-termen) beschrijft de samenhang tussen de verschillende (deel)objecten en kan hiermee ook redundantie voorkomen; bijvoorbeeld bij een perceel wordt niet de Polygon-geometrie opgeslagen, maar verwijzingen naar de grenzen. Het OpenGIS Consortium noemt verschillende keren het begrip topologie in haar abstracte specificaties. Wat echter nog steeds ontbreekt zijn implementatiespecificaties voor een DBMS-omgeving. Hetzelfde geldt voor de overeenkomstige standaarden van de ISO TC 211. Deze beschrijven de *topologische primitieven* (zonder een poging dit te vertalen): node, edge, face en solid. In de standaard worden de 3D-geometrie en topologische primitieven beide aangeduid met term *solid*. Voorgesteld wordt om de term *volume* voor de 3D-topologische primitieve te gebruiken.

Afbeelding 3: Kadastrale landmeters verzamelen attributen bij de topologische gestructureerde grens, nadat aanwijs van betrokken partijen achter de rug is (foto Kadaster).

In geval van topologie-structuurmanagement kan het DBMS de correctheid en de consistentie controleren en bewaken tijdens het bijhouden (face gesloten?, geen kruisende edges?). Tevens kunnen dan ook complexe operaties binnen het DBMS worden uitgevoerd. Ondanks het feit dat topologische structuren goed bekend zijn binnen de GIS-wereld en de verwijzingen prima in een DBMS opgeslagen kunnen worden, is het nog steeds een onbeantwoorde vraag hoe de topologische modellen efficiënt in een relationele DBMS zijn te implementeren. Een belangrijk aspect hierbij is het vertalen van de topologische primitieven in hun geometrische tegenhangers. Het is dan mogelijk om een DBMS-view op een topologische primitieve te definiëren, die het dan doet voorkomen als een geometrische primitieve. Zo kan het beste van twee werelden verkregen worden: aan de ene kant topologische structuren (zonder redundantie) en aan de andere kant het gemak van geometrische primitieven tijdens het bevragen, analyseren en presenteren van de geo-informatie. Een ander voordeel van een topologische structuur bestaat eruit dat deze het meest natuurlijke datamodel is voor sommige applicaties; bijvoorbeeld bij het kadastraal landmeten worden edges (grenzen) met hun attributen gemodelleerd (zie afbeelding 3).

Bij de TU Delft is er de afgelopen twee jaar onderzoek gedaan naar het managen van topologische structuren in een Oracle Spatial DBMS. Sinds kort is er ook een eerste commercieel product op de markt verschenen dat topologie-structuurmanagement in Oracle 9i ondersteunt, LaserScan Radius Topology. Gedurende het onderzoek is er regelmatig contact geweest met LaserScan en zijn de ervaringen uitgewisseld. Het LaserScan product wordt op dit moment door de TU Delft getest. Het is bovendien de verwachting dat Oracle zelf vanaf versie 10 ook topologie-management zal gaan bieden. De TU Delft zal in het voorjaar van 2003 de beta-versie van Oracle 10i gaan testen.

3D RUIMTELIJKE DATATYPEN

Tot nu toe is de functionaliteit beperkt gebleven tot twee dimensies (2D). Echter steeds vaker is er behoefte aan drie dimensionale (3D) data. De 0D (punt), 1D (lijn) en 2D (vlak) geometrische primitieven, zoals beschikbaar in Informix en Oracle, kunnen in een 3D ruimte worden gedefinieerd, elk punt kan naast een x- en y-waarde ook een z-waarde hebben. Echter, alle functies werken in 2D (projecteren op het vlak z=0). Bovendien is er geen 3D geometrische primitieve (volume) beschikbaar. De OpenGIS implementatiespecificatie voor 'simple features' is beperkt tot 2D, hoewel de abstracte standaarden OpenGIS en ISO wel 3D functionaliteit omvatten.

Bij de TU Delft, afdeling Geodesie is onderzoek gedaan naar een implementatiespecificatie (en getest binnen Oracle Spatial) voor een uitbreiding van het geo-DBMS met een 3D geometrische primitieve. De conclusie is dat het Polyhedron-type de meest passende en geschikte uitbreiding is. Een Polyhedron wordt gedefinieerd door een verzameling platte vlakken, die een volume omsluiten; zie afbeelding 4. Deze vlakken beschrijven exact één buitenrand en mogelijk één of meer binnenranden (volume met holtes er binnen in). Wel is de validatie van een dergelijk type niet

