Datenbanksysteme II: Implementation of Database Systems

Ulf Leser

Preface

- Slides in English, Vortrag auf Deutsch
- Much input from
  - Prof. J-C Freytag, HU Berlin
  - Prof. K-U Sattler, TU Illmenau
  - Prof. A Kemper, Dr. Eickler, TU München
- AGNES for Übung
- Prof. Leser / Prof. Weidlich
- What you should (must) know to follow this course
- What do you study?

Some Motivation

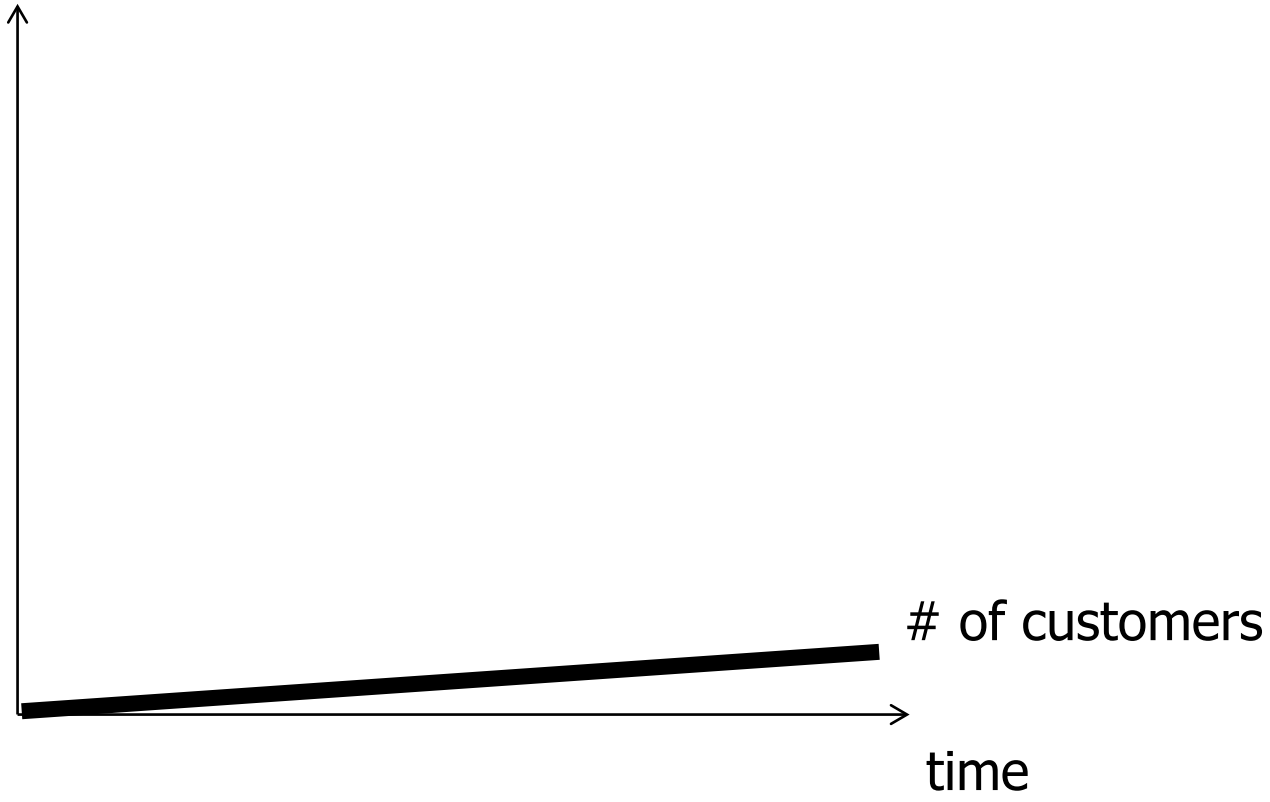
A classical first contact with database

- Company: "I need to track my customers"
  - Name, address, profession, prior contracts
- Naïve engineer: "No problem"
  - ~1984: Turbo Pascal 4.0, Schneider CPC646, 512 KB main memory
  - Each customer one record / line in file
  - Load customers from disk into memory
  - Repeat until "Q"
    - (S)earch and list customers
    - (E)dit customer
    - (D)elete customer
    - (I)nsert customer
    - (Q)uit
  - Write customers to disk
- Invoice: … DM

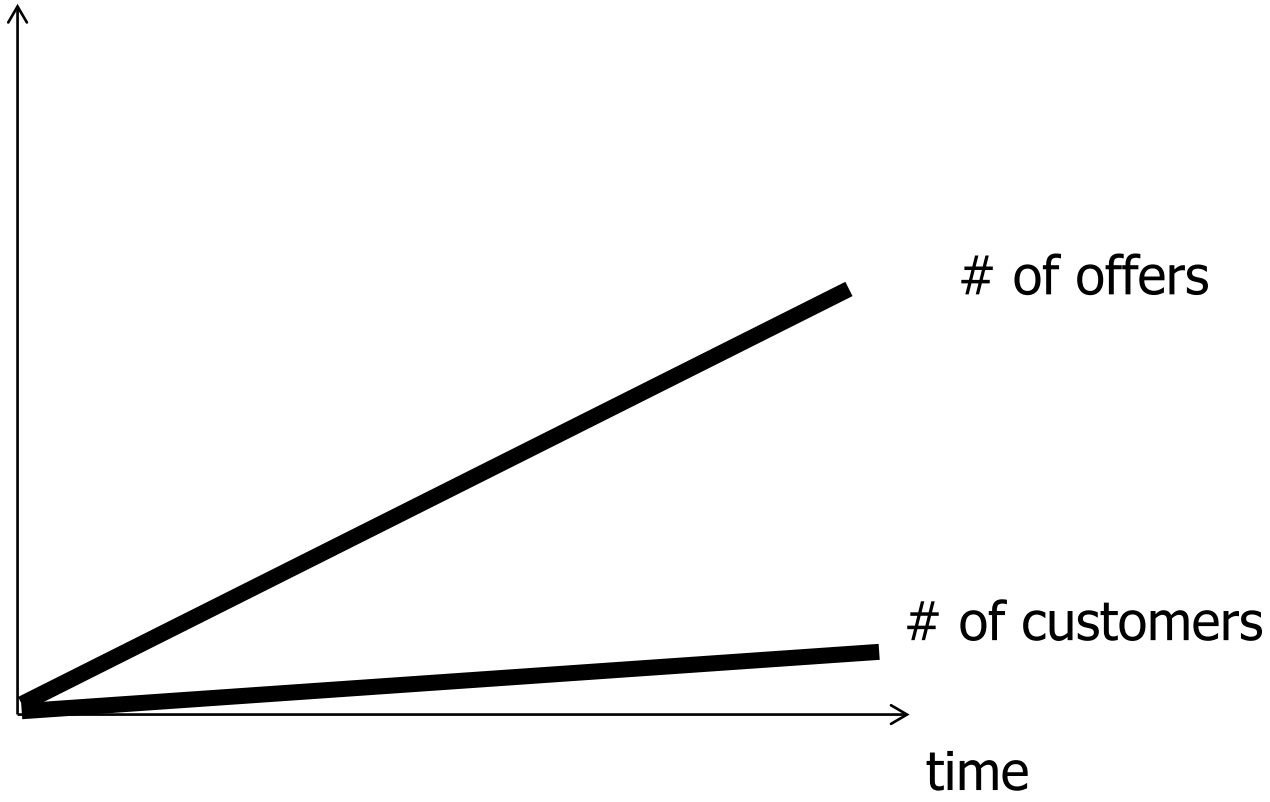
Story part 2

- Company: "I need to track my offers"
  - Customers have projects and call for bids, company makes offers
  - Many customers have many projects with many offers over time
- Naïve engineer: "No problem"
  - Reuse existing architecture
  - Load offers and customers from disk into memory
  - Repeat until "Q"
    - (S)earch and list offers
    - (E)dit offer
    - (D)elete offer
    - (I)nsert offer
    - (Q)uit
  - Write offers to disk
- Invoice: more DM

