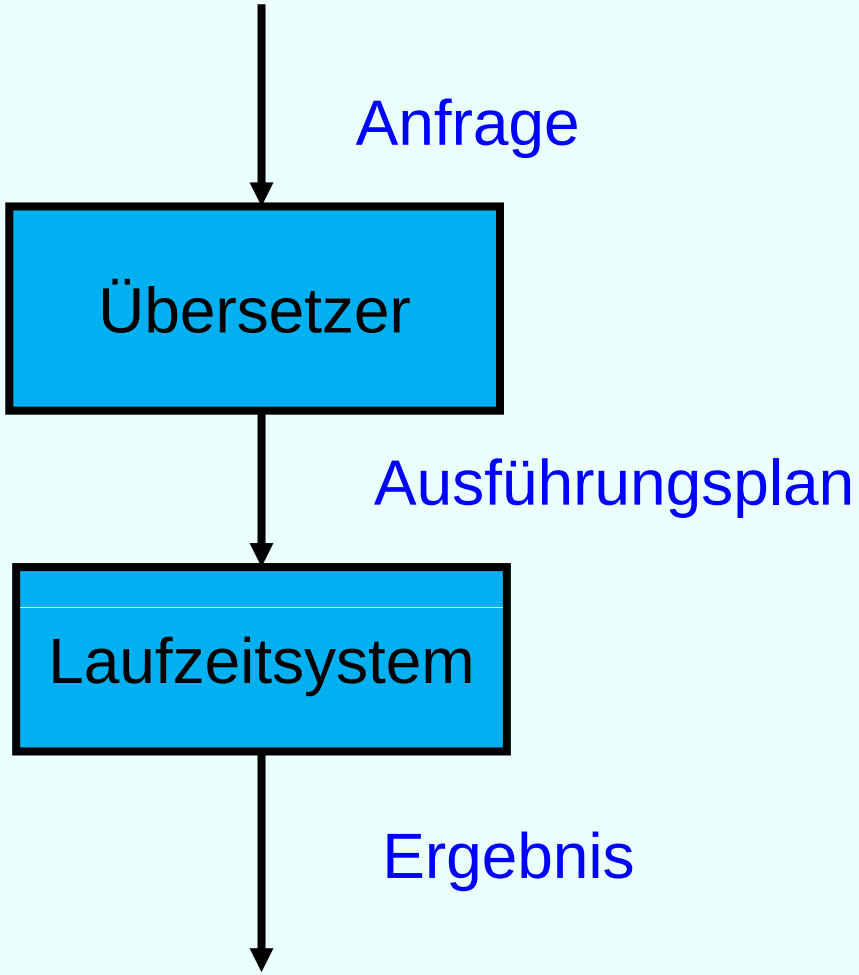
Anfragebearbeitung



Übersetzung

- SQL ist deklarativ, Übersetzung für Laufzeitsystem in etwas prozedurales
- DBMS übersetzt SQL in eine interne Darstellung
- weit verbreiteter Ansatz ist Übersetzung in eine relationale Algebra

Kanonische Übersetzung

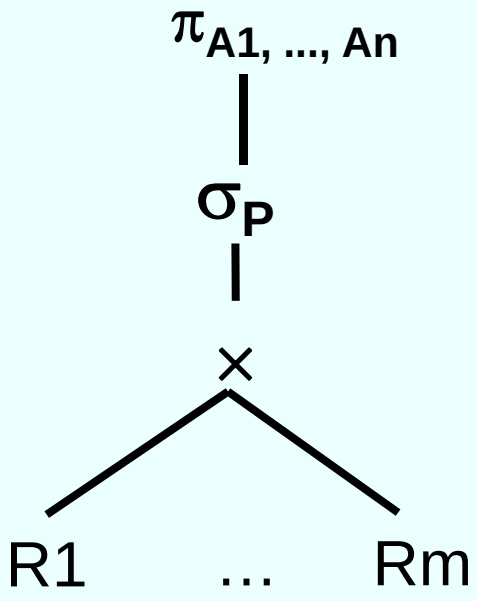
- Standardübersetzung von SQL in relationale Algebra
- Algebra-Ausdrücke werden oft graphisch repräsentiert

