



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
- Prof Freytag / Prof Leser / Prof. Weidlich
- What you should (must) know to follow this course
- What do you study?

Some Motivation

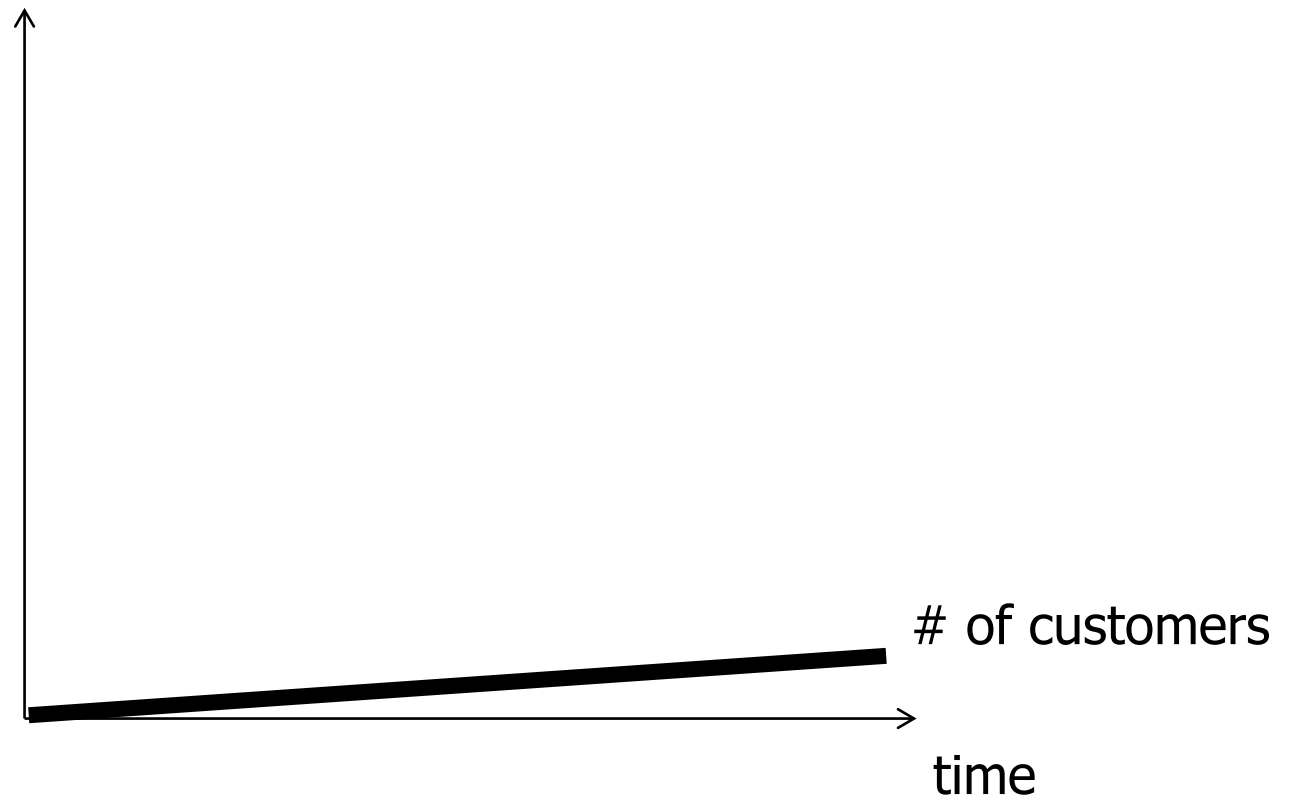
A classical first contact with database

- 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

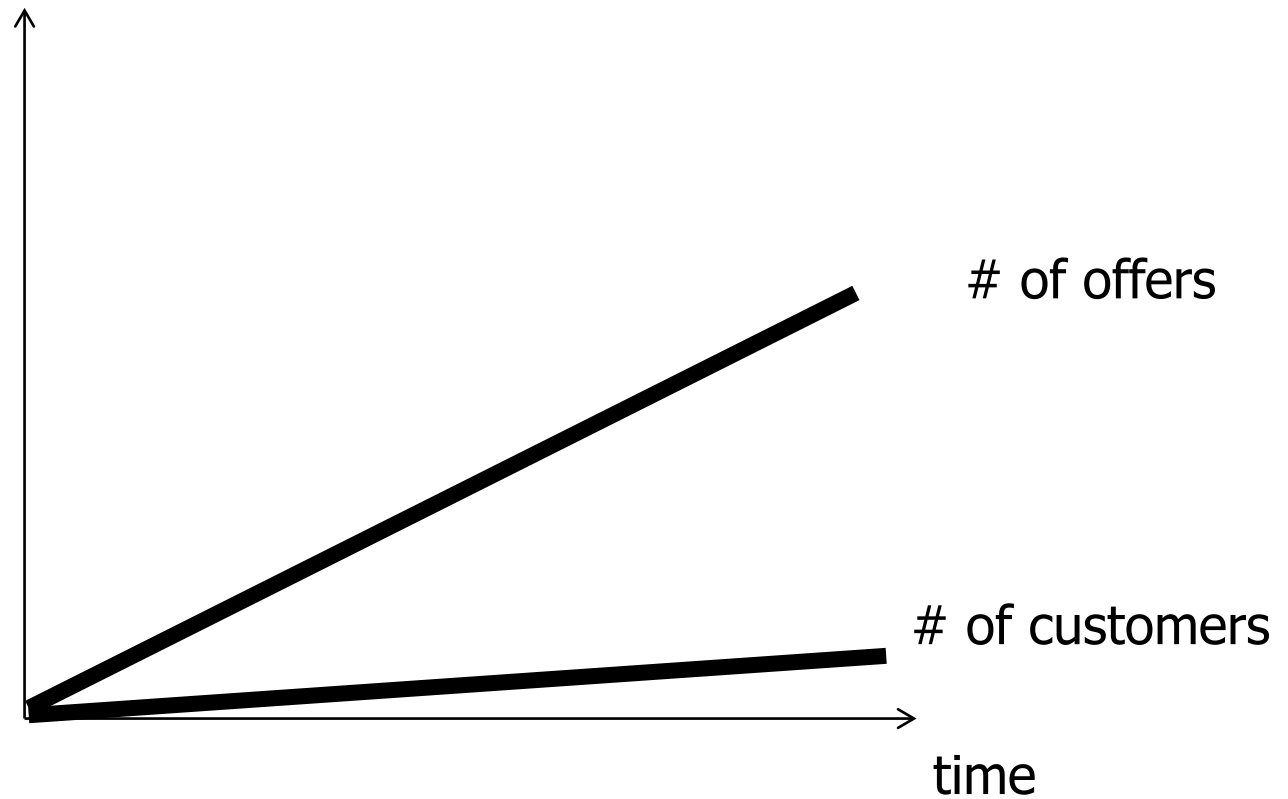
Story part 2

- 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 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