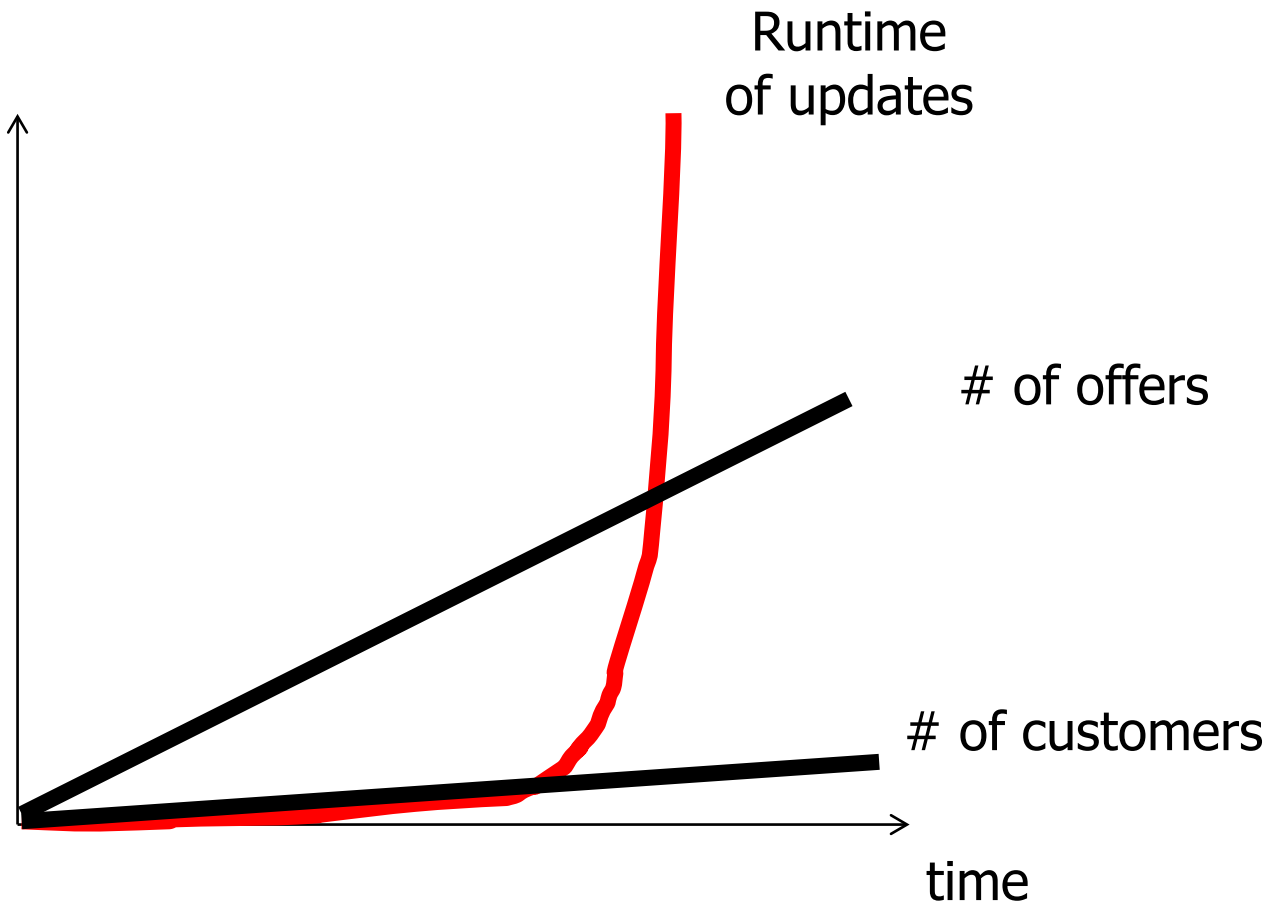
Story part 3



Story part 3



Story part 3



Part 3

- Disaster!
  - ~ 500 customers
  - ~ 40 offers per customer
  - ~ 2KB per offer
  - Gives 500*40*2.000 = 40 MEGABYTE
    - Recall: Schneider CPC646, 512 KB main memory
    - This was the size of a hard disc at that time
  - No way to load and hold all data at startup
- Wrong architecture
- Solution: Buy a RDBMS

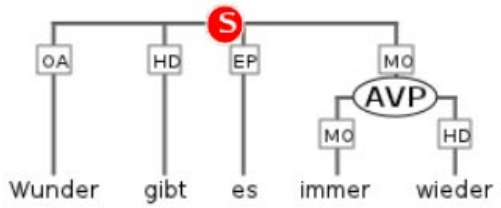
Lessons learned

- Scalability in data management is an issue
  - Scalable: Graceful runtime degradation with increasing data volumes
  - Not scalable: Works fine for small datasets, breaks down for large datasets
- Data is business-critical
  - If offers-file corrupted – company goes out of business
  - If computer crashes during edits – all changes since last start are lost
- Think before you start programming
  - Project 100% over budget (license for dBase IV)
  - Project 300% over time (6 months instead of 2)

Second Example: Linguistic Databases

(Victor Rosenfeld, 2013)

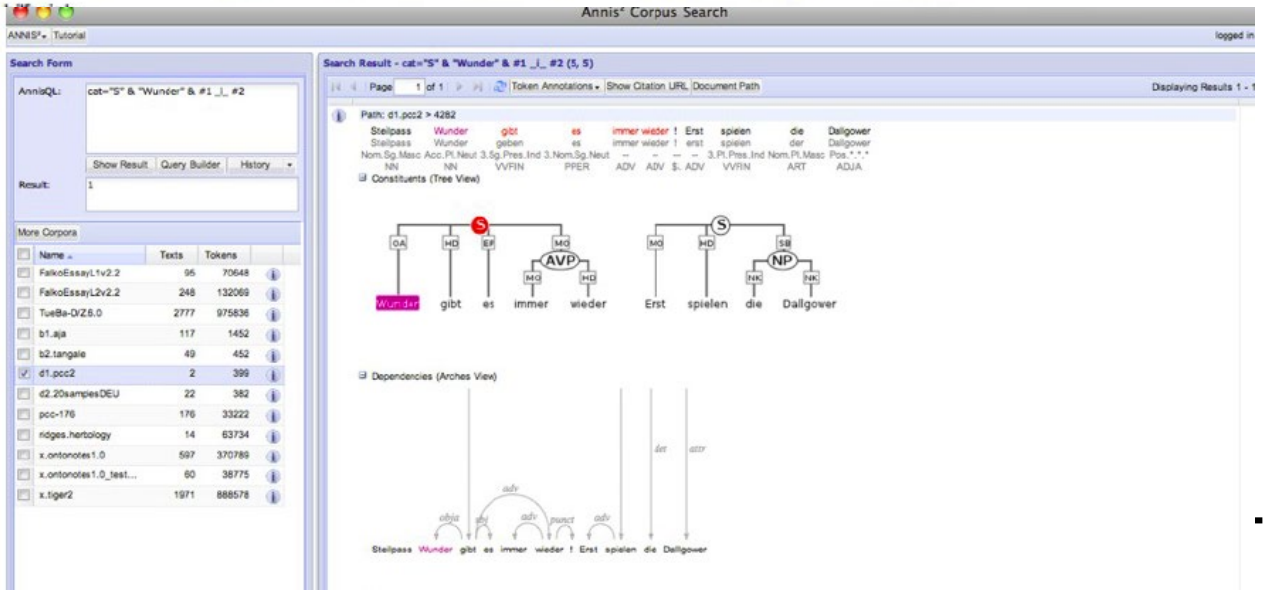
Wunder	gibt	es	immer	wieder	!
Wunder	geben	es	immer	wieder	!
Acc.Pl.Neut	3.Sg.Pres.Ind	3.Nom.Sg.Neut	--	--	--
NN	VVFIN	PPER	ADV	ADV	$.



Semantic types

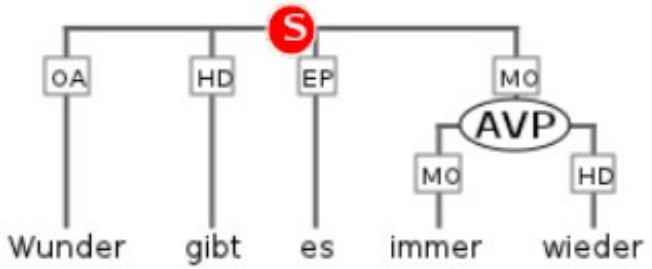
Steilpass Wunder gibt es immer wieder ! Erst spielen die Dallgower Gemeindevertreter so statisch und verzagt wie die deutsche Abwehrreihe der Fußballkicker . Und dann kommt aus der Tiefe solch ein fulminanter Steilpass , von dem man hofft , dass die Seeburger oder Groß-Glienicker Mitspieler ihn aufnehmen können . Ein Befreiungsschlag ist es allerdings nicht , weil es vorerst keine Gefahr fürs Dallgower Tor gab . Die Seeburger und einige Groß-Glienicker haben den Ball erst zurückgespielt und dann um so drängender wieder gefordert . Nun sollen sie zeigen , wie sie die Chance verwerten . Eine Diskussion , wo künftig die Trainerkabine stehen soll , wäre in der jetzigen Spielsituation verheerend . Und eine Parallele zu den deutschen Grotten-Kickern gibt es immer noch . Auch wenn die Spieler aus den verschiedenen Vereinen zusammengewürfelt sind , sie müssen sich daran gewöhnen , dass sie nun in einer Mannschaft " Döberitzer Heide " spielen . Und das heißt gemeinsam und nicht gegeneinander . Ermahnungen von der Seitenlinie , miteinander fair umzugehen und sich nicht beim kleinsten Schubser gegenseitig zu zerfleischen , sind normalerweise überflüssig . Vorerst allerdings