- Beispiel

  select *A*1, . . . ,*An*

  from *R*1, . . . , *Rm*

  where *p*



Optimierung

- Kanonischer Plan nicht effizient,
  z.B. Kreuzprodukt
- DBMS besitzt Optimierer zur Überführung des Plans in eine effiziente Form
- Finden eines optimalen Plans sehr schwieriges Problem:
  immer noch Gegenstand aktueller Forschung

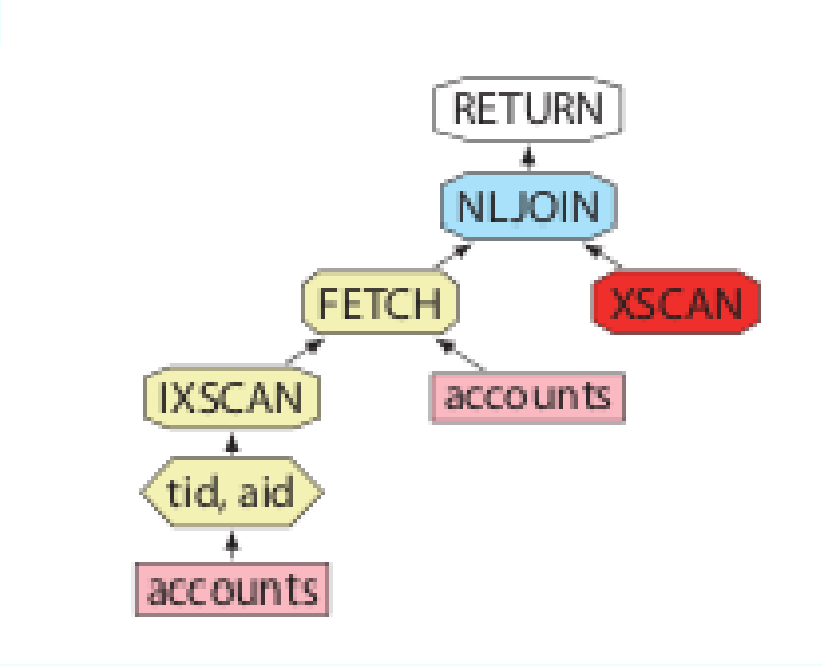
Optimierung(2)

- Benutzer ↔ Anfrageoptimierung?

- Benutzer können
  - o generierten Plan einsehen
  - o generierten Plan analysieren
  - o (gegebenenfalls Anfrage umbauen oder DBMS Hinweise zur Ausführung geben)

