Ludwig Maximilians Universität München
Institut für Informatik
Lehr- und Forschungseinheit für Datenbanksysteme

Vorlesung

Datenbanksysteme I

Wintersemester 2014/2015

Kapitel 12: Objekt-relationale Erweiterungen

Vorlesung: PD Dr. Arthur Zimek
Übungen: Sebastian Goebl, Jan Stutzki

http://www.dbs.ifi.lmu.de/Lehre/DBS

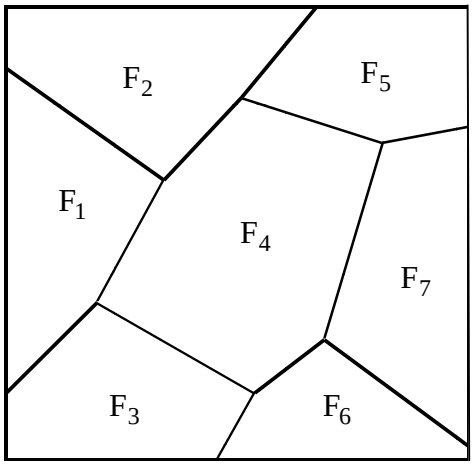
Grenzen des relationalen Modells

- Nichtstandard Anwendungen, z.B.
  - CAD / CAM / CIM
  - Geographie und Kartographie
  - Medizin und Biologie
  - Multimedia
- komplexe, unterschiedlich strukturierte Objekte
- Daten mit sehr großen Attributswerten (z.B. Multimedia-Inhalte)
- komplexe Integritätsbedingungen (z.B. aus der Geographie)
- häufig sehr lange Transaktionen auf wenigen Objekten (z.B. CAD)

Grenzen des relationalen Modells

- Beispiel-Anwendung: Darstellung geometrischer Objekte (hier Parzellen) in normalisierten Relationen:

  Eine redundanzfreie Repräsentation der Parzellen erfordert die Verteilung der Informationen auf drei Relationen: „Parzellen“, „Kanten“ und „Punkte“:



Parzellen	
FNr	KNr
F_1	K_1
F_1	K_2
F_1	K_3
F_1	K_4
F_4	K_2
F_4	K_5
F_4	K_6
F_4	K_7
F_4	K_8
F_4	K_9
F_7	K_7
F_7	K_{10}
F_7	K_{11}
F_7	K_{12}

Kanten		
KNr	PNr_1	PNr_2
K_1	P_1	P_2
K_2	P_2	P_3
K_3	P_3	P_4
K_4	P_4	P_1
K_5	P_2	P_5
K_6	P_5	P_6
K_7	P_6	P_7
K_8	P_7	P_8
K_9	P_8	P_3
K_{10}	P_6	P_9
K_{11}	P_9	P_{10}
K_{12}	P_{10}	P_7

Punkte		
PNr	X-Koord.	Y-Koord.
P_1	X_{P1}	Y_{P1}
P_2	X_{P2}	Y_{P2}
P_3	X_{P3}	Y_{P3}
P_4	X_{P4}	Y_{P4}
P_5	X_{P5}	Y_{P5}
P_6	X_{P6}	Y_{P6}
P_7	X_{P7}	Y_{P7}
P_8	X_{P8}	Y_{P8}
P_9	X_{P9}	Y_{P9}
P_{10}	X_{P10}	Y_{P10}

Grenzen des relationalen Modells

- Sollen jetzt (geometrische) Anfragen auf den Parzellen bearbeitet werden, so müssen die erforderlichen Informationen zunächst stets wieder zusammengesetzt werden
- *Beispiel:* Gesucht sind alle Eckpunkte der Parzelle mit Flurnummer 2

```
select   Punkte.PNr, X-Koord, Y-Koord
from     Parzellen, Kanten, Punkte
where    FNr = “F2”
         and
         Parzellen.KNr = Kanten.KNr
         and
         (Kanten.PNr1 = Punkte.PNr or Kanten.PNr2 = Punkte.PNr)
```