Cross-sentence links



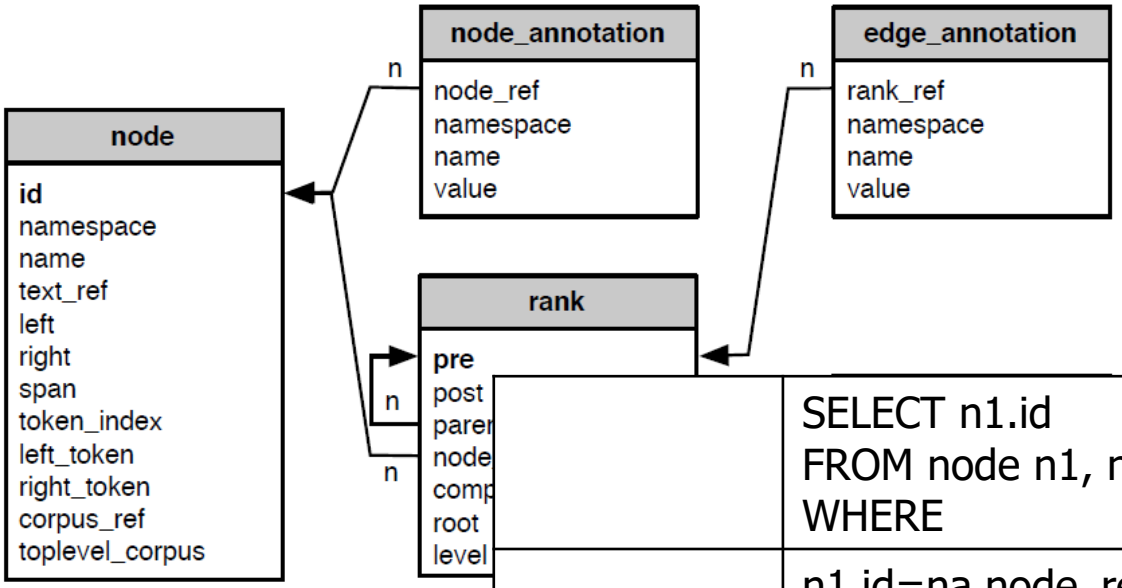
AQL Queries

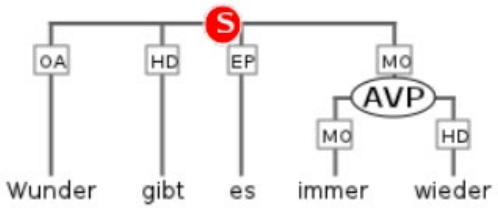
Wunder	gibt	es	immer	wieder	!
Wunder	geben	es	immer	wieder	!
Acc.Pl.Neut	3.Sg.Pres.Ind	3.Nom.Sg.Neut	--	--	--
NN	VVFIN	PPER	ADV	ADV	$.



cat=„S“ &	Find all sentences; bind to variable #1
„wunder“ &	Find all token „Wunder“; bind to variable #2
#1 _i_ #2	Join: remove #1 which do not include #2

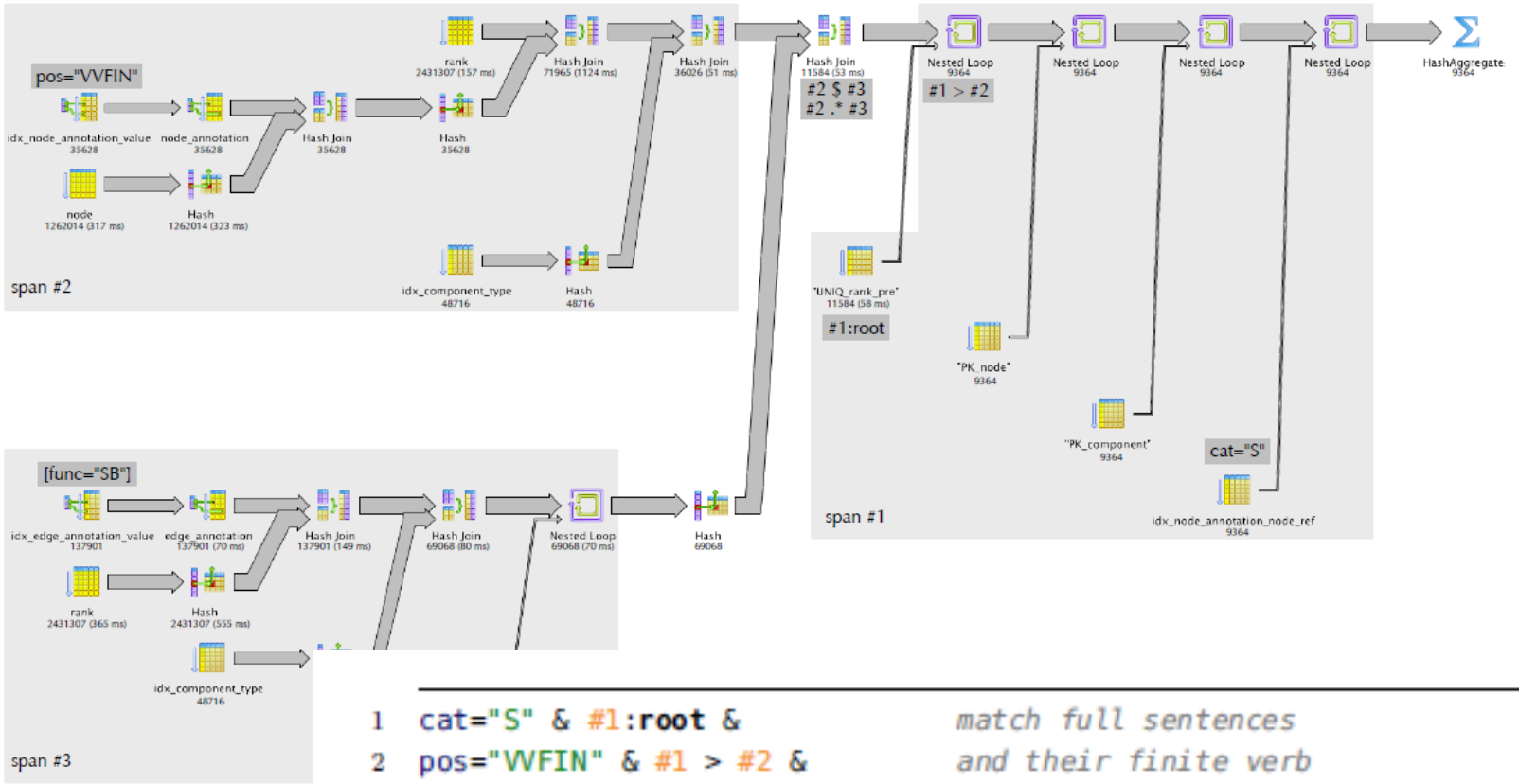
Let‘s do it right - PostGreSQL





	SELECT n1.id FROM node n1, node n2, node_annotation na WHERE
cat=„S“ &	n1.id=na.node_ref AND na.name=„cat“ AND na.value=„S“ AND
„wunder“ &	n2.span=„Wunder“ AND
#1 _i_ #2	n1.text_ref=n2.text_ref AND n2.right<=n1.right AND n2.left>=n1.left