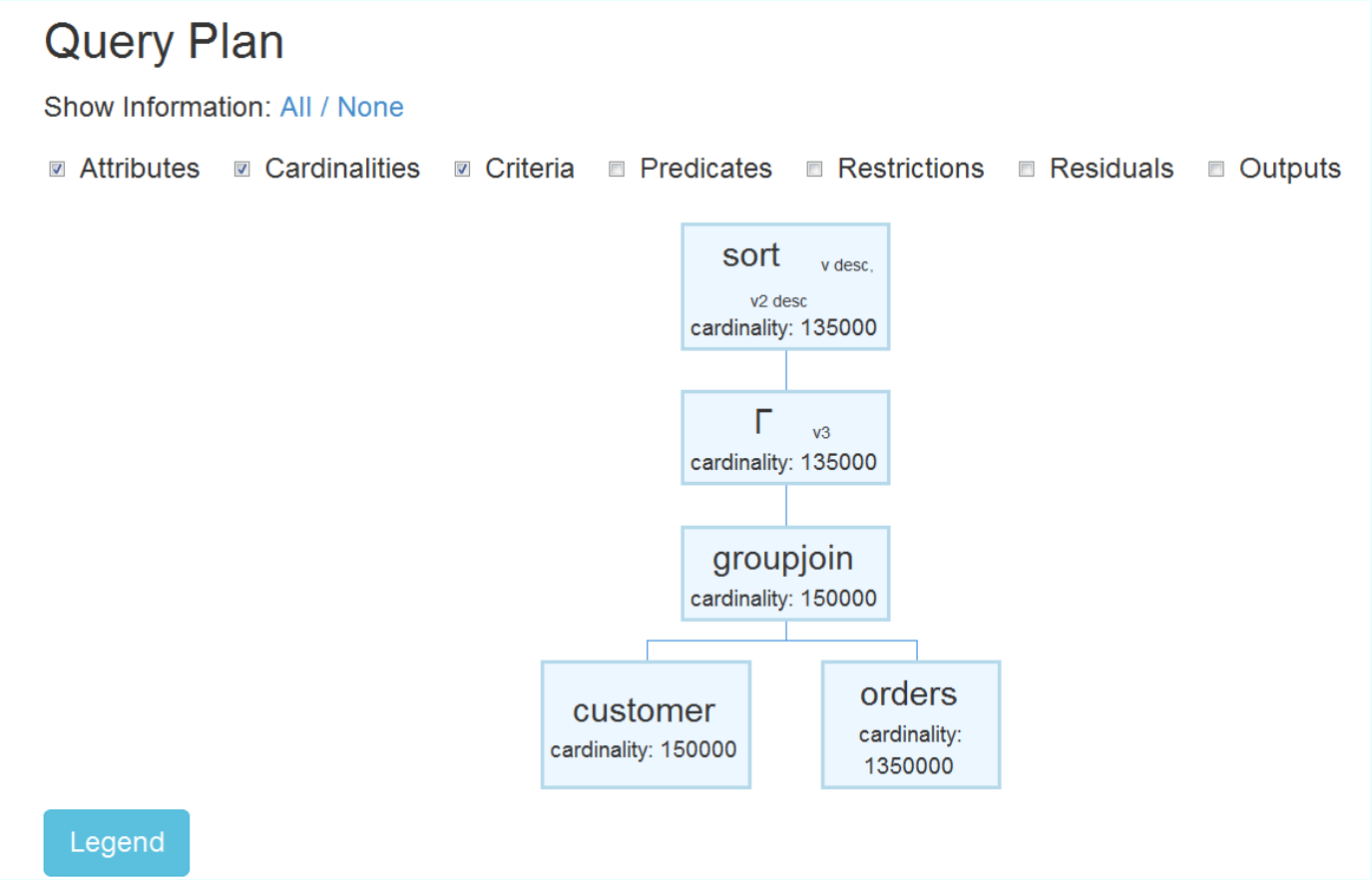
Visualisierung von Plänen

PostgreSQL:



Visualisierung von Plänen

Hyper:



Anfrageoptimierung 1x1

- Anfrageoptimierung ist kostenbasiert
- Abschätzung der Kosten mit Hilfe von Kostenmodellen und Statistiken
- Anwendung von Heuristiken
  alle möglichen Pläne anzuschauen viel zu teuer
- Zwei Ebenen der Optimierung:
  - Logische Ebene
  - Physische Ebene

Logische Ebene

- Ausgangspunkt: relationaler Algebra-Ausdruck
  entstanden nach kanonischer Übersetzung
- Optimierung: Transformation in äquivalente Ausdrücke (mit schnellerer Ausführung)
- Ziel der Umformungen: Ausgaben (Ergebnisse) der einzelnen Operatoren möglichst klein

Logische Ebene(2)

