erschienen in: Datenbank-Spektrum, Heft 22, dpunkt-Verlag, August 2007, 37-43

Thomas Brinkhoff

Open-Source-Geodatenbanksysteme

Der Einsatz von (objekt-) relationalen Geodatenbanksystemen hat in den letzten Jahren stetig zugenommen. Dies hat eine Reihe von Gründen; insbesondere die wachsende Integration von Geoinformationsanwendungen in die traditionelle IT und die rasante Zunahme von raumbezogenen Web-Applikationen sind hier zu nennen. Pionier im Bereich Geodatenbanksysteme war Informix mit seinen räumlichen Datablade-Erweiterungen. Das Geodatenbanksystem mit dem derzeit wohl größten Funktionsumfang stellt Oracle Spatial dar. Daneben haben sich in letzter Zeit aber auch zwei Open-Source-Datenbanksysteme in diesem Segment etabliert: PostgreSQL/PostGIS und MySQL. Beide Systeme unterscheiden sich in Funktionalität – und damit in ihrer Zielrichtung – deutlich voneinander. Dieser Beitrag soll diese Unterschiede herausarbeiten und einen Vergleich mit den relevanten Datenbankstandards vornehmen. Außerdem werden exemplarische Anwendungsszenarien für die beiden Open-Source-Systeme vorgestellt.

1 Einleitung

Traditionell war die Verarbeitung von Geodaten Domäne von *Geoinformationssystemen* (*GIS*). Diese waren neben der Eingabe, Analyse und Präsentation auch für die Verwaltung, Speicherung und Abfrage von raumbezogenen Daten verantwortlich. Die Entwicklung von Geoinformationssystemen war lange Zeit nicht durch die Verwendung bereits vorhandener Komponenten und Systeme geprägt, sondern erforderte Eigenentwicklungen seitens der GIS-Hersteller [Tomlinson 1984]. So wurden bis Ende der 90er-Jahre vielfach Geodaten ausschließlich in Dateien oder GIS-spezifischen Datenhaltungskomponenten gespeichert, da Standard-Datenbanksysteme keine adäquaten Modellierungs- und Anfragemöglichkeiten boten.

Durch das Vordringen von Geodaten in immer mehr Anwendungs- und Geschäftsfelder verlieren aber Geoinformationssysteme ihre zentrale Rolle, die sie zum Beispiel in Katasterverwaltungen oder bei Energieversorgungsunternehmen innegehabt hatten. Ein GIS muss sich daher nahtlos in die IT-Infrastruktur einer Organisation einbetten. Hierfür steht das seit Mitte der 90er-Jahre populäre Schlagwort *offenes GIS* (engl. *Open GIS*), das das Ziel verfolgt, über entsprechende Standards die mangelnde Interoperabilität zu beseitigen und Geodaten Standardanwendungen zugänglich zu machen. Ergebnis dieser Entwicklung sind offene *Geoinformationsinfrastrukturen (GDI)* auf Basis von standardisierten *Geodatendiensten* (engl. *Geospatial Web Services*). In diesem Rahmen wird die Geodatenhaltung weitgehend (objektrelationalen) *Geodatenbanksystemen* übertragen [Brinkhoff 2005] [Rigaux et. al 2002].

Ein weiterer wichtiger Aspekt, der die Verwendung von Geodatenbanksystemen prägt, ist der Einsatz für *raumbezogene Web-Applikationen*. Neben Digital-Earth-Anwendungen wie Google Earth und Google Maps sind hier das Tourismus-Marketing, Routenplaner, Informationsportale und auch die öffentliche Verwaltung als wichtige Triebfedern zu nennen. In letzter Zeit nimmt insbesondere die Bedeutung von *kartenbezogenen Mashups* – also der Kombination von mehreren Datenquellen zu einer neuen Webanwendung („into an integrated experience“ [Wikipedia 2006]) – zu [Brinkhoff 2007a].

Drei objektrelationale Geodatenbanksysteme befinden sich derzeit auf dem Markt, die nicht zur Kategorie der Open-Source-Systeme gehören [Lemmen 2007] [Kirchner & Rösler 2007]: IBM Informix Spatial DataBlade, IBM DB2 Spatial Extender und Oracle Spatial. Wesentliche Merkmale dieser Systeme sind, dass deren Geometriedatenmodell die Forderungen des *Simple Feature Models* abdeckt, dass räumliche Basisanfragen formuliert werden können und dass eine räumliche Indexstruktur zur Verfügung steht.

Daneben stellen zwei Open-Source-Datenbanksysteme Geodatenbankfunktionalität zur Verfügung: *PostgreSQL* und *MySQL*. Bei PostgreSQL sind zwei Varianten zu unterscheiden: Neben eingebauten geometrischen Datentypen gibt es die objektrelationale Erweiterung *PostGIS*, die seit der Version 1.0 konform zum Simple Feature Model ist. Die Funktionalität von PostgreSQL/PostGIS und MySQL soll in diesem Beitrag betrachtet und (sowohl miteinander als auch mit anderen Systemen und den wichtigsten Standards für Geodatenbanksysteme) verglichen werden.