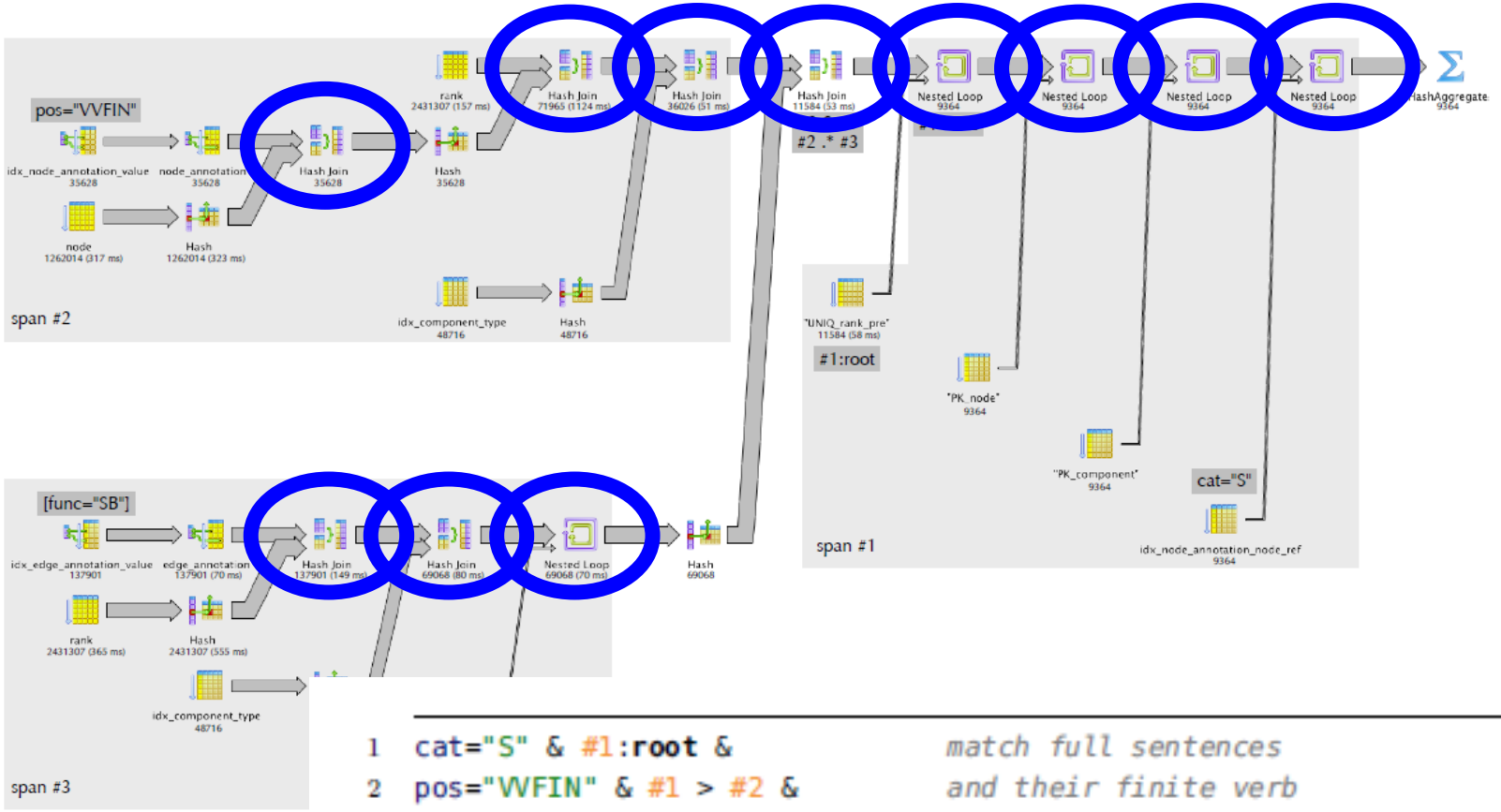
More Complicated Queries




```
1  cat="S" & #1:root &          match full sentences
2  pos="VVFIN" & #1 > #2 &      and their finite verb
3  node & #1 >[func="SB"] #3 &  and their subject
4  #2 .* #3 &                   where the verb precedes the subject
5  meta::Genre="Politik"        only consider documents of the genre Politik
```

Listing 1: Annis query matching sentences in which the subject follows the verb.

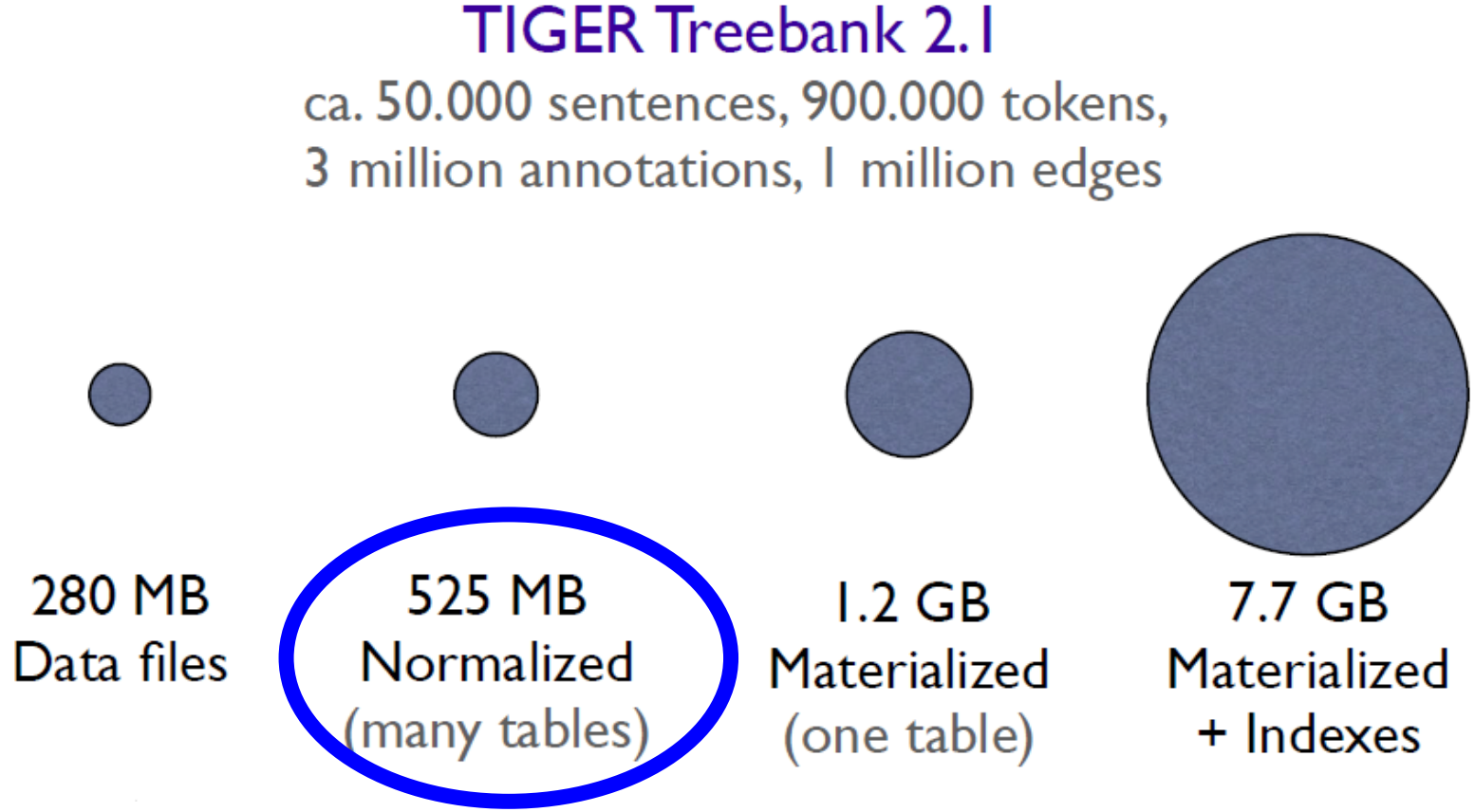
Did we do it right?




```
1  cat="S" & #1:root &          match full sentences
2  pos="VVFIN" & #1 > #2 &      and their finite verb
3  node & #1 >[func="SB"] #3 &  and their subject
4  #2 .* #3 &                   where the verb precedes the subject
5  meta::Genre="Politik"        only consider documents of the genre Politik
```

Listing 1: Annis query matching sentences in which the subject follows the verb.

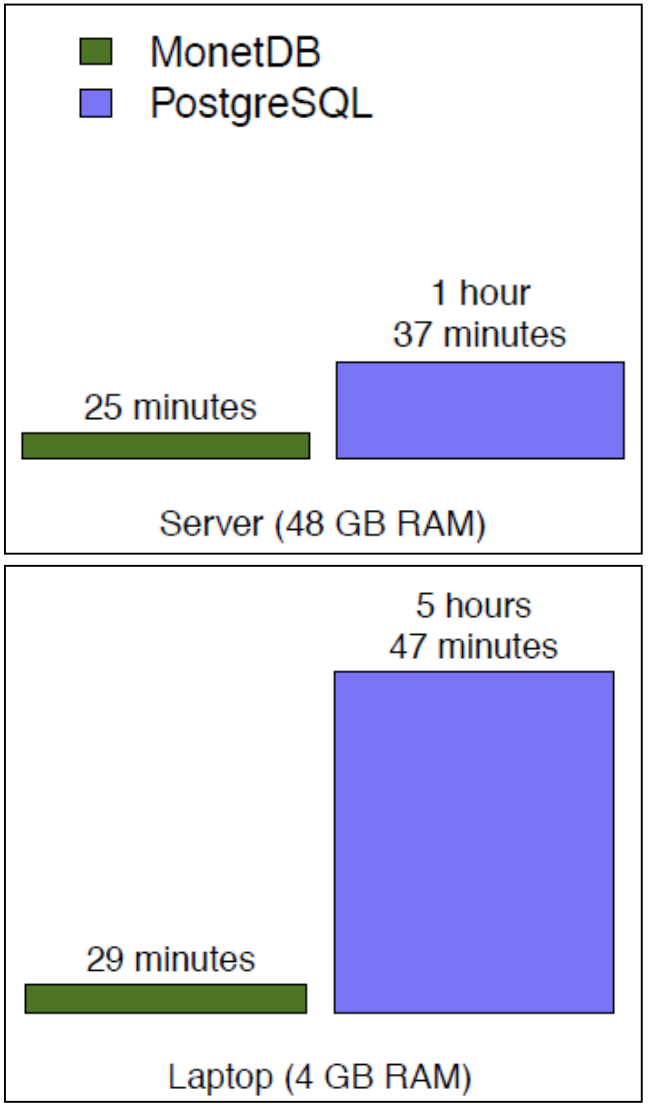
RDBMS Feature: Indexes, Materialized Views