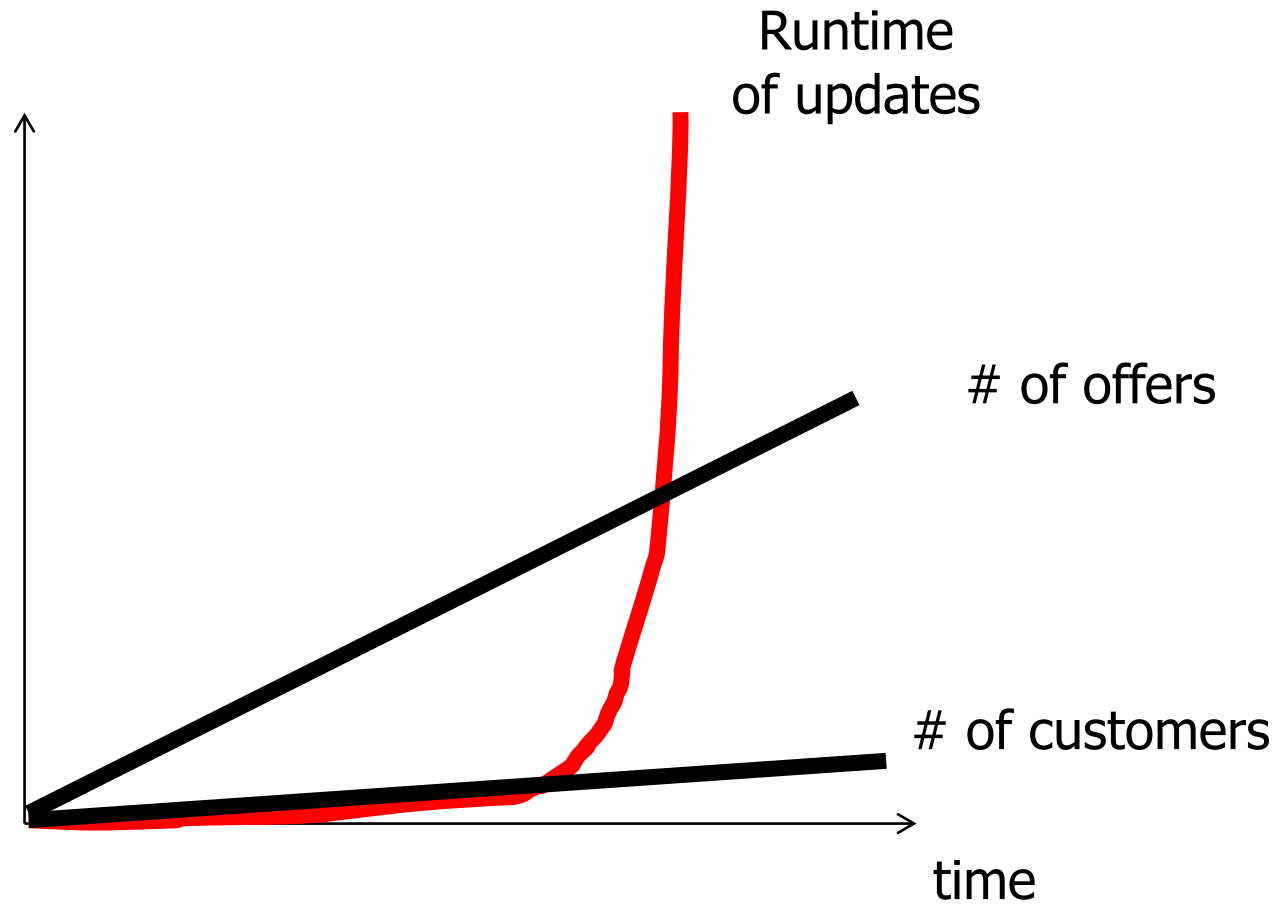
Story part 3



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](#)

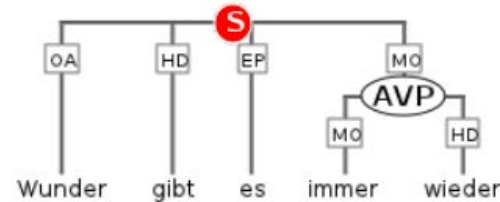
Lessons learned

- **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
 - Which will affect engineer too!
- **Think before** you start programming
 - Project 100% over budget (database license)
 - 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	VFIN	PPER	ADV ADV \$.



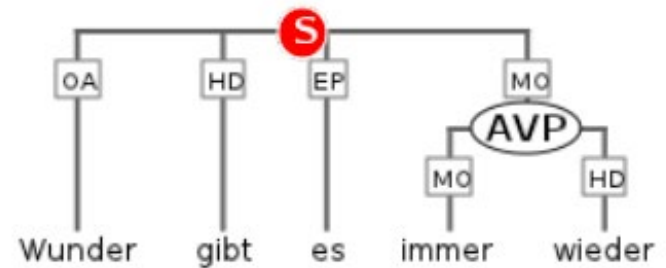
Semantic types

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 **Spitzenlinie**, miteinander fair umzugehen und sich nicht beim kleinsten Schubser gegenseitig zu zerfleischen, sind normalerweise überflüssig. Vorerst allerdings