Nachfolgend werden zunächst die beiden Geodatenbankstandards ISO 19125 »Simple Feature Access« und ISO/IEC 13249-3 »SQL/MM Spatial« kurz vorgestellt. Danach folgt eine Betrachtung des Umfangs und der Funktionalität der von PostgreSQL/PostGIS und MySQL bereitgestellten Geometriemodelle. Abschnitt 5 befasst sich mit der räumlichen Anfragebearbeitung. Der sechste Abschnitt stellt exemplarisch Anwendungsklassen vor, in denen PostgreSQL/PostGIS bzw. MySQL mit ihren jeweiligen Möglichkeiten geeignet zum Einsatz kommen. Die Betrachtungen in diesem Beitrag beziehen sich – wenn nicht anders erwähnt – auf PostgreSQL in der Version 8.2 [PostgreSQL 2007], PostGIS 1.2 [Refractions 2007] und MySQL 5.1 [MySQL 2007].

2 Geometrieklassenmodelle

Einen wichtigen Aspekt offener Geoinformationslösungen stellt die Standardisierung des verwendeten Geometrieklassenmodells dar. Hierzu haben das *Open Geospatial Consortium (OGC)* und die *International Organization for Standardization (ISO)* eine Reihe von Spezifikationen erarbeitet. Für Geodatenbanksysteme sind – wie bereits erwähnt – zwei Spezifikationen von Bedeutung: ISO 19125:2004 »Simple Feature Access« und ISO/IEC 13249-3:2006 »SQL/MM Spatial«.

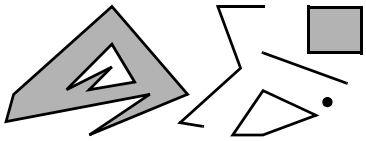

Abb. 1: Beispiele für Simple Features

2.1 ISO 19125 Simple Feature Access

Das *Simple Feature Model*, das ursprünglich vom OGC erarbeitet wurde, steht seit 2004 als ISO-Norm 19125:2004 *»Simple Feature Access«* zur Verfügung. *Simple Features* sind Geometrien im zweidimensionalen Raum, deren Stützpunkte geradlinig miteinander verbunden sind (vgl. Abbildung 1). Während der erste Teil der Norm das Klassenmodell mit geometrischen Datentypen und zugehörigen Operationen beschreibt, behandelt der zweite Teil (»SQL option«) dessen Umsetzung in ein Datenbankschema.

Das Geometrieklassenmodell ist in Abbildung 2 dargestellt. Die Oberklasse »Geometry« bündelt die Attribute und Methoden, die allen Simple Features gemein sind. So kann jeder Geometrie ein *räumliches Bezugssystem* (»SpatialReferenceSystem«) zugeordnet sein. Von der Klasse »Geometry« leiten sich vier Geometrieformen ab:

- »Point« beschreibt einen *Punkt*.
- »Curve« ist die abstrakte Oberklasse für *Linien*. *Streckenzüge* werden durch die Unterklasse »LineString« erzeugt. Es gibt zwei Spezialisierungen von Streckenzügen: Die Klasse »Line« entspricht einer *Strecke* und »LinearRing« repräsentiert einen *Ring*, also einen geschlossenen Streckenzug.
- »Surface« ist die abstrakte Oberklasse für *Flächen*. Konkrete Flächenobjekte werden durch die Klasse »Polygon« erzeugt, die *einfache Polygone* repräsentiert, die ggf. Löcher als Aussparungen besitzen können.
- »GeometryCollection« beschreibt eine Sammlung von Geometrien, die aus beliebig vielen Geometrien besteht. Für den Fall, dass alle Elemente der gleichen Klasse angehören, gibt es eine Reihe von spezialisierten Geometriekollektionen, wobei nur die Unterklassen »MultiPoint«, »MultiLineString« und »MultiPolygon« instanziierbar sind.

Die Methoden, die das Simple Feature Model unterstützt, sind in Tabelle 1 übersichtsmäßig dargestellt.

2.2 SQL/MM Spatial

Die ISO/IEC-Norm 13249-3:2006 *»SQL/MM Spatial«* liegt nun mit der Edition von 2006 in der dritten Fassung vor. SQL/MM Spatial kann man als eine Art Erweiterung des Simple Feature Models auffassen. Die wesentliche Ergänzungen im Geometrieklassenmodell sind (vgl. Abbildung 3):

- Die Klasse »ST_Curve« weist zwei zusätzliche Unterklassen auf: »ST_CircularString« und »ST_CompoundCurve« repräsentieren Linienzüge, die *Kreisbögen* enthalten, bzw. *zusammengesetzte Linienzüge*, die aus geradlinigen und kreisbogenförmigen Teillinien bestehen.
- Die Klasse »ST_Surface« weist eine zusätzliche Unterklasse auf: »ST_CurvePolygon«, deren innere und äußere Ringe durch Instanzen der Klasse »ST_Curve« repräsentiert werden und damit auch Kreisbögen enthalten können.

Daneben spezifiziert SQL/MM Spatial auch zusätzliche Funktionen, so u.a. Methoden zum Validieren von Geometrien, Koordinatentransformationen und Methoden zur Bereitstellung von GML (Geography Markup Language). Tabelle 1 zeigt eine Übersicht.

SQL/MM Spatial enthält seit der dritten Edition außerdem ein *Netzwerk- und Topologiemodell*, das in [Brinkhoff 2007b] näher erörtert wird. Da Topologiedaten bislang von Open-Source-Geodatenbanksystemen nicht explizit unterstützt werden, wird darauf im Folgenden nicht weiter eingegangen.