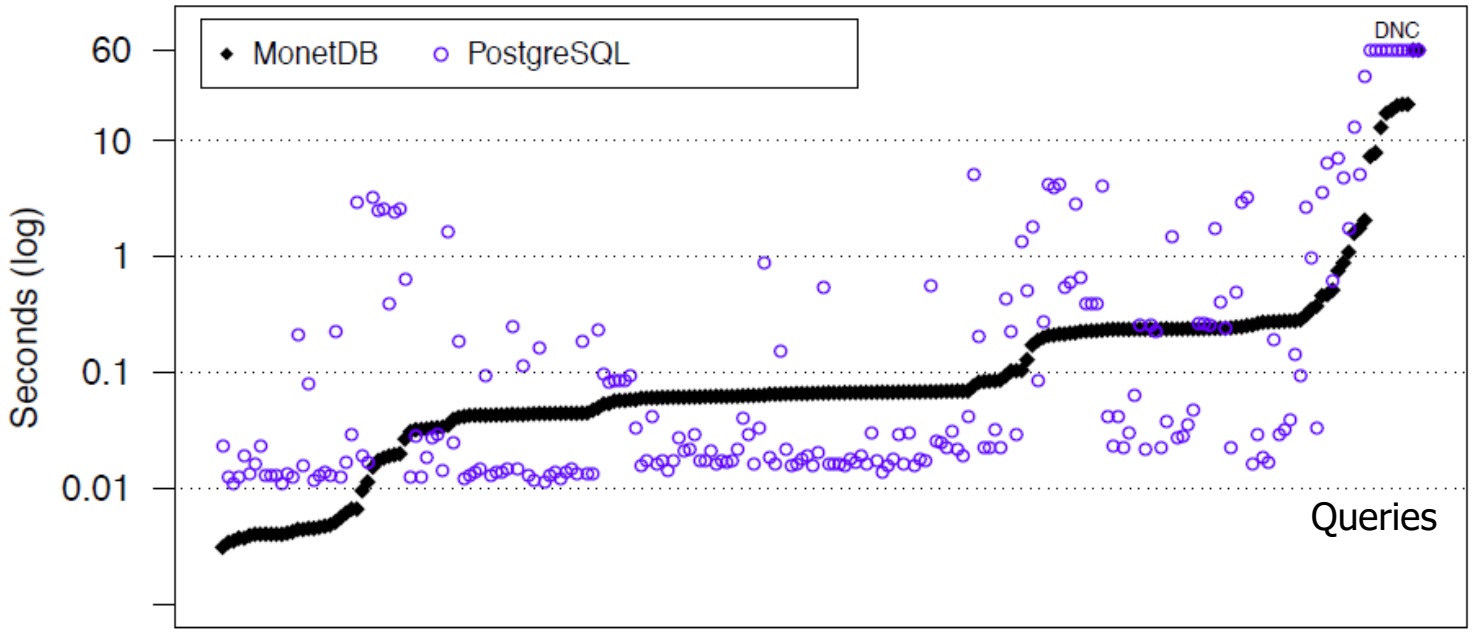
It is 2016 – we can keep this in memory!

MonetDB: A Main-Memory Column Store

- Workload: 330 real-life queries
- MonetDB is a RDBMS, but
  - All data kept in main memory
  - No indexes – all scans
  - Column store: Keep column values together, not tuples
- Advantages
  - No IO, buffering, caching, …
  - Much better cache utilization for scans (outweighs missing indexes)
  - Column compression (memory, faster scans)
- Still relational: Many joins

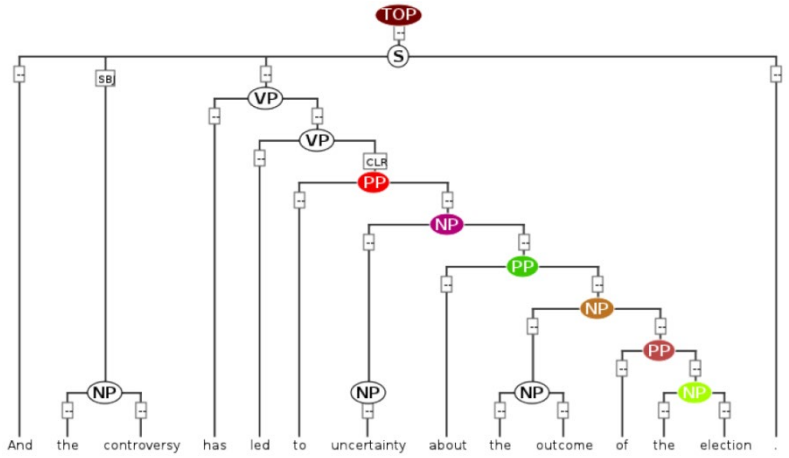


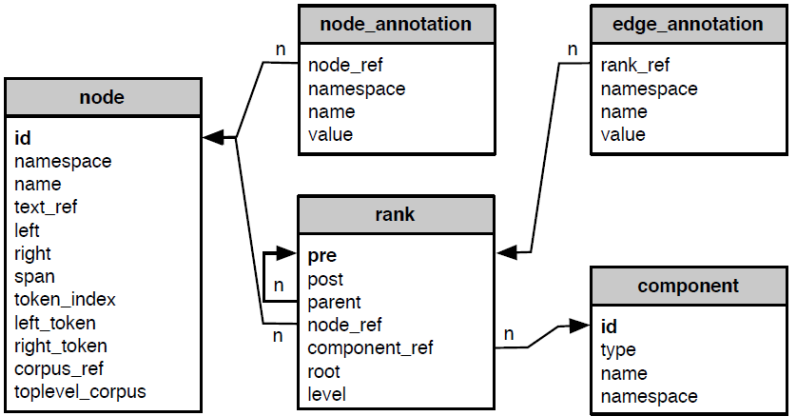
Query Optimization is Difficult to Predict



- PostGreSQL is faster for many queries despite IO
- But if it is slower, it is much slower (log scale)
- It all depends on selectivity of subqueries

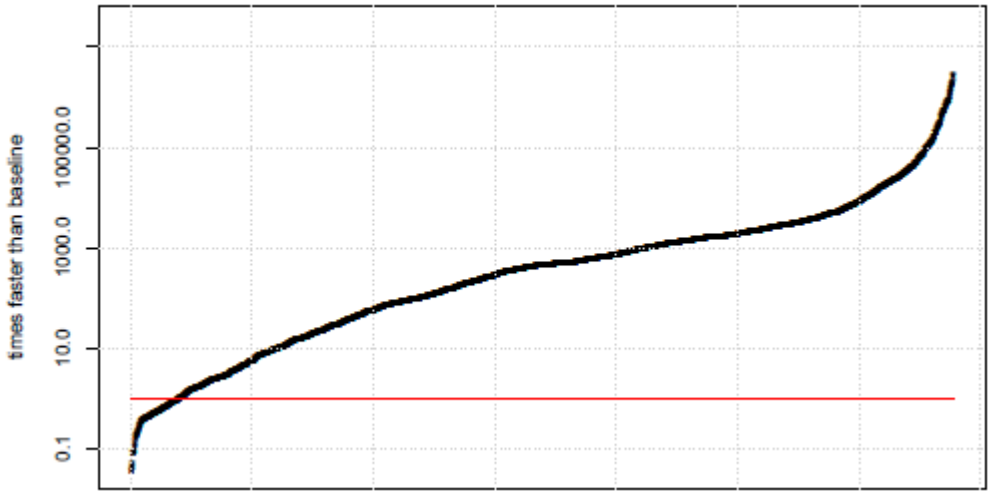
Even Better: A Graph Store





- AQL queries navigate through graphs
- Relational: One join for (almost) every edge traversal
- graphANNIS: AQL on a main-memory graph data model
  - No joins, but following pointers
    - Implemented as indexes into arrays
  - Indexes to find the right nodes quickly (to start traversals)

Thomas Krause [Krause, Leser, Lüdeling, 2017]



- Workload: ~3300 real-life queries against ~20 corpora
- graphANNIS versus PostGreSQL
  - ~40 times faster on the entire workload
  - Faster for 97% of all queries
  - Not much slower for the remaining 3%

Databases are Infrastructure

Today's Database Systems

- RDBMS are essential parts of enterprise infrastructures
  - More important than OS
  - Long-running, expensive, essential investment
  - Holds the most important business assets: Data, information
- Database administrator is a well-paid profession
  - Developers write SQL & business logic
  - Admins make SQL run fast
  - Many programmers, fewer DB developers, very few DB admins
  - A skills much demanded in industry
- RDMS became an often "invisible" piece of software
  - "So nützlich wie fließendes Wasser" (G. Weikum, MPI Saarbrücken)

Main Features

- Data needs to be stored
  - Disk access (or cache utilization) is the main bottleneck
  - Hence: Minimize access time -> minimize disk access