Grenzen des relationalen Modells

- Obwohl die Geometrie des Objektes noch gar keine Rolle gespielt hat, benötigen wir zur Bearbeitung dieser einfachen Anfrage zwei oder drei Joins.
- Ursache: Art der Datenmodellierung
  - Die Zerlegung der Objekte in einfache Komponenten (Flächen, Linien und Punkte) ergibt ein normalisiertes Datenbankschema
  - Die Informationen über ein Objekt (Parzelle) liegt aber nun über mehrere Relationen („Parzellen“, „Kanten“ und „Punkte“) verteilt

Grenzen des relationalen Modells

- Schwächen
  1. Aufspaltung der logischen Objekte:

     Ein logisches Objekt der Anwendung muss in mehrere Tupel verschiedener Relationen aufgespalten werden
     - Das macht alle Operationen auf den logischen Objekten komplex
     - Bei Updates des Objekts müssen alle zugehörigen Tupel aktualisiert werden
     - Zur Bearbeitung von Anfragen müssen die Objekte durch teure Joins aus den einzelnen Tupeln zusammengesetzt werden.

Grenzen des relationalen Modells

2. Keine Beziehungen zwischen den Relationen

   Beziehungen zwischen Objekten sollen modelliert werden können (z.B. Vererbung, um gemeinsame Eigenschaften wieder verwenden zu können)

   Beispiel: Rechtecke, Kreise, Polygone sind Spezialfälle von Flächen

Grenzen des relationalen Modells

3. Keine Modellierung des Verhaltens

   Relationales Modell behandelt nur die Struktur, nicht das Verhalten von Objekten, d.h. es werden nur Standard-Operationen zur Anfrage und zum Update von einzelnen Tupeln angeboten, aber keine anwendungs- bzw. objektspezifischen Operationen.

   – Objekt-Semantik ist in Anwendungsprogrammen versteckt
     - » mehrere Anwendungsprogramme für dieselbe Operation
     - » möglicherweise verschiedene Semantiken für dieselbe Operation
     - » *Beispiel*

       Für Polyeder sind u.a. folgende Operationen zu definieren:

       skaliere (Faktor)
       bewege (X, Y, Z)
       rotiere (Achse, Winkel)
       …

Grenzen des relationalen Modells

4. Problematische Schnittstelle DML-Programmiersprache

   Eine relationale DML, wie z.B. SQL, liefert Mengen von Tupeln als Antworten. Eine Programmiersprache, wie z.B. Java, kann jedoch nur ein Tupel auf einmal verarbeiten. Abhilfe schafft das sog. Cursor-Konzept um die Antworten einzeln zu verarbeiten.

   Ein Cursor besitzt jedoch folgende, für Non-Standard Anwendungen gravierende, Schwächen:

   - nur sequentieller Zugriff auf die Tupel in der gegebenen Reihenfolge (nur genau einmal); bei interaktiven Anwendungen ist jedoch meist wahlfreier Zugriff erforderlich
   - i.A. keine Updates möglich, z.B. wenn ein Cursor mit einer Anfrage assoziiert ist, die einen Join enthält, da es unentscheidbar ist, wie ein Update sich auf die zugrundeliegenden Relationen auswirkt.

Grenzen des relationalen Modells

- Zusammenfassung
  - Information über logische Objekte auf mehrere Relationen verteilt
  - Kein Objektverhalten
  - Keine Wiederverwendung von Code
  - Verwaltung verschiedener Repräsentationen desselben Objekts, die bei Updates alle miteinander zu ändern sind
  - Einhaltung von Konsistenzbedingungen aus der Anwendung
  - Abspeicherung von Objekten mit großen Attributwerten (z.B. Multimedia-Inhalte)
  - …

Vom relationalen zum objekt-relationalen Model

- Lösungsmöglichkeit 1: NF^2-Relationen (NF^2 = *non first normal form*)
  - Die erste Normalform des relationalen Modells wird umgangen, d.h. auch nicht-atomare Werte (Relationen) sind als Attribute zuzulassen
  - Beispiel: Parzellen