3 Geometrieklassenmodelle in PostgreSQL, PostGIS und MySQL

3.1 Geometrieklassen

Bei PostgreSQL muss man zwei unterschiedliche Geometriemodelle unterscheiden. PostgreSQL besitzt zum einen die fest eingebauten Geometrietypen »Point« (Punkt), »Line« (Gerade), »LSeg« (Strecke), »Box« (achsenparalleles Rechteck), »Path« (offener oder geschlossener Streckenzug), »Polygon« (Polygon ohne Löcher) und »Circle« (Kreis) [PostgreSQL 2007]. Diese Geometrietypen sind *nicht* konform zu den dargestellten Standards; auch ist die

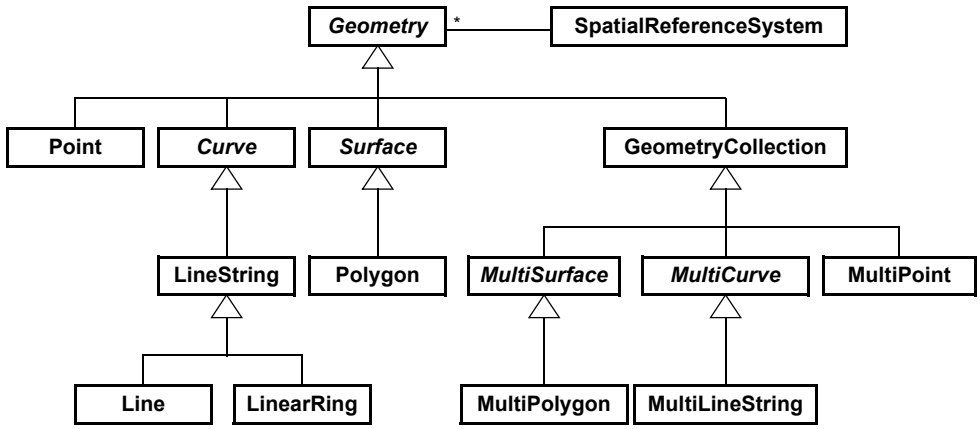

Abb. 2: Die wesentlichen Klassen und Beziehungen des Simple Feature Models (ISO-Norm 19125:2004 »Simple Feature Access«)

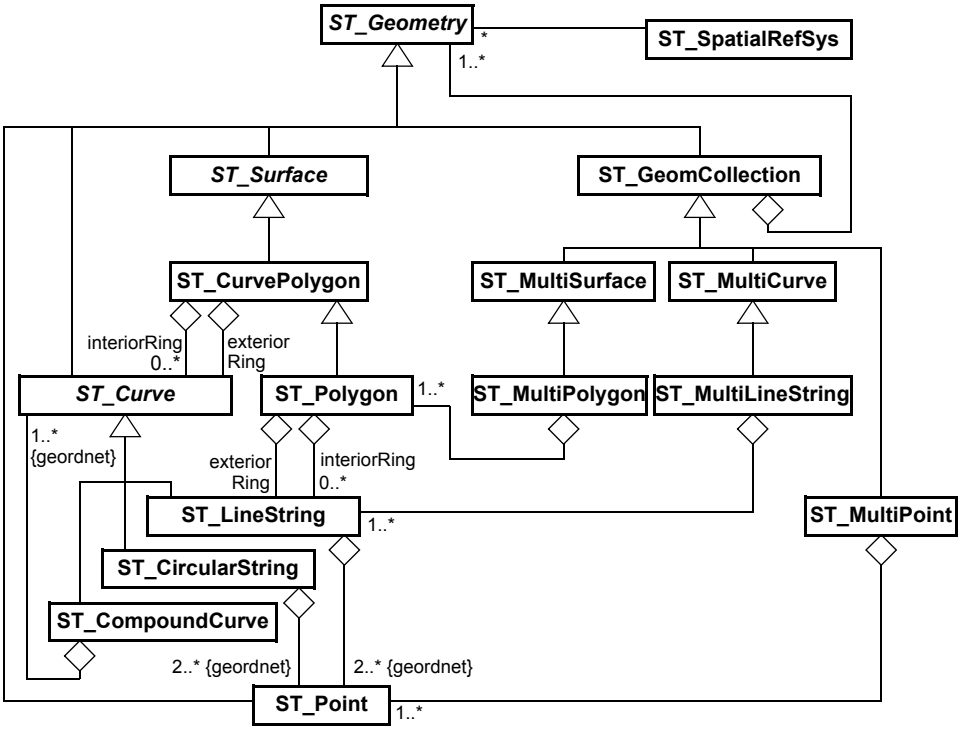

Abb. 3: Die wesentlichen Geometrieklassen und Beziehungen in SQL/MM Spatial

Funktionen	ISO 19125	SQL MM/ Spatial	PostgreSQL /PostGIS	MySQL
Zugriff auf *Basiseigenschaften* (»Dimension«, »GeometryType«, Zugriff auf die Geometriebestandteile)	X	X	X	X
Test von *Geometrieeigenschaften* (»IsSimple«, »IsClosed«, »IsRing«)	X	X	X	(X)
Messfunktionen (»Length«, »Area«, »Distance«)	X	X	X	X
Geometrische Funktionen auf einer Geometrie (»Distance«, »Envelope«, »Centroid«, »Buffer«, »ConvexHull«)	X	X	X	(X)
Verschneidungsoperationen	X	X	X	
Topologische Prädikate gemäß *Dimensionally-Extended 9-Intersection Matrix* [Clementini & Di Felice 1996]	X	X	X	(X)
Konvertierung in *Well-Known-Text* (*WKT*) und *Well-Known Binary* (*WKB*)	X	X	X	X
Konvertierung in *Geography Markup Language* (*GML*)		X	X	
Konvertierung in *Scalable Vector Graphics* (*SVG*) und *Keyhole Markup Language (KML)*			X	
Validierung von Geometrien		X	X	
Koordinatentransformationen		X	X	
Approximation von Kurvensegmenten durch Streckenzüge		X		
Gesonderte geometrische Funktionen und Verschneidungen auf *achsenparallelen Rechtecken*			X	
Generalisierung und *Polygonisierung* von Linien			X	
Geometrische Aggregatfunktionen			X	