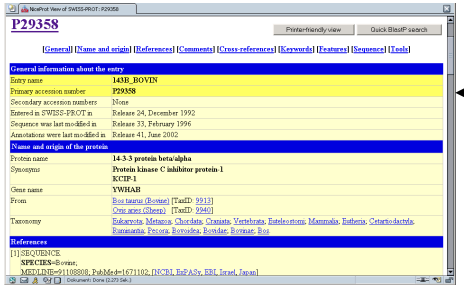
  DBS2
- Data is manipulated by many clients
  - Concurrent access quickly screws up data
  - Hence: Synchronize access

  DBS2
- Data used by diff apps with diff requirements
  - Avoid application specific "optimal" data structures
  - Use appl-independent languages (SQL) and models (ER)

  DBS1
- Systems crash
  - Crashes cannot be avoided
  - Protect consistency by transactions, enforce constraint, …

  DBS1/2

Classical Three-Tier Architecture



Servlets/
EJB

Presentation

Upwards: OO Interface
Application Server
Downwards: SQL

Application logic
"Business processes"

DBMS

storage, query optimization
and execution
recovery

Database 1

Database 2

DBS2: Implementation of Database Systems

- Lecture 4 SWS
  - Tuesday, 11 – 13 , 3.101
  - Thursday, 11 – 13 , 3.101

- Contact

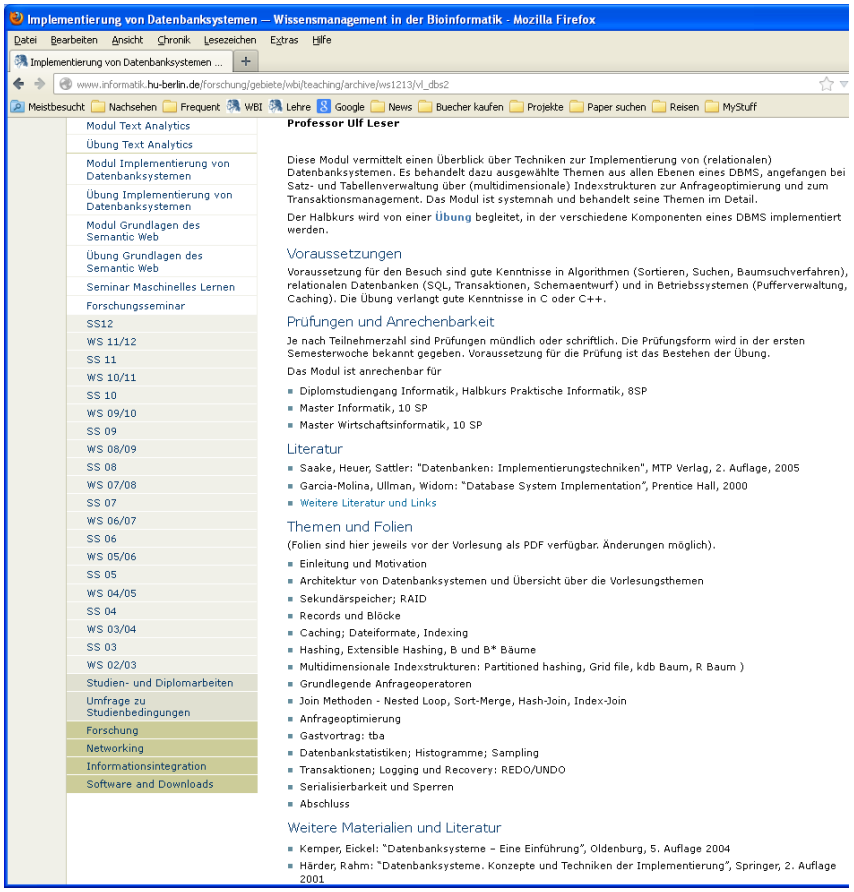
Ulf Leser
Room: IV.401
Tel: (030) 2093 – 3902
Mail: leser (at) informatik.hu-berlin.de

Exercises and Examination

- Exercises run by Arik Ermshaus
  - Presence & commitment are necessary
  - You will build groups
  - Implementation of file-/ buffer-/ index manager
  - Tuesday / Thursday, 9-11, 3.101
  - Starts 24.10 / 28.10 (next week)
- Examination
  - Oral or written?
  - Dates will be set mid-January
  - Admission: Passing the exercises

Slides

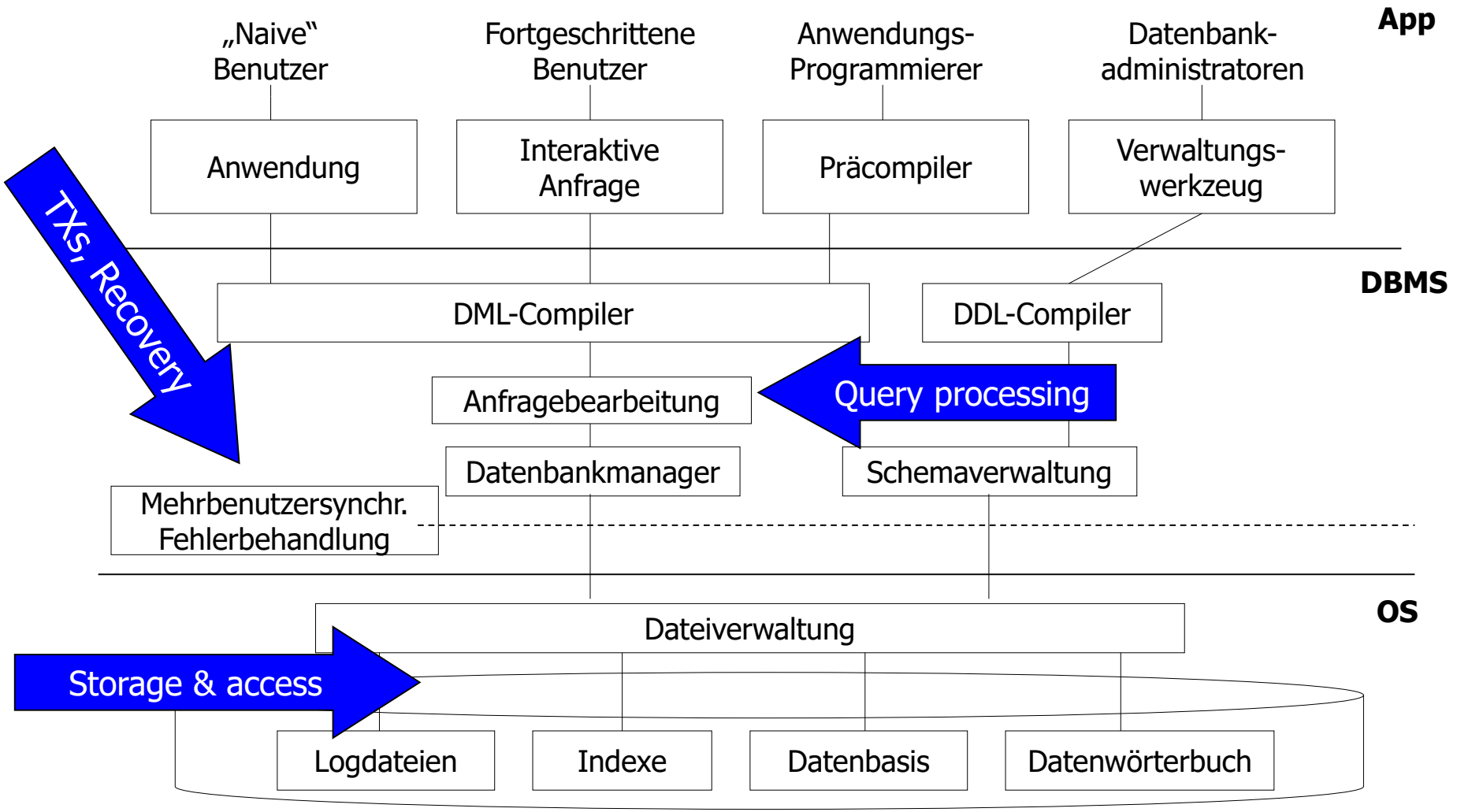
- Slides are available shortly after the lecture
- Please send me
  any errors
- Slides are
  - not a script
  - no substitution for listening to the lecture
  - not a substitute for books