Parzellen				
	Kanten			
		Punkte		
FNr	KNr	PNr	X-Koord.	Y-Koord.
F_1	K_1	P_1	X_{P1}	Y_{P1}
		P_2	X_{P2}	Y_{P2}
	K_2	P_2	X_{P2}	Y_{P2}
		P_3	X_{P3}	Y_{P3}
	K_3	P_3	X_{P3}	Y_{P3}
		P_4	X_{P4}	Y_{P4}
	K_4	P_4	X_{P4}	Y_{P4}
		P_1	X_{P1}	Y_{P1}
F_4	K_2	P_2	X_{P2}	Y_{P2}
		P_3	X_{P3}	Y_{P3}
	K_5	P_2	X_{P2}	Y_{P2}
		P_5	X_{P5}	Y_{P5}
	K_6	P_5	X_{P5}	Y_{P5}
		P_6	X_{P6}	Y_{P6}
	K_7	P_6	X_{P6}	Y_{P6}
		P_7	X_{P7}	Y_{P7}
	K_8	P_7	X_{P7}	Y_{P7}
		P_8	X_{P8}	Y_{P8}
	K_9	P_8	X_{P8}	Y_{P8}
		P_3	X_{P3}	Y_{P3}
F_7	K_7	P_6	X_{P6}	Y_{P6}
		P_7	X_{P7}	Y_{P7}
	K_{10}	P_6	X_{P6}	Y_{P6}
		P_9	X_{P9}	Y_{P9}
	K_{11}	P_9	X_{P9}	Y_{P9}
		P_{10}	X_{P10}	Y_{P10}
	K_{12}	P_{10}	X_{P10}	Y_{P10}
		P_7	X_{p7}	Y_{P7}

Vom relationalen zum objekt-relationalen Model

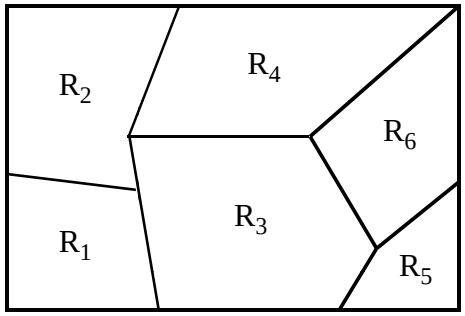
- Vorteile
  - Vermeidung von Joins durch Abspeicherung aller Informationen in einer Relation → höhere Effizienz bei der Anfragebearbeitung.
  - *Geclusterte* Abspeicherung der relationenwertigen Attribute wird möglich → weitere Effizienzsteigerung
- *Aber:*
  - Keine echte Unterstützung von Anfragen, die sich auf die Geometrie der Objekte beziehen
  - Kein Objektverhalten
  - Konsistenzbedingungen???
  - Große Attributwerte???

Vom relationalen zum objekt-relationalen Model

- Lösungsmöglichkeit 2: Objekt-relationale Erweiterung
  - Idee: Integration benutzerdefinierter Objekte und Operationen als elementare Datentypen in den Kern eines Datenbanksystems.
  - Durch Aufgreifen dieser Idee lassen sich Wertebereiche wie z.B. *Rechteck* oder *Polygon* realisieren, die eine adäquate Modellierung geometrischer Szenen ermöglichen und objekt-spezifische Methoden bereitstellen.

Vom relationalen zum objekt-relationalen Model

– Beispiel: Forstgebiete (Revier: Polygon, Förster: String, Fläche: Integer)



Forstgebiete		
Revier	Förster	Fläche (m^2)
R_1	Schmidt	3900
R_2	Behrens	4250
R_3	Schultz	6700
R_4	Schmidt	5400
R_5	Meier	1900
R_6	Schröder	4600

– Durch diese Art der Datenmodellierung werden Anfragen folgender Art möglich:

In einer Schonung definiert durch das Polygon *S* soll eine Neuanpflanzung durchgeführt werden. Welche Förster sind von der Aktion betroffen?

select Förster
from Forstgebiete
where ObjectIntersects(S, Revier)

Vom relationalen zum objekt-relationalen Model

- Innerhalb des Systems steht Datentyp *Polygon* mit dem Prädikat *ObjectIntersects* zur Verfügung, das testet, ob sich zwei Polygone schneiden