Tabelle 1: Vergleich der Funktionen im Simple Feature Model (ISO 19125), SQL MM/Spatial, PostgreSQL/ PostGIS und MySQL

Funktionalität eingeschränkt. Daher werden Geometrien in PostgreSQL in der Regel durch die Erweiterung PostGIS repräsentiert und verarbeitet. Deren Klassenmodell ist konform zu dem Simple Feature Model. Das Hinzufügen eines geometrischen Attributs erfolgt über die PostGIS-Funktion »AddGeometryColumn ([<Schemaname>,] <Tabellenname>, <Spaltenname>, <ID des räumlichen Bezugssystems>, <Geometrietyp>, <Dimension>)«, wobei der Geometrietyp als Zeichenkette übergeben wird. Die Funktion speichert zudem alle notwendigen Metadaten der geometrischen Attribute in der systemseitigen Tabelle »GEOMETRY_COLUMNS«.

Mit der Version 1.2 von PostGIS stehen nun – mit gewissen Einschränkungen – auch (gemäß SQL/MM Spatial) Konstruktoren für Geometrien mit Kreisbogensegmenten zur Verfügung.

Die Geometrieklassen von MySQL entsprechen den Klassen des Simple Feature Models. Sie können wie herkömmliche Datentypen bei der Vereinbarung eines Spaltentyps verwendet werden.

3.2 Funktionen

Die Funktionen, die von PostgreSQL/PostGIS und MySQL angeboten werden, sind in Tabelle 1 übersichtsmäßig dargestellt. Zusammenfassend lässt sich für PostgreSQL/PostGIS sagen, dass eine volle Unterstützung der ISO 19125-Funktionalität gegeben ist. Außerdem wird eine Auswahl von SQL/MM Spatial-Funktionen angeboten (so die Validierung von Geometrien, die Bereitstellung von GML und Koordinatentransformationen). Darüber hinaus stehen auch weitergehende Konvertierungsfunktionen nach SVG und seit PostGIS 1.2 auch nach KML zur Verfügung. Zusätzlich sind zum Beispiel eine Generalisierung gemäß Douglas-Peuker und die Bildung von Polygonen aus Streckenzügen möglich.

Auch ist zu erwähnen, dass es gesonderte geometrischen Operationen gibt, die auf den *minimal umgebenden Rechtecken* der Geometrien arbeiten – dies ist effizienter als die vergleichbaren Operationen auf den exakten Geometrien und wird im Rahmen der mehrstufigen Anfragebearbeitung benötigt (vgl. Abschnitt 4.2).

Für die Funktionen zur Unterstützung von *linearen Bezugssystemen* in PostgreSQL/PostGIS sei auf die Erörterung in [Brinkhoff 2007b] verwiesen.

Die Funktionalität von MySQL ist deutlich eingeschränkter. MySQL implementiert nur eine Untermenge der Funktionen des Simple Features Models. So sind die Testfunktionen »IsSimple« und »IsRing« nicht funktionsfähig. Bei den geometrischen Funktionen auf einer Geometrie stehen »Buffer« und »ConvexHull« ebenso nicht zur Verfügung wie alle Verschneidungsfunktionen. Topologische Prädikate können nur auf Basis der minimal umgebenden Rechtecke geprüft werden.

3.3 Räumliche Bezugsysteme

Geometrien in den vorgestellten Klassenmodellen können ein *räumliches Bezugssystem* aufweisen. Diese erlauben die Interpretation der gespeicherten Koordinaten als Beschreibung von Lage- und Ausdehnungsinformationen in einem (realen) Datenraum. Ein räumliches Bezugssystem besteht aus einem *Koordinatensystem*, einem Geltungsbereich und Angaben, die es erlauben, Daten aus unterschiedlichen Koordinatensystemen auf ein globales System abzubilden.

Am stärksten verbreitet sind *EPSG-Bezugssysteme*. Diese werden von dem »Surveying & Positioning Committee« der »International Association of Oil & Gas Producers« (OGP) gepflegt, in das im Jahr 2005 die »European Petroleum Survey Group« (EPSG) aufgegangen ist. Die EPSG-Datenbank enthält mehrere Tausend georeferenzierende räumliche Bezugssysteme, die durch einen eindeutigen *EPSG-Schlüssel* identifiziert werden [S&P 2007].