Cross-sentence links

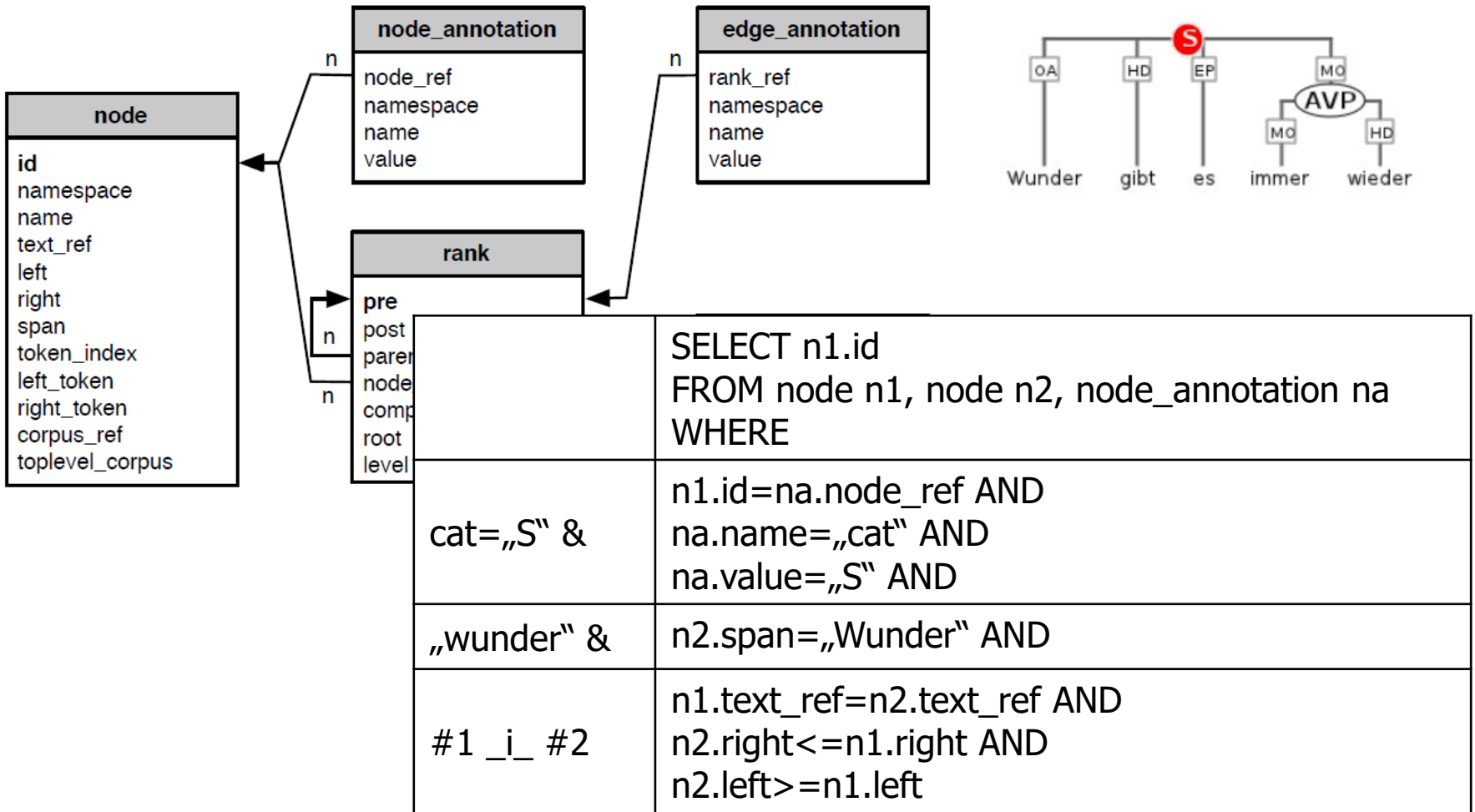
AQL Queries

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

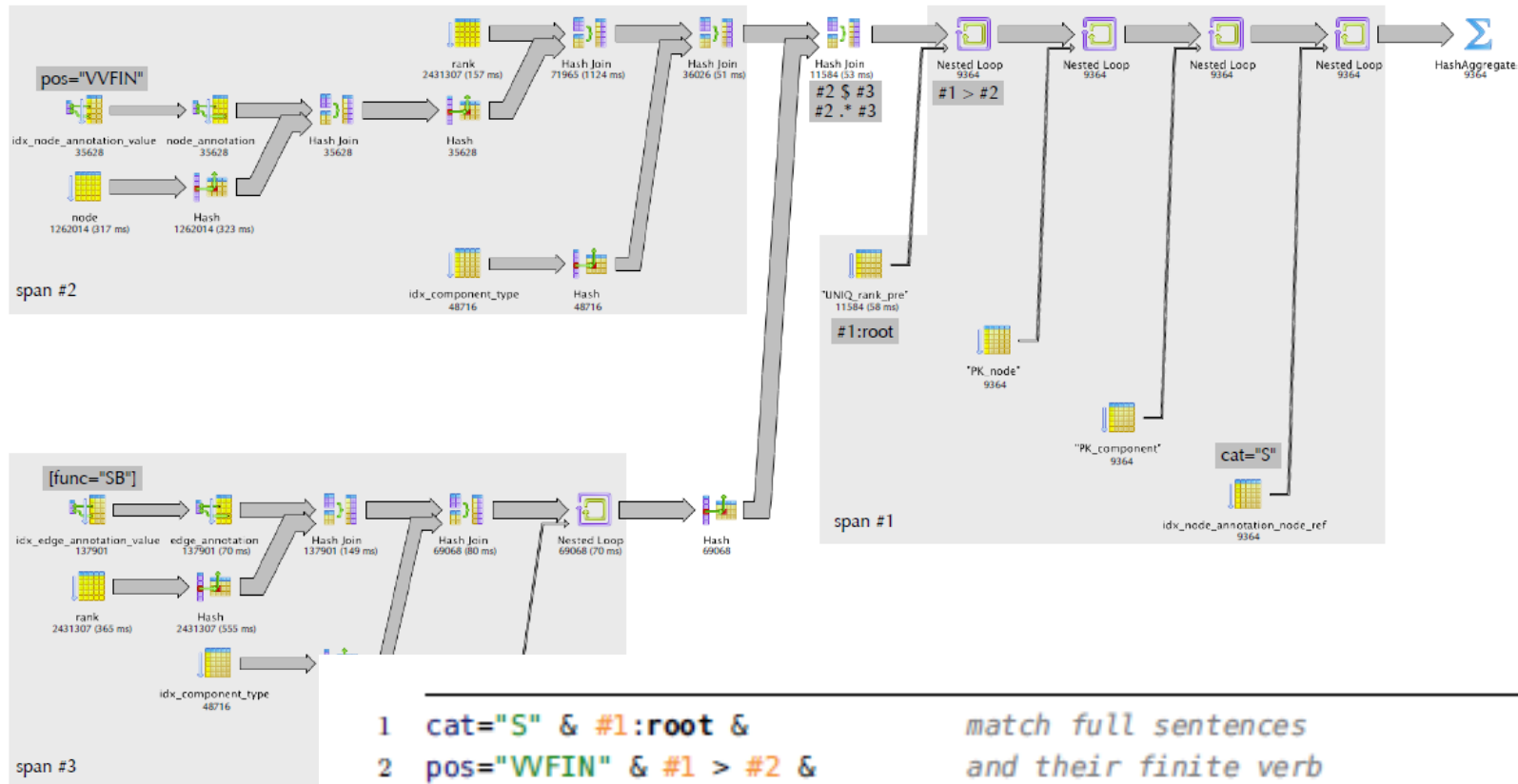


cat=„S“ &	Find all sentences; bind to variable #1
„wunder“ &	Find all token „Wunder“; bind to variable #2
#1 _i_ #2	Join: remove #1 which do not include #2

Let's do it right - PostgreSQL



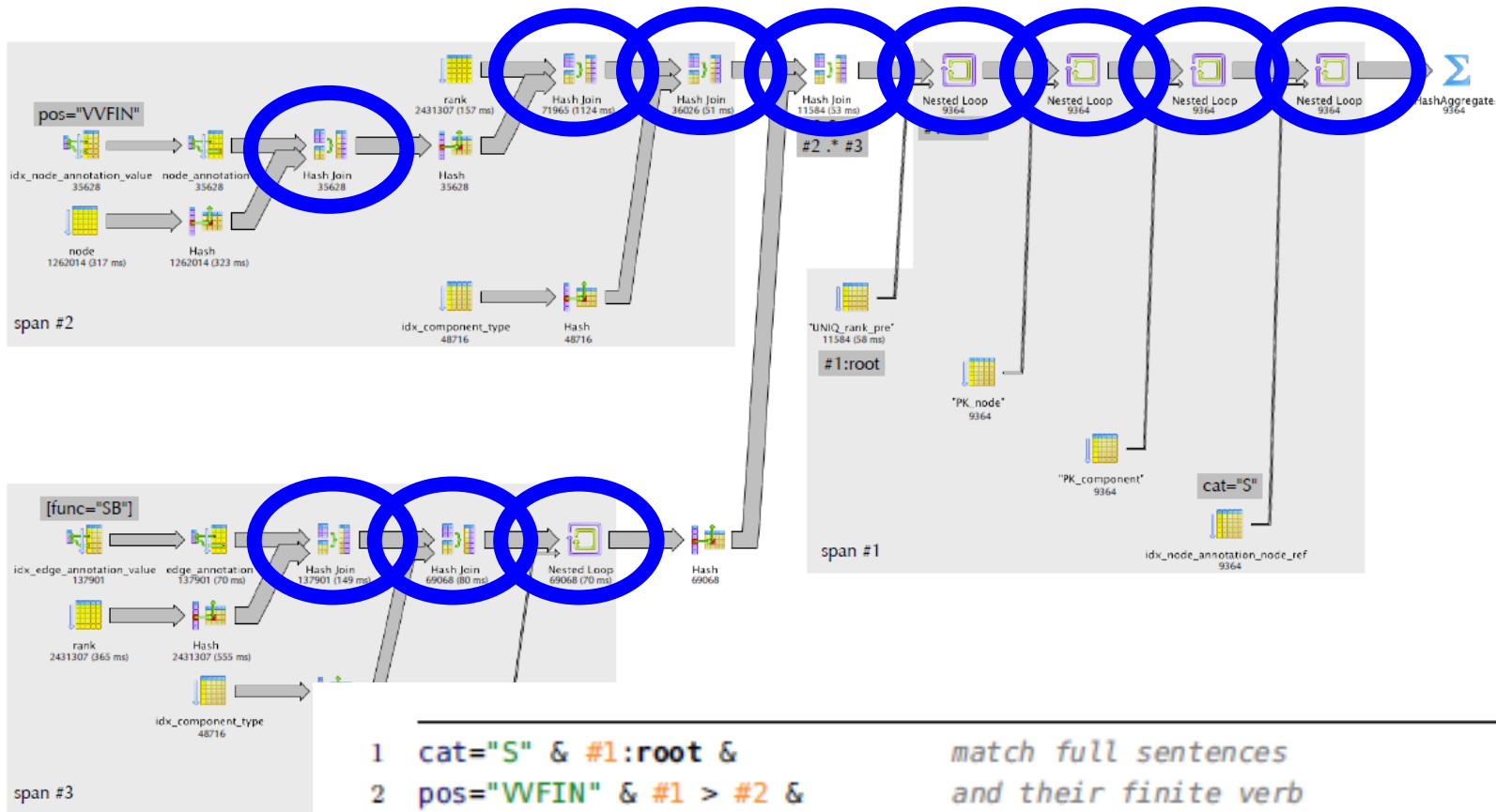
More Complicated Queries



- | | |
|--|---|
| <ol style="list-style-type: none"> 1 cat="S" & #1:root & 2 pos="VFIN" & #1 > #2 & 3 node & #1 >[func="SB"] #3 & 4 #2 .* #3 & 5 meta::Genre="Politik" | <p><i>match full sentences
and their finite verb
and their subject
where the verb precedes the subject
only consider documents of the genre Politik</i></p> |
|--|---|

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

Did we do it right?