– Vorteile dieser Datenmodellierung

- Integration geometrischer Datentypen erfordert lediglich eine *Erweiterung* des bestehenden relationalen Modells sowie der zugehörigen Datenbanksprachen
- Geometrische Attribute werden als *eigenständige* Datentypen aufgefasst. Dies erleichtert den Datenbankentwurf und die Formulierung von Anfragen.
- Geometrische Datentypen (Wertebereiche und Operationen) können durch effiziente Datenstrukturen und Algorithmen im Kern eines Datenbanksystems realisiert werden

Einige objekt-relationale Konzepte

- Mengenwertige Attribute
  - Attribut kann Menge von Werten als Domain besitzen
  - Anfragesprache muss Schachtelung (Bildung) bzw. Entschachtelung („Flachklopfen“) von Mengen unterstützen
- Geschachtelte Relationen
  - Attributwerte können selbst wieder Relationen sein
- Benutzerdefinierte Datentypen
  - Wertebasierte, abstrakte Datentypen können nur als Komponente eines Tupels vorkommen
  - Tupeltypen (auch Objekttypen, *row types*) können als eigenständiger Datensatz in einer Relation vorkommen

Einige objekt-relationale Konzepte

- Referenzen
  - Attributwert ist direkte Referenz auf Tupel/Objekte
  - Dadurch keine Fremdschlüssel mehr nötig, auch *n:m*-Beziehung nun ohne eigenständige Relation umsetzbar
- Objektidentität
  - Zur Identifikation von Objekten
  - Nicht veränderbar (im Ggs. zum Schlüssel)
- Vererbung
  - Generalisierung/Spezialisierung
- Operationen
  - Modellierung objektspezifischen Verhaltens
- Große Objekte (LOBs = Large OBjects)
  - Speicherung von großen Attributwerten (GB-Bereich)

Large Objects (LOBs)

- Arten
  - CLOB (Character LargeOBjects)
    - Speicherung langer Texte
    - Verbesserter Zugriff im Vergleich zu **varchar()**
  - BLOB (Binary LargeOBjects)
    - Werden vom DBMS nicht interpretiert sondern nur gespeichert (als Bitsequenz), z.B. Videos, etc.
  - NCLOB (National Character LargeOBjects)
    - Für Texte mit Sonderzeichen (z.B. Unicode) da CLOBs nur Texte mit 1-Byte Kodierung speichert
- Wichtig:
  - Bei Verarbeitung von LOB-Daten vermeide Transfer der Daten vom DBS zum Anwendungsprogramm (insbesondere in Client/Server-Szenarien) => *Locator*-Operationen

Large Objects (LOBs)

- Beispiel

```
create table Professoren
(
    PersNr          integer primary key,
    Name            varchar(30) not null,
    Rang            character(2)
                        check Rang in (`C2´, `C3´, `C4´, `W1´, `W2´, `W3´),
    Raum            integer unique,
    Passfoto        BLOB(2M),
    Lebenslauf      CLOB(75K)
);
```