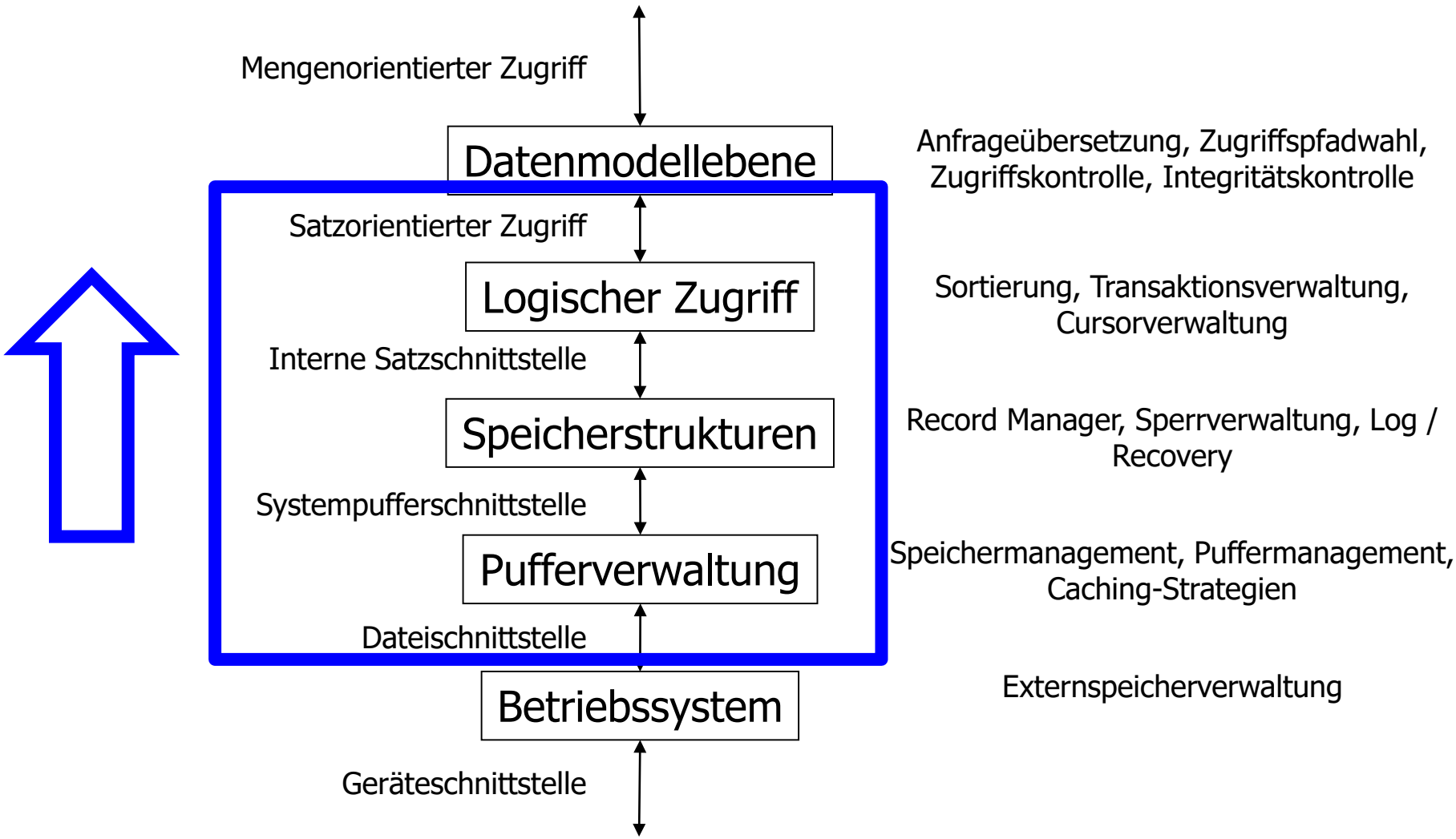
Literature

- Primary
  - Saake, Heuer, Sattler "Datenbanken: Implementierungs-techniken", mitp Verlag, 2005 (2. Auflage)
  - Garcia-Molina, Ullman, Widom: "Database System Implementation", Prentice Hall, 2000
- Other
  - Kemper, Eickel: "Datenbanksysteme – Eine Einführung", Oldenburg, 5. Auflage 2004
  - Härder, Rahm: "Datenbanksysteme. Konzepte und Techniken der Implementierung", Springer, 2. Auflage 2001
  - R. Elmasri und S.B. Navathe: Fundamentals of Database Systems, Benjamin Cummings
    - Deutsche Übersetzung: „Grundlagen von Datenbanksystemen", Pearson, 2002

Überblick


App
„Naive" Benutzer
Fortgeschrittene Benutzer
Anwendungs-Programmierer
Datenbank-administratoren
Anwendung
Interaktive Anfrage
Präcompiler
Verwaltungs-werkzeug
DBMS
TXs, Recovery
DML-Compiler
DDL-Compiler
Query processing
Anfragebearbeitung
Datenbankmanager
Schemaverwaltung
Mehrbenutzersynchr.
Fehlerbehandlung
OS
Dateiverwaltung
Storage & access
Logdateien
Indexe
Datenbasis
Datenwörterbuch

Contents


Mengenorientierter Zugriff
Datenmodellebene
Anfrageübersetzung, Zugriffspfadwahl, Zugriffskontrolle, Integritätskontrolle
Satzorientierter Zugriff
Logischer Zugriff
Sortierung, Transaktionsverwaltung, Cursorverwaltung
Interne Satzschnittstelle
Speicherstrukturen
Record Manager, Sperrverwaltung, Log / Recovery
Systempufferschnittstelle
Pufferverwaltung
Speichermanagement, Puffermanagement, Caching-Strategien
Dateischnittstelle
Betriebssystem
Externspeicherverwaltung
Geräteschnittstelle

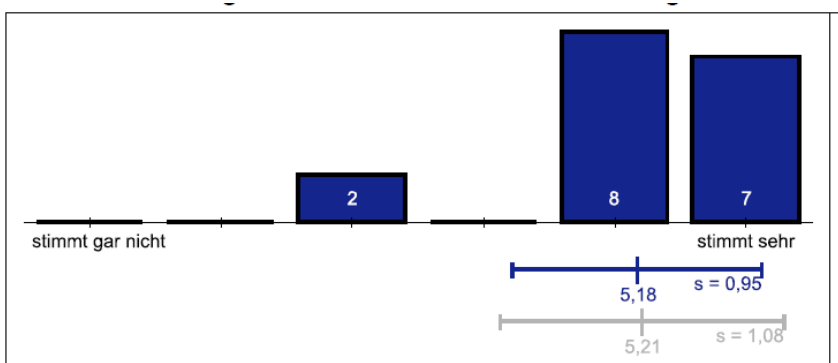
Contents

- Introduction
- Overview and architecture
- Storage and access methods
  - B*-Trees, Extensible hashing, index-sequential files …
  - Multidimensional indexing: Grid-files, kd-Trees, R-Trees …
- Query processing and optimization
  - Physical relational operators
  - Cost-based optimization
- Recovery
- Transactions and concurrency control

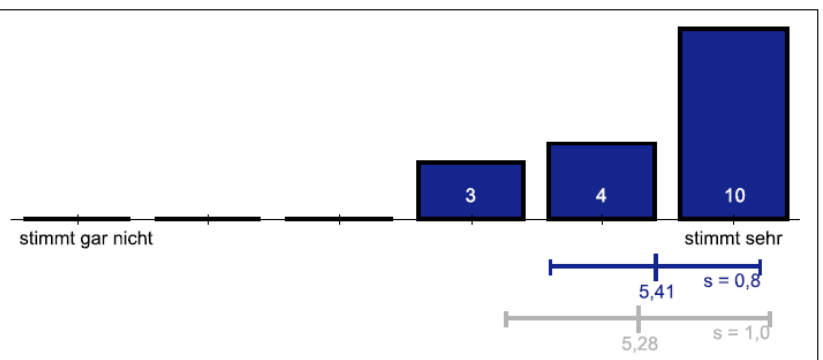
Issues

- Vorlesung muss entfallen am
  - 31.10.23
  - 2.11.23
  - 7.11.23
  - 9.11.23
  - 21.11.23
- Vorlesung wird vertreten am
  - 26.10.23
- Vorlesung aus der Reihe
  - 19.10.23: Zwei Vorlesungsblöcke, 9-11 und 11-13 (3.101)

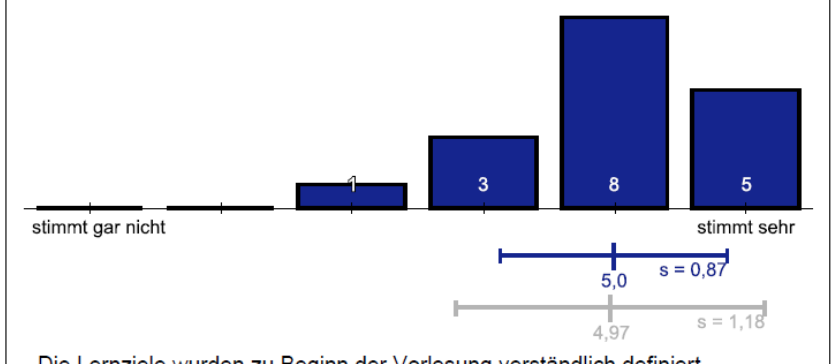
Feedback 2019/2020



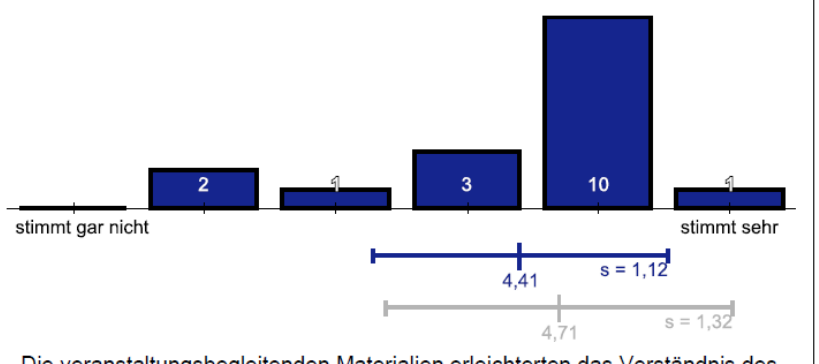
Konzeption und Gliederung der Vorlesung waren klar.