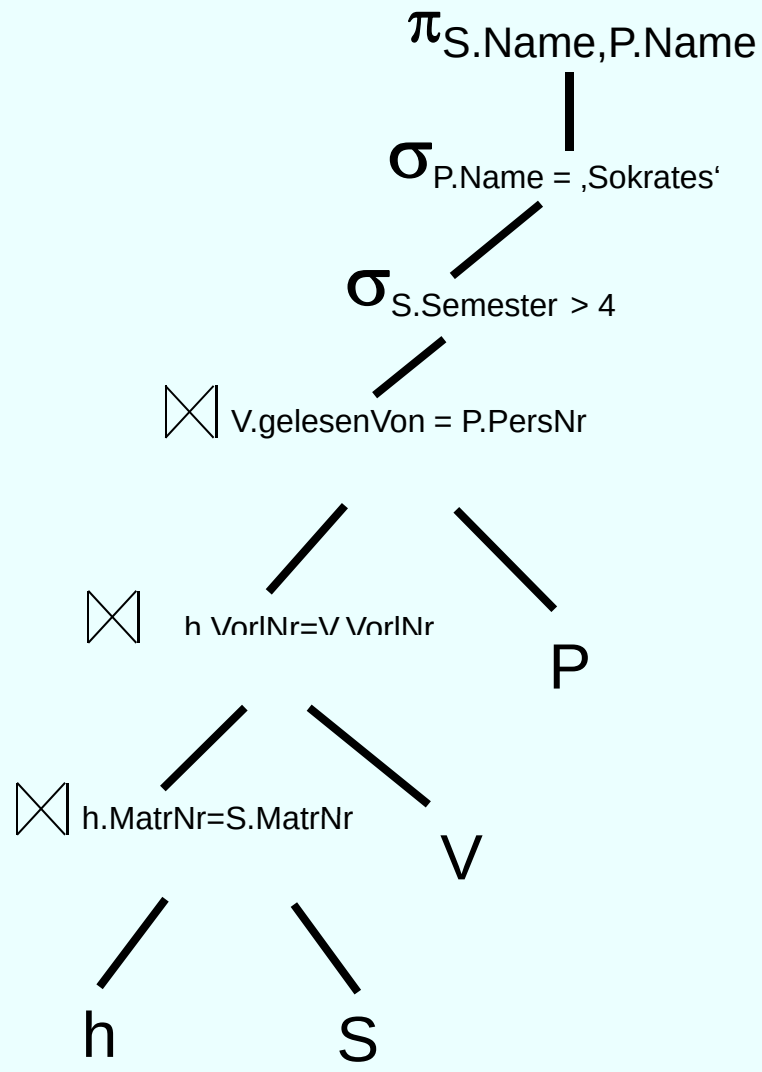
Grundlegende Techniken - **Regeln**:

- Aufbrechen von Selektionen
- Verschieben von Selektionen nach "unten" im Plan
- Zusammenfassen von Selektionen und Kreuzprodukten zu Joins
- Bestimmung der Joinreihenfolge
- Einfügen von Projektionen
- Verschieben von Projektionen nach "unten" im Plan