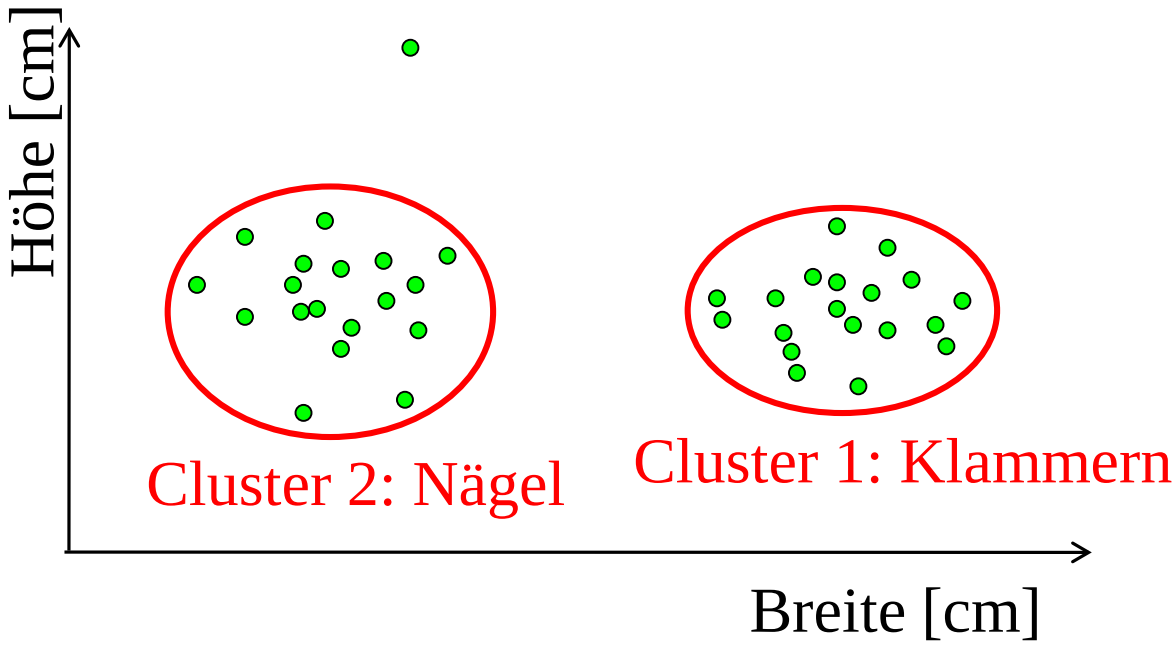
- Herstellerspezifische Features
  - LOB-Attribute vom Logging freistellen
  - Gesonderten Tablespace für Speicherung definieren
  - Optionale komprimierte Speicherung

Ausblick

weitere Vorlesungen vertiefen verschiedene Konzepte:

- DBS II (SoSe 2015, Kröger):
  - Transaktionskontrolle
  - Anfragebearbeitung
  - Datensicherheit
- Anfragebearbeitung und Indexstrukturen in Datenbanksystemen
- Datenbankpraktikum
- Geo-Informationssysteme
- Spatial, Temporal, and Multimedia Databases
- Managing Massive Multiplayer Online Games
  (SoSe 2015, Schubert)
- Knowledge Discovery in Databases (KDD)
  (SoSe 2015, Zimek)

Ausblick: KDD

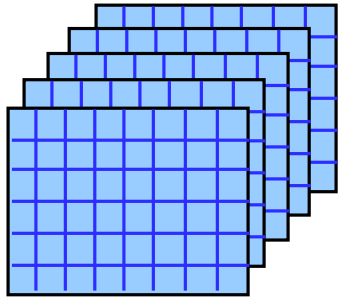


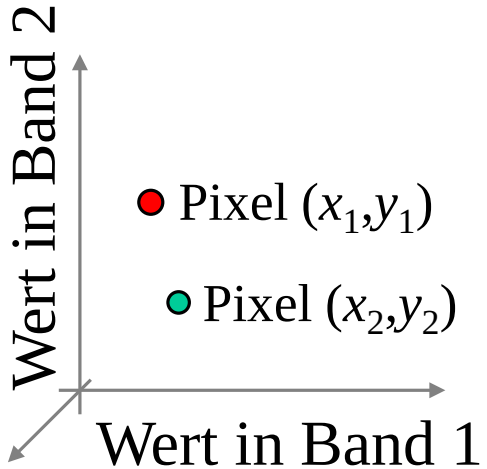
Clustering heißt: Zerlegung einer Menge von Objekten (bzw. Feature-Vektoren) in Teilmengen (Cluster) ähnlicher Objekte

Idee: Die verschiedenen Cluster repräsentieren meist unterschiedliche Klassen von Objekten; bei unbek. Anzahl und Bedeutung der Klassen