PostgreSQL/PostGIS hält die EPSG-Bezugssysteme in der Systemtabelle »SPATIAL_REF_SYS« vor. Zu einer angemessenen Unterstützung von räumlichen Bezugssystemen gehört insbesondere auch deren Auswertung für Funktionen, die Längen- und Flächenangaben liefern. So ist man bei Verwendung von geografischen WGS-’84-Koordinaten in der Regel bei Entfernungen und Flächen

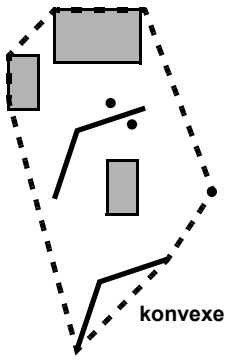

Abb. 4: Berechung der konvexen Hülle (gestrichelt) über die Geometrien eines geometrischen Attributs (= dargestellte Geometrien) mit Hilfe einer Aggregatfunktion

nicht an einer Gradangabe, sondern an einem Resultat in einer *metrischen Einheit* interessiert. Eine solche Umrechnung unterstützt PostGIS im Rahmen der Funktionen »Length«, »Area« und »Distance« nicht. Stattdessen stehen die beiden folgenden Funktionen zur Verfügung:

- »Distance_Spheroid (<Punkt>, <Punkt>, <Spheroid>)« für die Berechnung des Abstands zwischen zwei Punkten und
- »Length_Spheroid (<Geometrie>, <Spheroid>)« für die Berechnung der Länge einer Liniengeometrie.

Der Parameter »Spheroid« beschreibt dabei den Ellipsoiden, auf dem die Abstands- bzw. Längenberechnung erfolgen soll, in einer OGC-Notation. Für Flächen gibt es keine entsprechende Funktion. Hier kann/muss man sich behelfen, indem man eine Geometrie in ein projiziertes Koordinatensystem auf metrischer Basis umrechnet und dann die Abstands- bzw. Längenberechnung ausführt. Damit diese Berechnung möglichst genau ist, muss der Benutzer ein projiziertes Koordinatensystem wählen, das für die vorliegenden Daten einen möglichst geringe Fehler aufweist.

MySQL bietet keine Unterstützung zur Berechnung metrischer Längen und Flächen für geografische Koordinaten.

3.4 Geometrische Aggregatfunktionen

SQL stellt bekanntermaßen *Aggregat-*

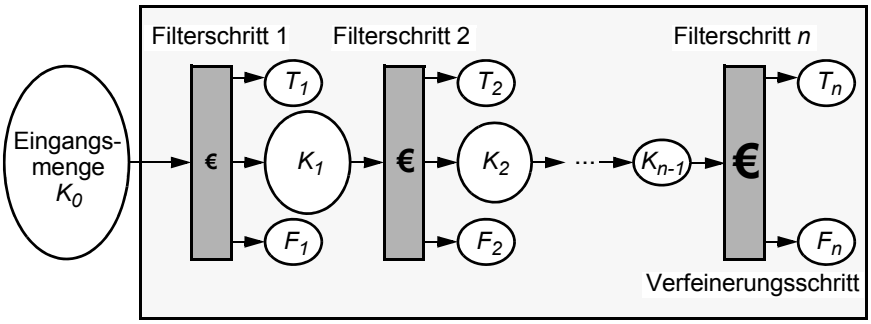

Abb. 5: Das Prinzip der mehrstufigen Anfragebearbeitung (die Größe des Euro-Symbols gibt die Höhe der Kosten für die Verarbeitung einer Geometrie an)

funktionen für einfache Datentypen zur Verfügung. Ähnliche Wichtigkeit haben auch *geometrische Aggregatfunktionen*, die die Berechnung auf geometrischen Datentypen durchführen. Da – seltsamerweise – beide Geodatenbankstandards keine geometrischen Aggregate definieren, soll hier Oracle Spatial [Oracle 2005] als Maßstab genommen werden. Dieses System stellt Aggregatfunktionen zur Berechnung des minimal umgebenden Rechtecks, der konvexen Hülle (siehe Abbildung 4), des Schwerpunkts und der geometrischen Vereinigung bereit. Diese Funktionen können an allen Stellen verwendet werden, an denen herkömmliche Aggregatfunktionen erlaubt sind.

Der Ansatz von PostgreSQL/PostGIS unterscheidet sich davon: Die »Collect«-Funktion konstruiert aus einer Relation von Geometrien eine Geometriekollektion, die die Geometrien der Relation enthält. Somit ist man in der Lage, auf der Geometriekollektion alle relevanten geometrischen Funktionen anzuwenden. In Ergänzung steht die Aggregatfunktion »GeomUnion« zur Verfügung, die die geometrische Vereinigung berechnet.

4 Räumliche Anfragebearbeitung

Neben der Datenrepräsentation stellt die (effiziente) Bearbeitung von *räumlichen Anfragen* eine essenzielle Anforderung an Geodatenbanksysteme dar. Solche Anfragen umfassen *räumliche Selektionen*:

- *Punktanfragen* bestimmen alle Datensätze, deren Geometrie einen Anfragepunkt enthält.
- *Rechteckanfragen* bestimmen alle Datensätze, deren Geometrie ein achsenparalleles Anfragerechteck schneidet.
- Analog sind *Regionsanfragen* für Anfragepolygone definiert.
- *Abstandsanfragen* bestimmen alle Datensätze, deren Geometrie sich innerhalb einer vorgegebenen Entfernung zu einer Anfragegeometrie befindet.

Als komplexere Anfragen sind zu erwähnen:

- Der *räumliche Verbund* (engl. Spatial Join) ist eine Verbundoperation zwischen zwei (oder mehr) Relationen, die mindestens eine *räumliche Verbundbedingung* beinhaltet. Solche Verbundbedingungen können topologischen Prädikaten oder Abstandsbedingungen entsprechen.
- Die *Nächste-Nachbarn-Anfrage* (engl. Nearest Neighbor Query) bestimmt aus einer Relation die nächstgelegenen Datensätze in Hinblick auf eine Anfragegeometrie.