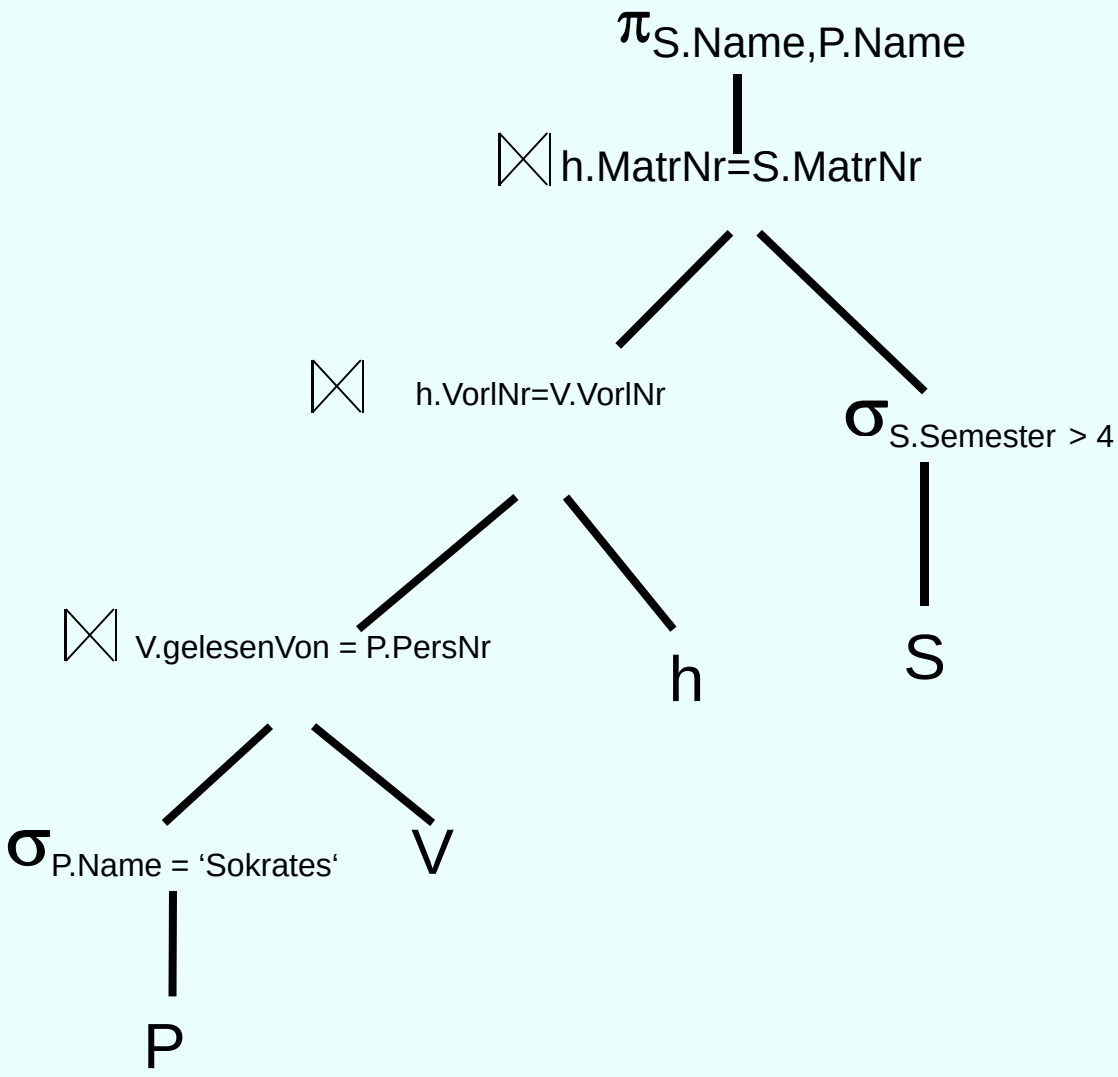
Beispielanfrage

```
select S.Name, P.Name
from Studenten S, hoeren h, Vorlesungen V,
   Professoren P
where S.MatrNr = h.MatrNr
and h.VorlNr = V.VorlNr
and V.gelesenVon = P.PersNr
and S.Semester > 4
and P.Name = 'Sokrates';
```