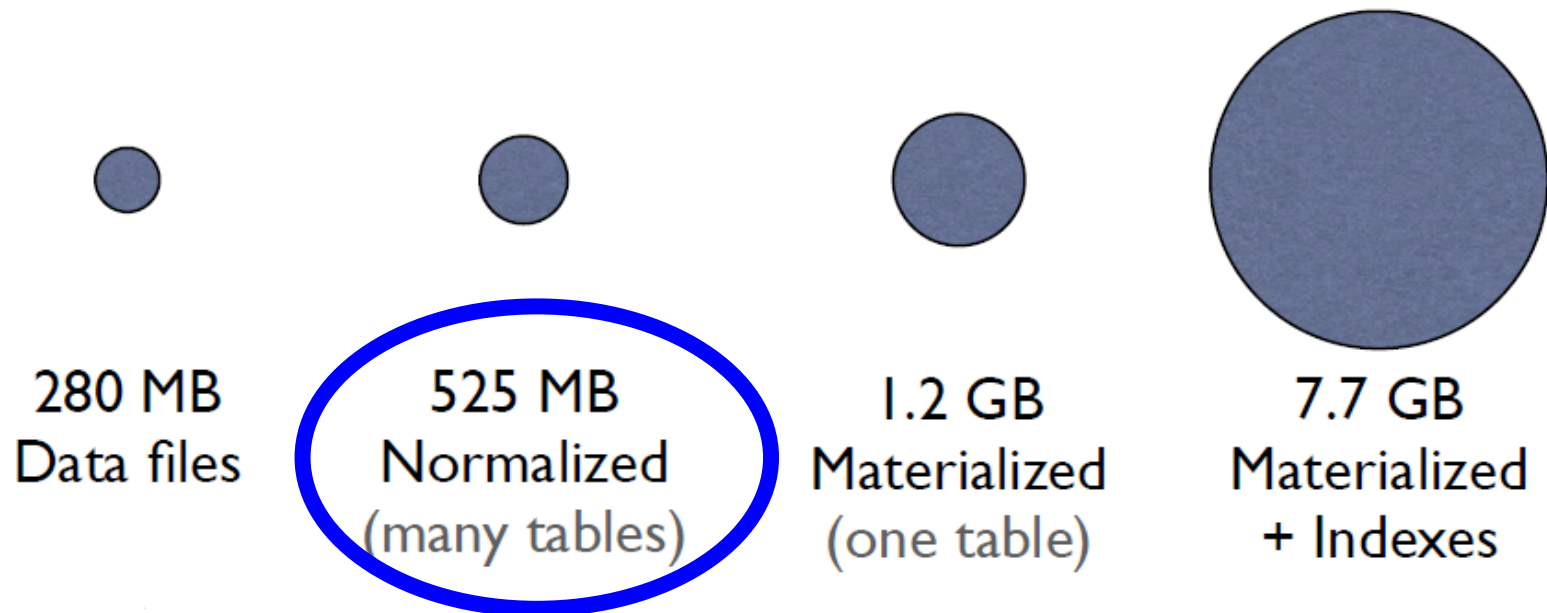
-
- | | |
|--|---|
| <ol style="list-style-type: none"> 1 <code>cat="S" & #1:root &</code> 2 <code>pos="WFIN" & #1 > #2 &</code> 3 <code>node & #1 >[func="SB"] #3 &</code> 4 <code>#2 .* #3 &</code> 5 <code>meta::Genre="Politik"</code> | <p><i>match full sentences
and their finite verb
and their subject
where the verb precedes the subject
only consider documents of the genre Politik</i></p> |
|--|---|
-

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

RDBMS Feature: Indexes, Materialized Views

TIGER Treebank 2.1

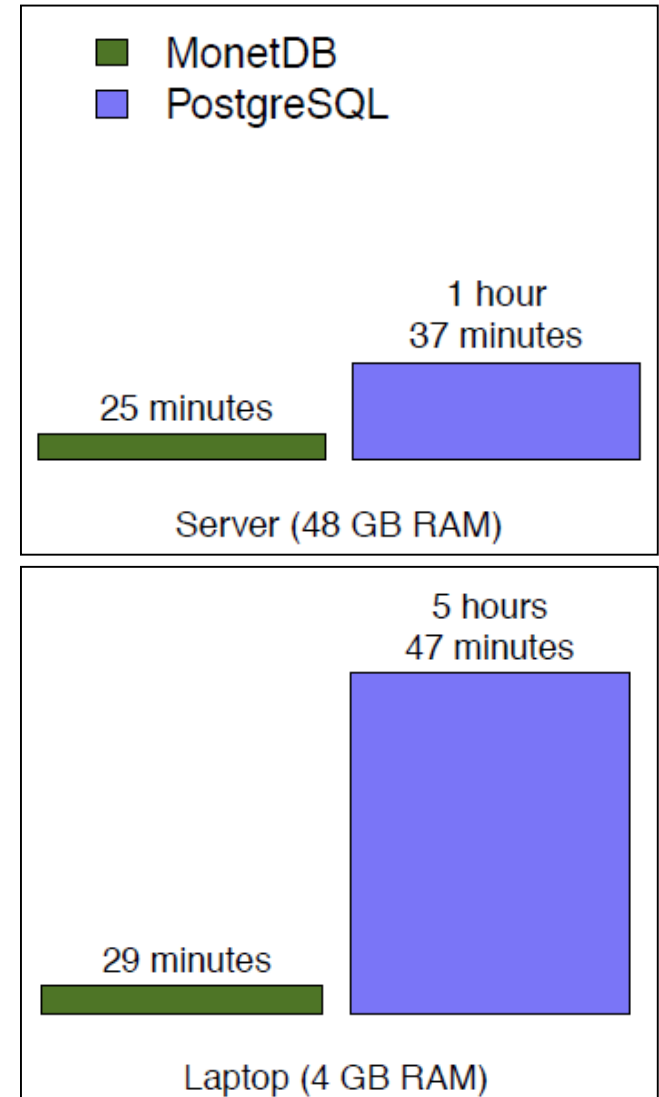
ca. 50.000 sentences, 900.000 tokens,
3 million annotations, 1 million edges