Da die Verarbeitung von Geometrien erheblich aufwändiger als die von einfachen Datentypen ist, wenden Geodatenbanksysteme das *Prinzip der mehrstufigen Anfragebearbeitung* an, das in Abbildung 5 skizziert dargestellt ist. Dabei wird durch mehrere *Filterschritte* versucht, die Menge der Datensätze, deren Geometrien die Anfragebedingung potenziell erfüllen können (die *Kandidaten*), in frühen Filterschritten soweit wie möglich einzuschränken. Zusätzliches Ziel ist es, möglichst viele Datensätze, die die Anfrage sicher erfüllen (*Treffer*), durch die Filterschritte zu identifizieren. Im letzten Schritt, dem *Verfeinerungsschritt*, wird schließlich der aufwändige exakte Test der Anfragebedingung auf einer reduzierten Kandidatenmenge ausgeführt [Orenstein 1989].

In der Praxis von Geodatenbanksystemen hat sich als Filterstrategie der Einsatz eines *räumlichen Indexes* in Kombination der *Approximation* der Geometrien durch achsenparallele *minimal umgebende Rechtecke* (*MURs*) durchgesetzt.

4.1 Räumliche Indexe in PostgreSQL, PostGIS und MySQL

Für die in Abschnitt 3.1 genannten Geometrietypen von PostgreSQL kann ein *R-Baum* [Guttman 1984] als räumlicher Index eingesetzt werden. Die Erweiterung PostGIS verwendet hingegen nicht diese R-Baum-Implementierung, sondern realisiert einen R-Baum auf Basis des *GiST* (*Generalized Search Tree* [Hellerstein et al. 1999]). Hierfür werden zwei Gründe angegeben: der GiST in PostgreSQL ist robust gegenüber Nullwerten und erlaubt, dass nur Objektteile (sprich MURs) im Index gespeichert werden können.

Auch MySQL unterstützt räumliche Indexe. Bei Verwendung der Default-Speicher-Engine MyISAM werden R-Bäume angelegt. Bei anderen Speicher-Engines ist die Unterstützung durch räumliche Indexe eingeschränkt.

4.2 Räumliche Anfragen in PostGIS und MySQL

PostGIS nutzt räumliche Indexe nur für die Operationen, die für minimal umgebende Rechtecke bereitgestellt werden. Daher muss dies bei der Formulierung von räumlichen Anfragen berücksichtigt werden. So stellt die Anfragebedingung »WHERE geom && GeomFromText('POLYGON((5 5, 5 15, 15 15, 15 5, 5 5))')« einen Filterschritt für eine Rechteckanfrage dar (»&&« ist der Operator, der die MURs der Geometrien auf Schnitt testet). »WHERE Intersects(geom, GeomFromText('POLYGON((5 5, 5 15, 15 15, 15 5, 5 5))'))« führt hingegen einen Verfeinerungsschritt für die Rechteckanfrage ohne Nutzung des räumlichen Indexes aus. Nur die Disjunktion der beiden

Anfragebedingungen führt zu einer mehrstufigen Anfragebearbeitung mit exaktem Ergebnis. Auf diese Weise werden alle räumlichen Selektionen (außer der Abstandsanfrage) und der räumliche Verbund (bei Nutzung von topologischen Prädikaten in der Verbundbedingung) von PostGIS unterstützt.

Abstandsanfragen und -bedingungen müssen für eine effiziente Ausführung im Filterschritt auf Rechteckanfragen zurückgeführt werden. Dazu stellt PostGIS die Funktion »Expand (<Geometrie>, <Abstand>)« zur Verfügung, die das MUR einer Geometrie bestimmt und es in allen vier Richtungen der Achsen des Koordinatensystems gemäß der Abstandsangabe vergrößert.

Da in MySQL der Test auf topologische Prädikate nur auf minimal umgebenden Rechtecken implementiert ist, können nur die entsprechenden Filterschritte bei räumlichen Selektions- und Verbundanfragen ausgeführt werden; eine Unterscheidung zwischen Rechteck- und Regionsanfrage ist damit nicht möglich.

Nächste-Nachbar-Anfragen werden weder von PostGIS noch von MySQL unterstützt.

5 Anwendungen

Aufgrund der unterschiedlichen Mächtigkeit in der Geodatenbankfunktionalität unterscheidet sich der Einsatz der beiden Open-Source-Systeme deutlich voneinander. Da MySQL auf vielen Web Servern zur Verfügung steht, sind räumliche Webanwendungen dessen Haupteinsatzfeld. Dabei müssen allerdings gewisse Voraussetzungen gegeben sein: Entweder sollten die Anforderungen an die geometrische Anfragebearbeitung gering sein, so dass z.B. Rechteckanfragen auf MUR-Approximationen hinreichend sind oder man ist alternativ (funktional und seitens des Antwortzeitverhaltens) in der Lage, die Filterergebnisse von MySQL weiter zu verarbeiten; z.B. mittels Java-Servlets oder -Applets, die die benötigte Funktionaliät nachbilden bzw. aus Geometriebibliotheken beziehen.

Ein Beispiel für Anwendungen mit geringen Anforderungen hinsichtlich der Geometrieverarbeitung sind typischerweise Mashups. Abbildung 6 zeigt die Webanwendung »WikiByMap« (http://de.wikibymap.org), die Wikipedia-Artikel in einer Karte mittels Google Maps bereitstellt. Dazu muss die zugrunde liegende MySQL-Datenbank im Wesentlichen zwei Aufgaben übernehmen: 1.) die Auswahl von georeferenzierten Wikipedia-Artikeln in einem Anfragerechteck und 2.) die Identifikation von Artikeln, die die gleichen Koordinaten aufweisen. Beides ist mit MySQL problemlos möglich, da die Georeferenzierung nur über Punkte erfolgt. Somit kann eine Anwendung wie »WikiByMap« mit den normalen Mitteln, die ein Web-Provider den Kunden zur Verfügung stellt, betrieben werden.