Anfrageplan



Optimierter Anfrageplan



Physische Optimierung

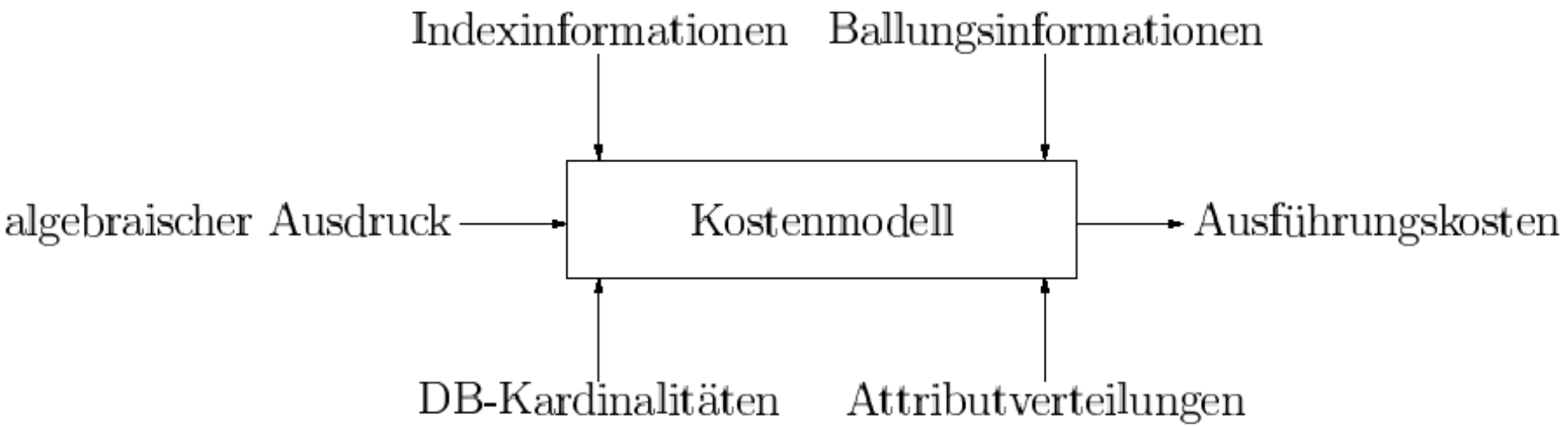
- Unterscheidung logische und physische Algebra-Operatoren
- physische Algebra-Operatoren stellen Realisierung der logischen dar
- mehrere physische Operatoren für einen logischen Operator möglich
- Optimierung auf der physischen Ebene bedeutet:
  - einen dieser Operatoren auszuwählen
  - zu entscheiden, ob Indexe benutzt werden sollen
  - Zwischenergebnisse zu materialisieren, etc.

Implementierung Operatoren

- Selektion:
  - Scan
  - Indexscan

- Join:
  - Nested-Loop-Join
  - (Sort-)Merge-Join
  - Index-Join
  - Hash-Join

Kostenmodelle



Kostenabschätzungen

- Selektivitäten
  Anteil der qualifizierenden Tupel einer Operation
- Abschätzung der Selektivität durch
  - Formeln
  - Stichprobenverfahren
- Selektionskosten
- Joinkosten
- Joinreihenfolge

→ hierfür braucht man Statistiken

Wann Statistiken sammeln?

Wann Statistiken fortschreiben? (Auszug aus IBM DB2 Manual)

unter anderem:

- Nachdem Daten in eine Tabelle geladen und geeignete Indizes erstellt wurden.
- Nachdem ein neuer Index für eine Tabelle erstellt wurde.
- Nachdem eine Tabelle mit dem Dienstprogramm REORG reorganisiert wurde.
- Nachdem eine Tabelle und die zugehörigen Indizes durch UPDATE-, INSERT- oder DELETE-Operationen in erheblichem Umfang geändert wurden.
- Vor dem Binden von Anwendungsprogrammen, deren Leistung von kritischer Bedeutung ist.
- Nachdem der Befehl REDISTRIBUTE DATABASE PARTITION GROUP ausgeführt wurde.

Welche Statistiken gibt es?

(Auszug aus IBM DB2 Manual)

SYSCAT: read-only view – Systemkatalog

SYSSTAT: änderbarer view - Statistikdaten

- Tabellenstatistiken (SYSCAT/SYSSTAT.**TABLES**)
- Spaltenstatistiken (SYSCAT/SYSSTAT.**COLUMNS**)
- Statistiken für Spaltengruppen (SYSCAT/SYSSTAT.**COLGROUPS**)
- Verteilungsstatistiken für Spaltengruppen (SYSCAT/SYSSTAT.**COLGROUPDIST**)
- Verteilungsstatistiken für Spaltengruppen (SYSCAT/SYSSTAT.**COLGROUPDISTCOUNTS**)
- Indexstatistiken (SYSCAT/SYSSTAT.**INDEXES**)
- Spaltenverteilungsstatistiken (SYSCAT/SYSSTAT.**COLDIST**)

Zusammenfassung

- Anfragebearbeitung und -optimierung sind wichtige Aufgaben eines DBMS
- auch Benutzer sollten Ahnung davon haben, da Entwurfsentscheidungen und Anfrageformulierung einen Einfluss auf die Performanz eines DBMS haben