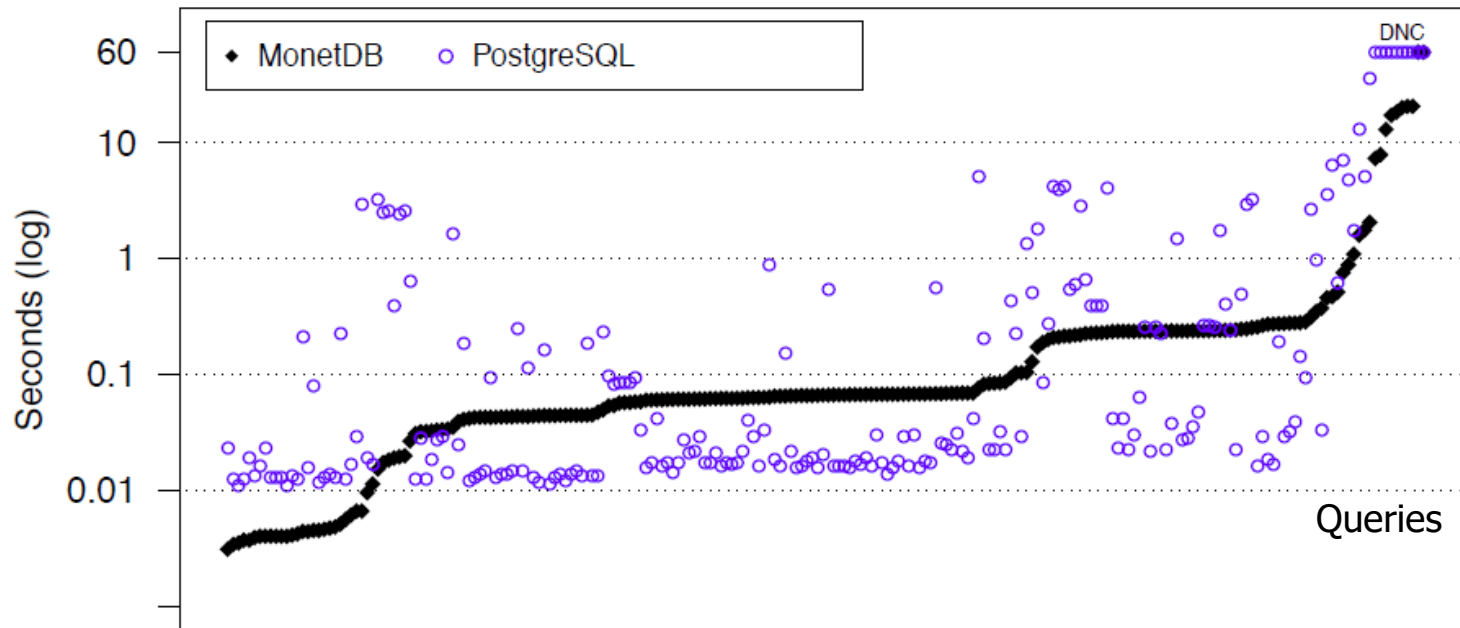
It is 2016 – we can keep this in memory!

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**

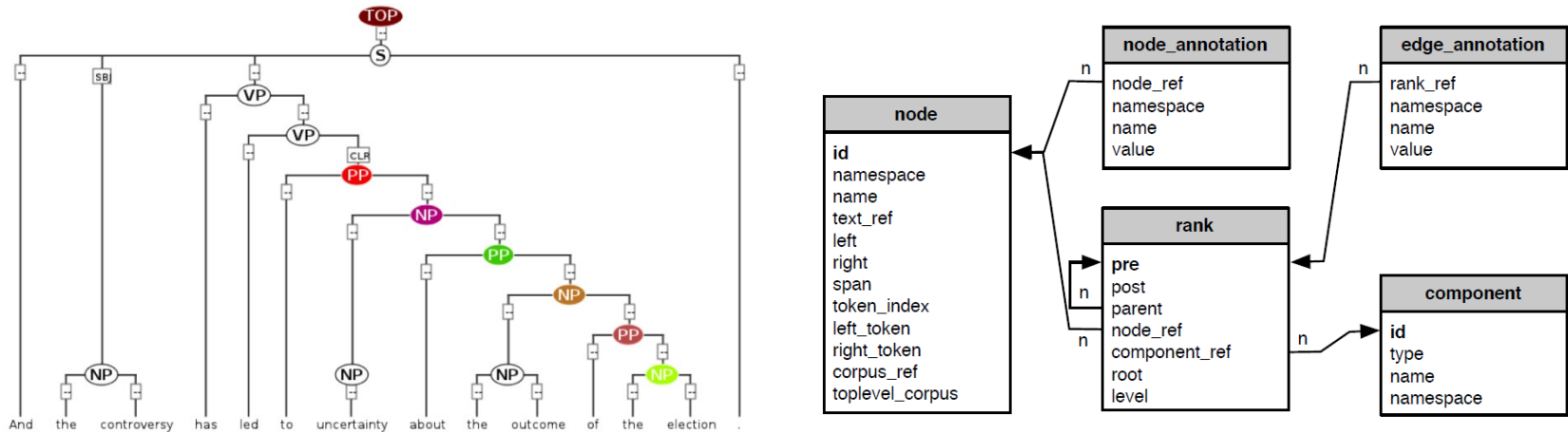


Query Optimization is Difficult to Predict



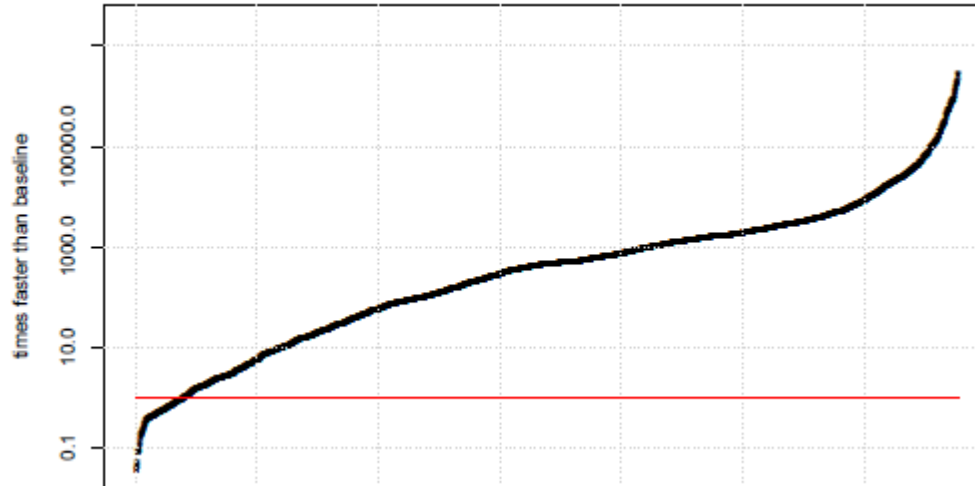
- PostGreSQL is faster for many queries despite IO
- But if it is slower, it is **much slower** (log scale)