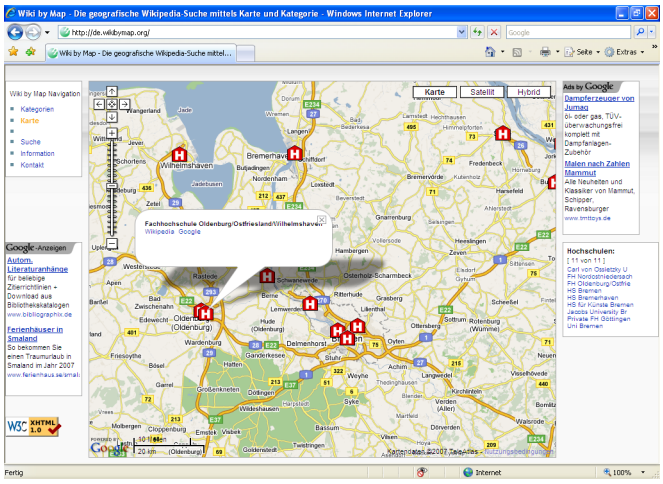

Abb. 6: Mashup »WikiByMap« von Wikipedia und Google Maps auf Basis von MySQL

Ein Beispiel für das zweite Anwendungsszenario ist die dynamische Visualisierung von Nahverkehrsbussen in Verkehrsleitstellen [Englich 2007]. Hierfür wurde ebenfalls MySQL als Datenbanksystem eingesetzt. Die Anwendung erforderte aber (u.a.) auch die Verschneidung von Geometrien und Koordinatentransformationen. Dies wurde in Servlets unter Verwendung der Open-Source-Bibliotheken JTS (http:/www.jump-project.org/project.php?PID=JTS&SID=OVER) und GeoTools (http://www.geotools.org) realisiert.

PostgreSQL/PostGIS dominiert hingegen im Umfeld der OGC-konformen Geodatendienste. Insbesondere der Geodatendienst WFS (*Web Feature Service*,

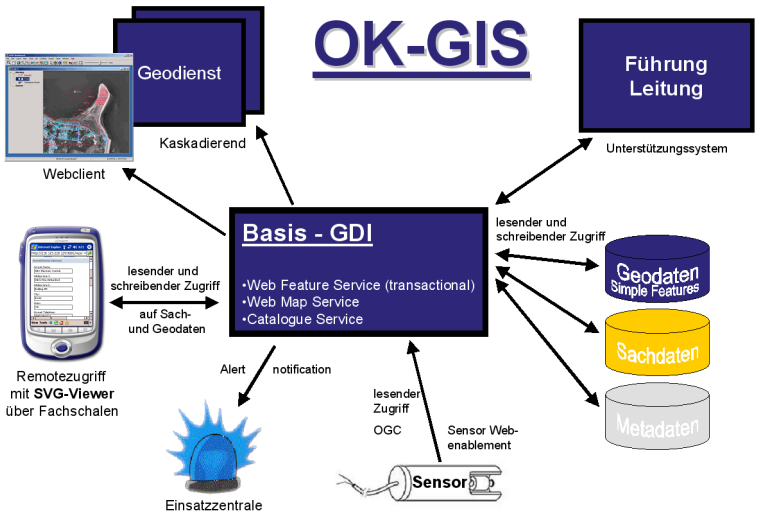

Abb. 7: Aufgaben einer Geodateninfrastruktur im Rahmen des Katastrophenmanagements

[OGC 2004]) ist auf die Abdeckung der vollen Funktionalität des Simple Feature Models angewiesen, da er über das OGC Filter Encoding nicht nur Bedingungen auf Sachattributen, sondern auch komplexe räumliche Anfragebedingungen unterstützen muss. Diese Notwendigkeit gewinnt an Bedeutung, da vielfach andere raumbezogene OGC-Webdienste wie der »Web Map Service« oder »Web Processing Services« auf dem WFS aufsetzen.

Ein typisches Anwendungsbeispiel für die Nutzung von PostgreSQL/PostGIS stellt das Projekt *»OK-GIS«* dar (http://www.okgis.de/, Abbildung 7), das eine Geodateninfrastruktur für das Katastrophenmanagement auf Basis von freien Software-Komponenten entwickelt. So werden nicht nur Kartendaten, sondern auch Einsatzdaten, Mitteilungen, Straßenetze für Routing-Zwecke und Sensordaten als Objekte mit Raumbezug aufgefasst und über PostGIS verwaltet. Neben den genannten Standarddiensten setzen auch anwendungsspezifische Webdienste auf PostgreSQL/PostGIS auf. In dem Projekt hat sich PostGIS als robustes und hinreichend mächtiges und effizientes Geodatenbanksystem bewährt; die Kopplung mit »deegree«, einem Open-Source-Framework für OGC-Geodienste (http://www.deegree.org), ist einfach und problemlos möglich.

6 Fazit

Der Beitrag macht deutlich, dass die bisherigen Geodatenbanksysteme mit den Open-Source-Systemen eine ernsthafte Konkurrenz bekommen haben. Insbesondere die Kombination PostgreSQL mit PostGIS überzeugt durch ihren Leistungsumfang; für viele Webanwendungen ist sogar die Funktionalität von MySQL ausreichend.