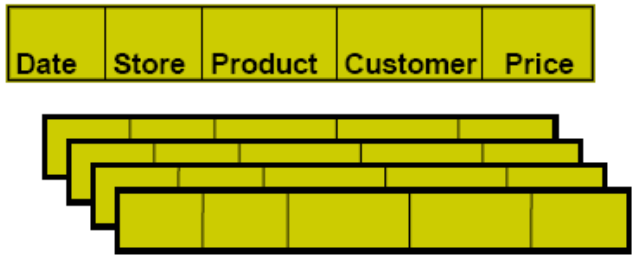
Neue Entwicklungen

- Hauptspeicher-Datenbanksysteme, z.B.
  - Times Ten (Oracle)
  - VoltDB (einige Datenbankforscher, open source)
  - Monet DB (CWI, Amsterdam, open source)
  - TREX (SAP)
  - HYPER (Informatik, TUM)
- Columns Store Datenbanksysteme, z.B.
  - C-Store / Vertica (HP)
  - Monet DB (CWI, Amsterdam, open source)
  - TREX (SAP)
  - HYPER (Informatik, TUM)

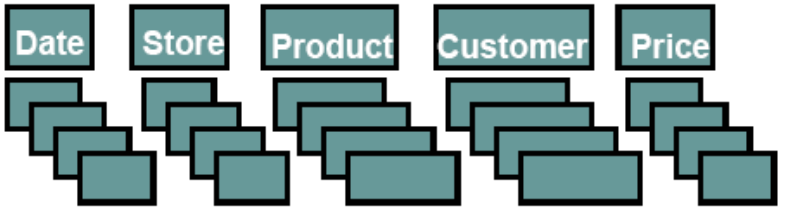
Column Stores

What is a column-store?

row-store



column-store



+ easy to add/modify a record

- might read in unnecessary data

+ only need to read in relevant data

- tuple writes require multiple accesses

=> suitable for read-mostly, read-intensive, large data repositories

Row Store versus Column Store

Verkäufe				
Produkt	Kunde	Preis	Filiale	...
Handy	Kemper	345	Schwabing	...
Radio	Mickey	123	Bogenhausen	...
Handy	Minnie	233	Schwabing	...
Kühlschrank	Urmel	240	Augsburg	...
Beamer	Bond	740	London	...
Handy	Lucie	321	Bogenhausen	...

Row Store versus Column Store

Produkt	
ID	Produkt
0	Handy
1	Radio
2	Handy
3	Kühlschrank
4	Beamer
5	Handy

Kunde	
ID	Kunde
0	Kemper
1	Mickey
2	Minnie
3	Urmel
4	Bond
5	Lucie

Preis	
ID	Preis
0	345
1	123
2	233
3	240
4	740
5	321

Filiale	
ID	Filiale
0	Schwabing
1	Bogenhausen
2	Schwabing
3	Augsburg
4	London
5	Bogenhausen

Komprimierung

Interessant sind insbesondere die Komprimierungmöglichkeiten:

Dictionary	
ID	Wort
0	Augsburg
1	Beamer
2	Bogenhausen
3	Handy
4	Kühlschrank
5	London
6	Radio
7	Schwabing
...	...

Produkt	
ID	Produkt
0	3
1	6
2	3
3	4
4	1
5	3

Filiale	
ID	Filiale
0	7
1	2
2	7
3	0
4	5
5	2

NoSQL

NoSQL
No (kein) SQL - Not only (nicht nur) SQL

Charakteristika: Schema-frei, skalierbar/web scale (aber Aufweichung ACID), verteilt (scale-out), meist key-value store (hash tables: key and pointer to the value), (spezielle Daten, z.B. Graphen)

CAP Theorem:

- Consistency
- Availability
- Partition Tolerance

Nur zwei der drei Zeile lassen sich erreichen

NoSQL cont.

Kurze Erläuterung:
http://www.youtube.com/watch?v=pHAItWE7QMU&list=PLB9uLawXQoggpG9MGz5v9wDodr9f_p4lg

‚Streit' RDBMS – NoSQL (Mongo DB):
http://www.youtube.com/watch?v=b2F-DItXtZs