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)

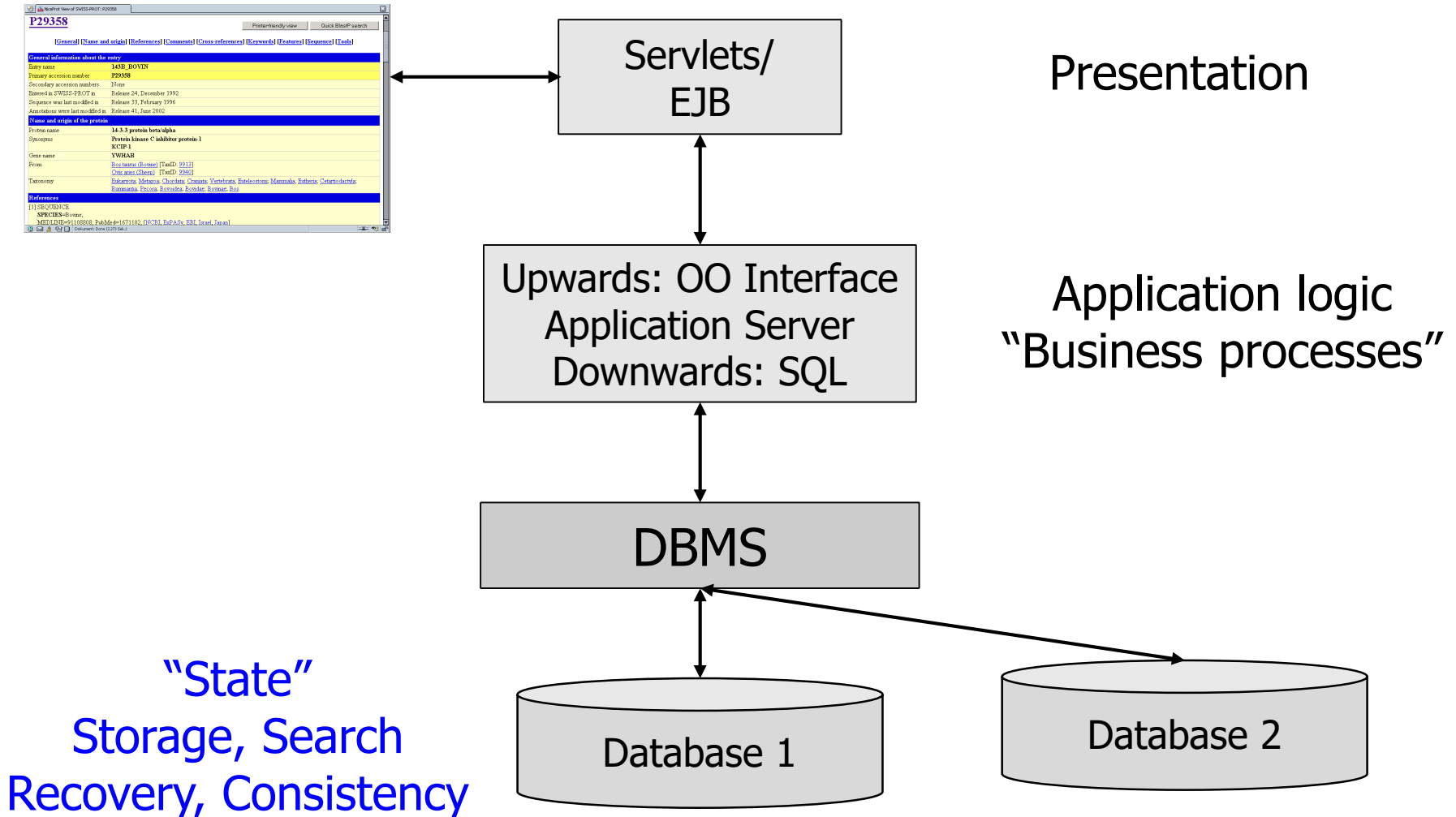
Thomas Krause [Krause, Leser, Lüdeling, 2017]



- 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

Classical Three-Tier Architecture



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**DBS1/2
- 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 **application independent** languages and modelsDBS1
- Systems crash
 - Crashes cannot be avoided
 - **Protect consistency** by logging, constraint enforcement, ...DBS1/2

DBS2: Implementation of Database Systems

- Lecture 4 SWS
 - Monday, 11 – 13 , 3.113
 - Thursday, 11 – 13 , 3.001

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

Exercises and Examination

- Exercises run by Patrick Schäfer
 - Presence & commitment are necessary
 - Implementation of file-/ buffer-/ index manager in C++
 - Monday / Wednesday, 13-15,
 - Both are in 4.112
 - Starts 28.10 / 31.10 (in two 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
 - no substitution for **reading a book**

Implementierung von Datenbanksystemen — Wissensmanagement in der Bioinformatik - Mozilla Firefox

www.informatik.hu-berlin.de/forschung/gebiete/wbi/teaching/archive/ws1213/vl_dbx2

Professor Ulf Leser

Diese Modul vermittelt einen Überblick über Techniken zur Implementierung von (relationalen) Datenbanksystemen. Es behandelt dazu ausgewählte Themen aus allen Ebenen eines DBMS, angefangen bei Satz- und Tabellenverwaltung über (multidimensionale) Indexstrukturen zur Anfrageoptimierung und zum Transaktionsmanagement. Das Modul ist systemnah und behandelt seine Themen im Detail.

Der Halbkurs wird von einer Übung begleitet, in der verschiedene Komponenten eines DBMS implementiert werden.