Die Vorlesung hat mir viele neue inhaltliche Erkenntnisse gebracht.

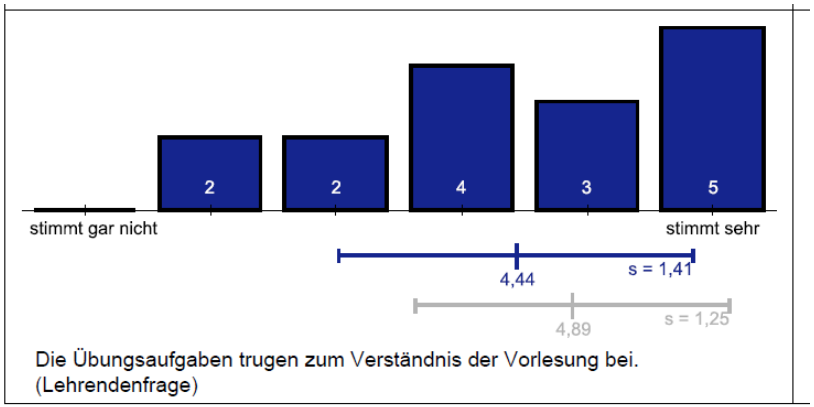


Die Lernziele wurden zu Beginn der Vorlesung verständlich definiert.



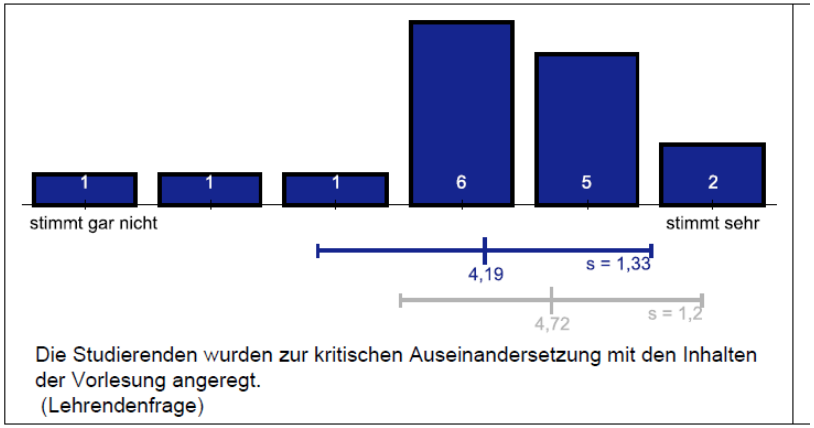
Die veranstaltungsbegleitenden Materialien erleichterten das Verständnis des Inhalts/Stoffes.

Feedback 2019/2020

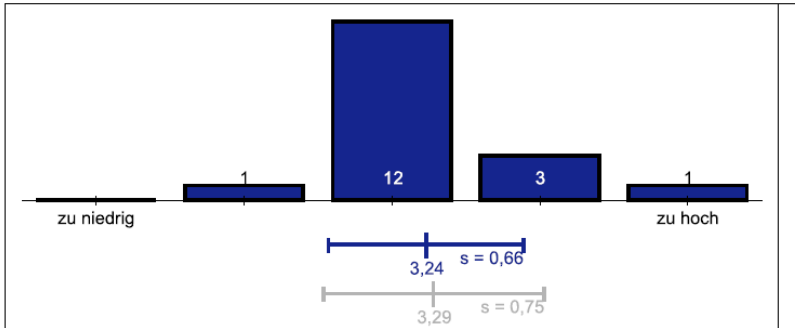


Die Übungsaufgaben trugen zum Verständnis der Vorlesung bei.
(Lehrendenfrage)

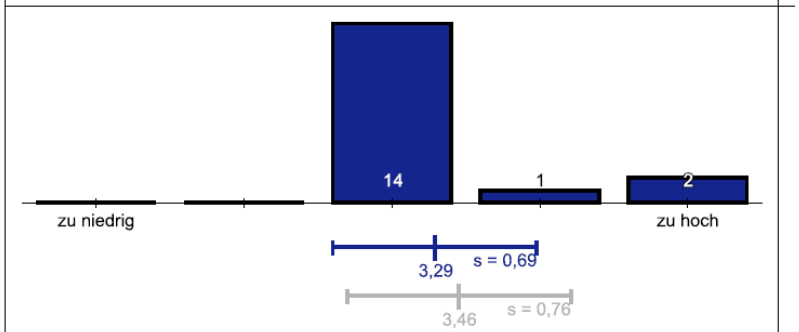
Interaktion - Vorlesung



Die Studierenden wurden zur kritischen Auseinandersetzung mit den Inhalten der Vorlesung angeregt.
(Lehrendenfrage)



Das Tempo der Veranstaltungen empfand ich als



Den Arbeitsaufwand durch die Vorlesung empfand ich als

Positiv

- Beispiele; saubere (nicht-überladene) Folien; entspannte Athmosphäre
- - die Beispiele
  - die Übungsaufgaben
  - Übungen gut vorgestellt
  - die Beispiele an der Tafel halfen dem Verständnis
- Die ungezwungene, offene Art des Lehrenden
  Die Anekdoten zwischendurch, weil sie die Möglichkeit geben, eine Konzentrationspause zu machen und danach wieder frisch dabei zu sein
- Guten und tiefen Einblick in die Funktionsweise von relationalen Datenbanken.
  Dass der Dozent sich immer Notizen zu seinen Folien macht und diese verbessert, bevor er sie bereitstellt.

- Guter roter Faden, viele interessante Techniken, gute Beispiele
- Interessante Inhalte
  Angenehme Atmosphäre
- - Professor wirkt motiviert und enthusiastisch
  - man darf Ihn jederzeit unterbrechen und Fragen stellen
  - die Inhalte der Vorlesung wurden manchmal eingebettet in Geschichten aus der Praxis
- Sehr strukturierte Vorlesung, deren Information spannend durch Herrn Leser vermittelt wurden. Dabei wurde eine angenehme lern Atmosphäre geschaffen.
- Viele Beispiele
  Übungsaufgaben waren sehr praktisch (und teilweise auch nervig) aber man hat vorallem den b+baum damit komplett verstanden und sich mit dem Blockmanagement ebenfalls viel auseinander gesetzt
  Die kurzen Ausflüge in dir Unipolitik waren auch Auflockerung, wenn auch manchmal zu lange
- - Wenige Abgaben in der Übung, die dafür einen wichtigen Inhalt vermitteln und genug Zeit zur Auseinandersetzung mit dem Thema und dem Bearbeiten der Aufgabe lassen
  - Präsentation eigener Forschungsergebnisse mit Kritik und Einordnung

Verbesserungespotential

- Den Vorlesungsinhalt evtl. etwas modernisieren. Wie lösen heutige, moderne DBMS gewisse Probleme.
- Die Folien könnten besser gestaltet sein, aktuell sind sie nicht besonders hilfreich bei der Nachbereitung
- Etwas langsamer sprechen und gerade bei konkreten Beispielen wäre es super, sie etwas langsamer durchzunehmen bzw nicht zu überspringen
- Folien etwas aktualisieren, manche Grafiken sind (v.a. im Selbststudium später) unübersichtlich und schwer zu verstehen.
- Gesamtstruktur der Vorlesung nicht immer klar
- - mehr/genauere Testfälle für die Abgaben anbieten

- bessere Tafelstifte besorgen
- - neue Marker für die Tafeln kaufen
  - die Vorlesung ist für Wirtschaftsinformatiker nur bedingt empfehlenswert, da Sie sehr anspruchsvoll ist. Jemand, der keine große Vorbildung im Bereich der Informatik hat, hat keinen guten Zugang zu den Inhalten. ("Wer von Ihnen hat bei mir AlgoDat gehört?")
- Nur Werbung für die Data Warehousing Vorlesung machen, wenn diese auch in den kommenden Semestern angeboten wird; sonst kann man sie ja gar nicht belegen und ärgert sich nur, wenn man sie hören wollte
- Vielleicht auch etwas zu Graph-Datenbanken, NoSQL sagen? Zumindestens abgrenzen

Datenbanken@Informatik

- A predefined focus area in our Master

- Datenbanken 1: Grundlagen (BA)
- Information Retrieval (BA)
- Datenbanken 2: Implementierung (Ms)
- Data Warehousing und Data Mining (Ms)
- Informationsintegration (Ms, inkl. verteilter Anfrageopt.)
- Datenbanktheorie
- Distributed Query Processing
- Process Mining
- …