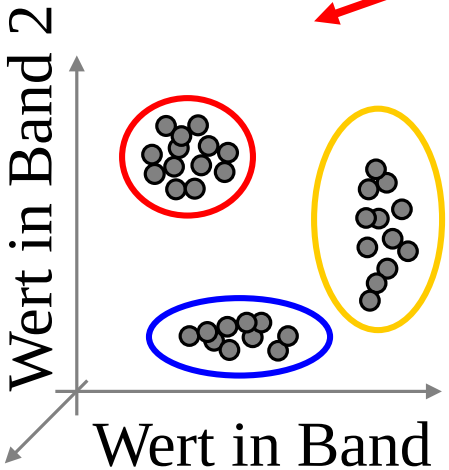
Ausblick: KDD

Aufnahme der Erdoberfläche
in 5 verschiedenen Spektren



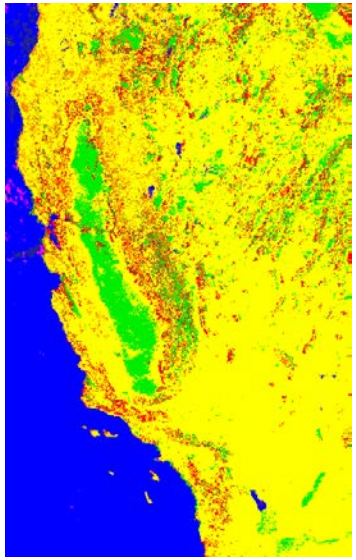


Cluster-Analyse

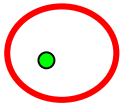


Rücktransformation
in *xy*-Koordinaten

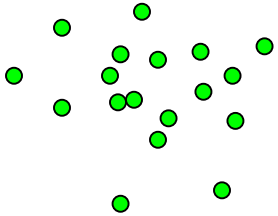
Farbcodierung nach
Cluster-Zugehörigkeit

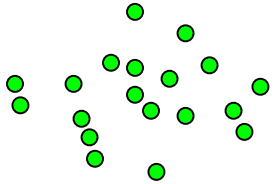


Ausblick: KDD



Datenfehler?
Betrug?



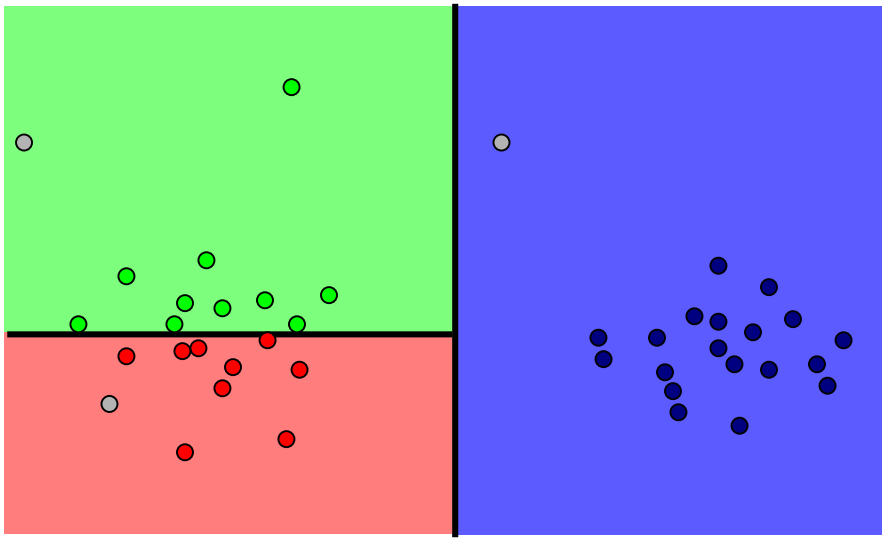


Outlier Detection bedeutet:
Ermittlung von untypischen Daten

Idee: Outlier könnten hindeuten auf

- Missbrauch etwa bei
  - Kreditkarten
  - Telekommunikation
- Datenfehler

Ausblick: KDD



- Schrauben
- Nägel
- Klammern

} Trainings-daten

- Neue Objekte

Aufgabe:
Lerne aus den bereits klassifizierten *Trainingsdaten* die *Regeln*, um neue Objekte nur aufgrund der Merkmale zu klassifizieren

Das Ergebnismerkmal (Klassenvariable) ist nominal (*kategorisch*)

Klausur

- Merkblatt zur Klausurankündigung beachten
- weitere organisatorische Informationen (Räume) zur Klausur zu gegebener Zeit auf der Vorlesungswebseite
- Anmelden nicht vergessen!

Viel Erfolg!

- nächster Freitag: Klausurbesprechung