Schwächen bestehen noch im Bereich der Nutzung von räumlichen Bezugssystemen und zum Teil bei der Umsetzung von SQL/MM Spatial und der Anfragebearbeitung (Nächste-Nachbarn-Anfrage, Formulierung der Anfragen). Viele Geoinformationssysteme, die nicht aus dem Open-Source-Bereich stammen (z.B. ESRI ArcGIS), erlauben die direkte Nutzung von Open-Source-Geodatenbanksystemen bislang nicht; hierfür muss man sich mit entsprechender Adapter-Software behelfen.

Bei Aufgabenbereichen, die von den Standards nicht oder erst seit kurzem abgedeckt werden, fehlt es allerdings (z.B. im Vergleich zu Oracle Spatial) noch an Funktionalität. Hier sind u.a. Netzwerk- und Topologiedatenmodelle, die Speicherung und Abfrage von Rasterdaten sowie die Unterstützung komplexer 3D-Geodaten zu nennen. So ist die Unterstützung von SQL/MM Spatial, von (Netzwerk-) Topologien, von linearen Bezugssystemen und der Nächsten-Nachbarn-Anfrage in der Zielplanung zu PostGIS 2.0 [Ramsey 2007] enthalten.

7 Literatur

[Brinkhoff 2005] *Brinkhoff, T.:* Geodatenbanksysteme in Theorie und Praxis. Wichmann-Verlag, Heidelberg, 2005.

[Brinkhoff 2007a] *Brinkhoff, T.*: Increasing the Fitness of OGC-Compliant Web Map Services for the Web 2.0. 10th AGILE International Conference on Geographic Information Science, Mai 2007, Aalborg, Dänemark. In; Lecture Notes in Geoinformation, Springer, 2007.

[Brinkhoff 2007b] *Brinkhoff, T.*: Räumliche Netzwerk- und Topologiedatenbanken. Datenbank-Spektrum, Heft 21, Mai 2007.

[Clementini & Di Felice 1996] *Clementini, E.; Di Felice, P.*: A Model for Representing Topological Relationships between Complex Geometric Features in Spatial Databases. Information Sciences 90, 1996, S. 121-136.

[Englich 2007] *Englich, T.*: Dynamische Objekt-Visualisierung in Echtzeit für Verkehrsleitstellen auf Basis von Open Source Komponenten. Forum Freie GI-Systeme, Oldenburg, Februar 2007.

[Guttman 1984] *Guttman, A.*: R-trees: A Dynamic Index Structure for Spatial Searching. Proceedings ACM SIGMOD International Conference on Management of Data, Boston, MA, 1984, S. 47-57.

[Hellerstein et al. 1999] *Hellerstein, J.* et al.: The GiST Indexing Project. 1999, http://gist.cs.berkeley.edu/

[Kirchner & Rösler 2007] *Kirchner, K.; Rösler, R.*: Geomania – Vier Geodatenbanken im Vergleich. iX 1/2007, S. 52-56.

[Lemmen 2007] *Lemmen, C.*: Product Survey on Geo-databases. GIM International, Mai 2007, S. 40-41.

[MySQL 2007] *MySQL AB*: MySQL 5.1 Reference Manual. Mai 2007, http://dev.mysql.com/doc/refman/5.1/en/index.html

[OGC 2004] *Open Geospatial Consortium, Inc.*: Web Feature Service (WFS) Implementation Specification, Version 1.1.0, OGC 04-094.

[Oracle 2005] *Oracle Corp.*: Oracle Spatial – User's Guide and Reference, 10g Release 2, Juni 2005.

[Orenstein 1989] *Orenstein, J.A.*: Redundancy in Spatial Databases. Proceedings ACM SIGMOD International Conference on Management of Data, Portland, OR, 1989, S. 294-305.

[PostgreSQL 2007] *The PostgreSQL Global Development Group*: PostgreSQL 8.2.4 Documentation. 2007, http://www.postgresql.org/docs/8.2/static/index.html

[Ramsey 2007] *Ramsey, P.*: The Road to PostGIS 2.0. http://docs.google.com/View?docid=dg99qr76_2dgt26j

[Refractions 2007] *Refractions Research*: PostGIS Manual, Version 1.2.1. Januar 2007, http://postgis.refractions.net/docs/

[Rigaux et al. 2002] *Rigaux, P.; Scholl, M.; Voisard, A.*: Spatial Databases, With Applications to GIS. Morgan Kaufmann Publishers, San Francisco, 2002.

[S&P 2007] *OPG Surveying & Positioning Committee*: EPSG Geodetic Parameter Dataset, Version 6.12. Februar 2007, http://www.epsg.org/Geodetic.html

[Tomlinson 1984] *Tomlinson, R.F.*: Geographic Information Systems – A New Frontier, The Operational Geographer/La Géographie Appliquée, No. 5, 1984, S. 31-36.

[Wikipedia 2006] *Wikipedia*: Mashup (web application hybrid), http://en.wikipedia.org/wiki/Mashup_(web_application_hybrid)

Prof. Dr. Thomas Brinkhoff
FH Oldenburg/Ostfriesland/Wilhelmshaven
Institut für Angewandte Photogrammetrie und Geoinformatik (IAPG)
Ofener Str. 16/19
26121 Oldenburg
thomas.brinkhoff@fh-oldenburg.de
http://www.fh-oow.de/institute/iapg/personen/brinkhoff/