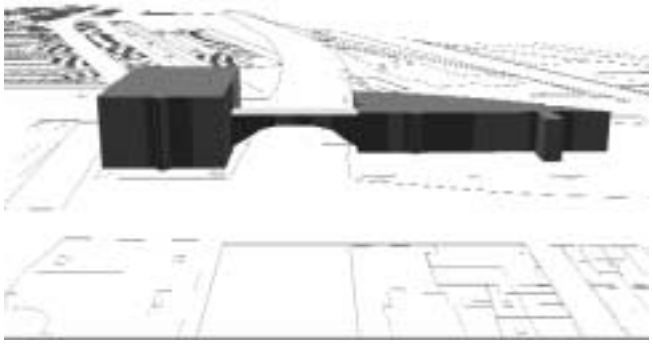
De conclusie is dat het Polyhedron-type de meest geschikte uitbreiding is

eenvoudig: zijn alle vakken plat, sluiten ze goed aan, doorkruisen de vlakken elkaar niet, zijn de buitenrand en de mogelijke binnenranden gesloten, zijn er geen loshangende vlakken?

Is een 3D object gevalideerd dan kan deze worden opgeslagen en geïndexeerd, bijvoorbeeld met een 3D r-tree. Er is bovendien een hele verzameling 3D operatoren denkbaar voor de 0D tot en met 3D geometrische primitieven. In het kader van een afstudeerproject bij de TU Delft, Geodesie is er een implementatie gemaakt door Călin Arens.

CONCLUSIE

In dit artikel is een overzicht gegeven van de op dit moment gestandaardiseerde geometrische functionaliteit en de beschikbaarheid hiervan in een aantal commerciële databases. Ook is er een benchmark uitgevoerd, waarbij de verschillende databases met behulp van een flinke test-dataset met elkaar zijn vergeleken. Geconcludeerd kan worden dat de huidige generatie geo-DBMS'en al behoorlijk volwassen is. Naast de op dit moment beschikbare functionaliteit is er ook aandacht besteed aan een tweetal zaken die op dit moment nog niet aanwezig zijn: topologie-structuurmanagement en 3D geometrische functionaliteit. Met deze functionaliteit erbij vormen de geo-DBMS'en voor nog meer GIS-toepassingen de geschikte opslag- (en voor een flink deel analyse-) omgeving. Er zijn echter altijd meer onderwerpen



AFBEELDING 4: 3D RUIMTELIJK OBJECT (GEBOUW MET BOOG) GEDEFINIEERD DOOR PLATTE VLAKKEN.

te bedenken die (de komende tijd) de aandacht zullen vragen:

1. Uitbreiding van de geometrische datatypen en structuren met bijvoorbeeld geo-rasters, efficiënte puntenwolken, driehoekennetwerken (TIN's);
2. Ruimtelijke-temporele data: integrale ondersteuning van zowel de ruimtelijke als temporele kenmerken van objecten (in hun topologische samenhang);
3. Visualisatie en fysieke objecteigenschappen; met name voor 3D modellen zijn zaken als textuur en reflectie-eigenschappen van de objecten van belang;
4. Gedistribueerde geo-DBMS'en: protocollen voor heterogene omgevingen met verschillende typen geo-DBMS'en die elk een deel van de data managen (de OpenGIS Web Feature Server and Web Coverage Server specificaties);
5. Ondersteuning van XML (GML) als uitwisselingsformaat, maar ook als datatype in het DBMS.

Naast de hier genoemde commerciële DBMS-producten, zijn er ook een enkele niet-commerciële DBMS'en die ruimtelijke gegevens ondersteunen. Postgres is hier een goed voorbeeld van, omdat het al sinds de eerste versie (eind jaren 80) ruimtelijke datatypen en indexering ondersteunt. In het begin in de eigen taal Postquel (en dit alles ruim voor het OpenGIS-tijdperk), later ook in SQL en volgens de OpenGIS-specificatie (PostGIS). Een andere veel gebruikte niet commerciële DBMS is MySQL. Deze biedt tot nu toe geen ondersteuning voor ruimtelijke data, maar hier lijkt verandering in te komen, want in de aankondiging van de nieuwste versie (4.1), wordt ruimtelijke functionaliteit expliciet genoemd.

Meer informatie is te vinden op de website van de sectie GIS-technologie TU Delft (http://www.geo.tudelft.nl/gist) en dan bij publicaties zoeken naar (Spatial) DBMS, 3D, testing, topology. De rapporten en artikelen daar bevatten vele literatuurverwijzingen.

Voor de medewerking aan dit artikel dankt de auteur het Kadaster voor het beschikbaar stellen van de data, verder de DBMS-leveranciers en tenslotte de onderzoekers bij de TU Delft; Theo Tijssen, Wilko Quak, Jantien Stoter, Sisi Zlatanova. ●

Peter van Oosterom (oosterom@geo.tudelft.nl) is verbonden aan de Afdeling Geodesie van de TU Delft.