Voraussetzungen

Voraussetzung für den Besuch sind gute Kenntnisse in Algorithmen (Sortieren, Suchen, Baumsuchverfahren), relationalen Datenbanken (SQL, Transaktionen, Schemaentwurf) und in Betriebssystemen (Pufferverwaltung, Caching). Die Übung verlangt gute Kenntnisse in C oder C++.

Prüfungen und Anrechenbarkeit

Je nach Teilnehmerzahl sind Prüfungen mündlich oder schriftlich. Die Prüfungsform wird in der ersten Semesterwoche bekannt gegeben. Voraussetzung für die Prüfung ist das Bestehen der Übung.

Das Modul ist anrechenbar für

- Diplomstudiengang Informatik, Halbkurs Praktische Informatik, 8SP
- Master Informatik, 10 SP
- Master Wirtschaftsinformatik, 10 SP

Literatur

- Saake, Heuer, Sattler: "Datenbanken: Implementierungstechniken", MTP Verlag, 2. Auflage, 2005
- Garcia-Molina, Ullman, Widom: "Database System Implementation", Prentice Hall, 2000
- Weitere Literatur und Links

Themen und Folien

(Folien sind hier jeweils vor der Vorlesung als PDF verfügbar. Änderungen möglich).

- Einleitung und Motivation
- Architektur von Datenbanksystemen und Übersicht über die Vorlesungsthemen
- Sekundärpeicher; RAID
- Records und Blöcke
- Caching; Dateiformate, Indexing
- Hashing, Extensible Hashing, B und B* Bäume
- Multidimensionale Indexstrukturen: Partitioned hashing, Grid file, kdb Baum, R Baum)
- Grundlegende Anfrageoperatoren
- Join Methoden - Nested Loop, Sort-Merge, Hash-Join, Index-Join
- Anfrageoptimierung
- Gastvortrag: tba
- Datenbankstatistiken; Histogramme; Sampling
- Transaktionen; Logging und Recovery: REDO/UNDO
- Serialisierbarkeit und Sperren
- Abschluss

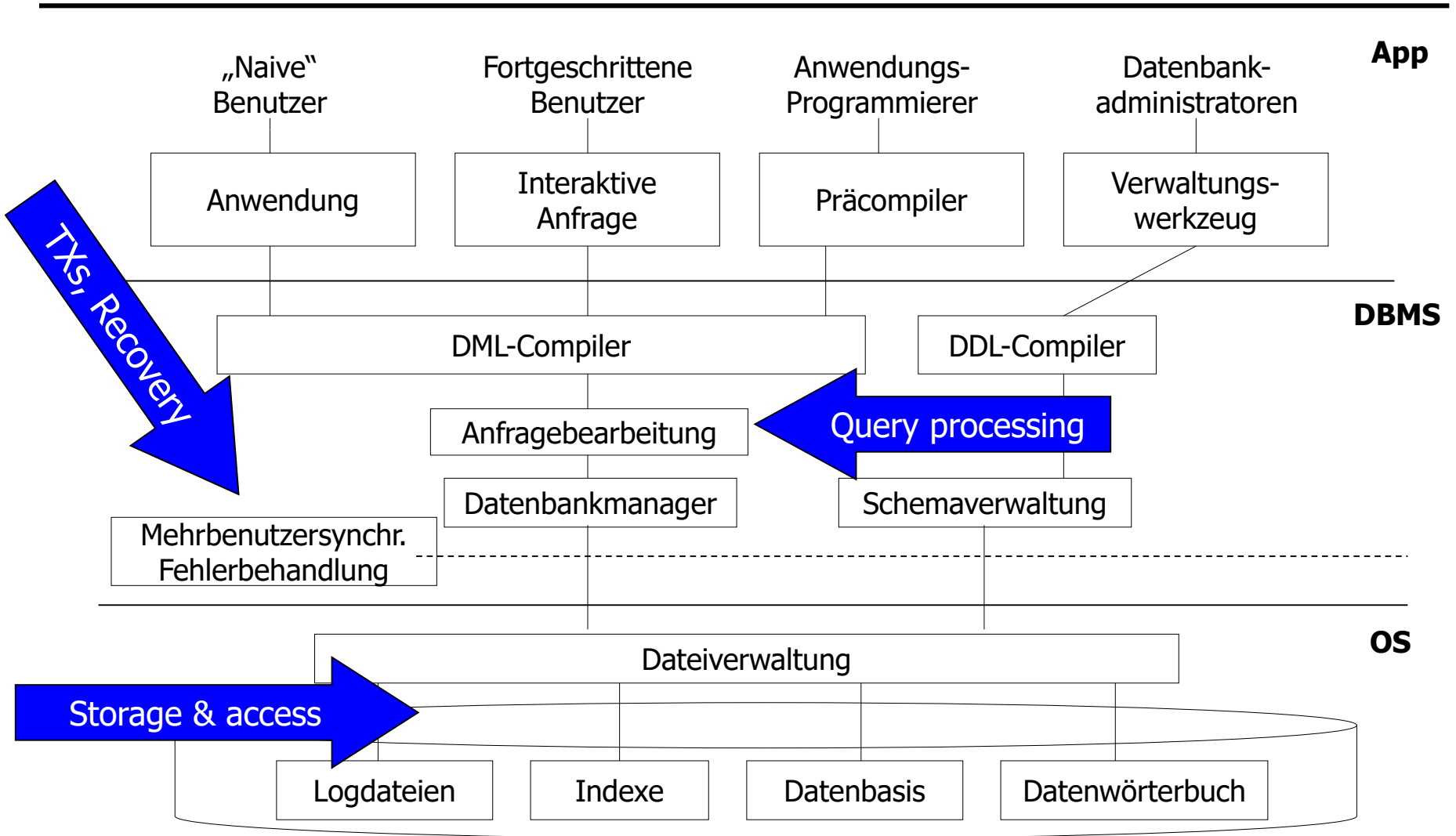
Weitere Materialien und Literatur

- Kemper, Eickel: "Datenbanksysteme – Eine Einführung", Oldenbourg, 5. Auflage 2004
- Härder, Rahm: "Datenbanksysteme. Konzepte und Techniken der Implementierung", Springer, 2. Auflage 2001

Literature

- Primary
 - Saake, Heuer, Sattler "Datenbanken: Implementierungstechniken", 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

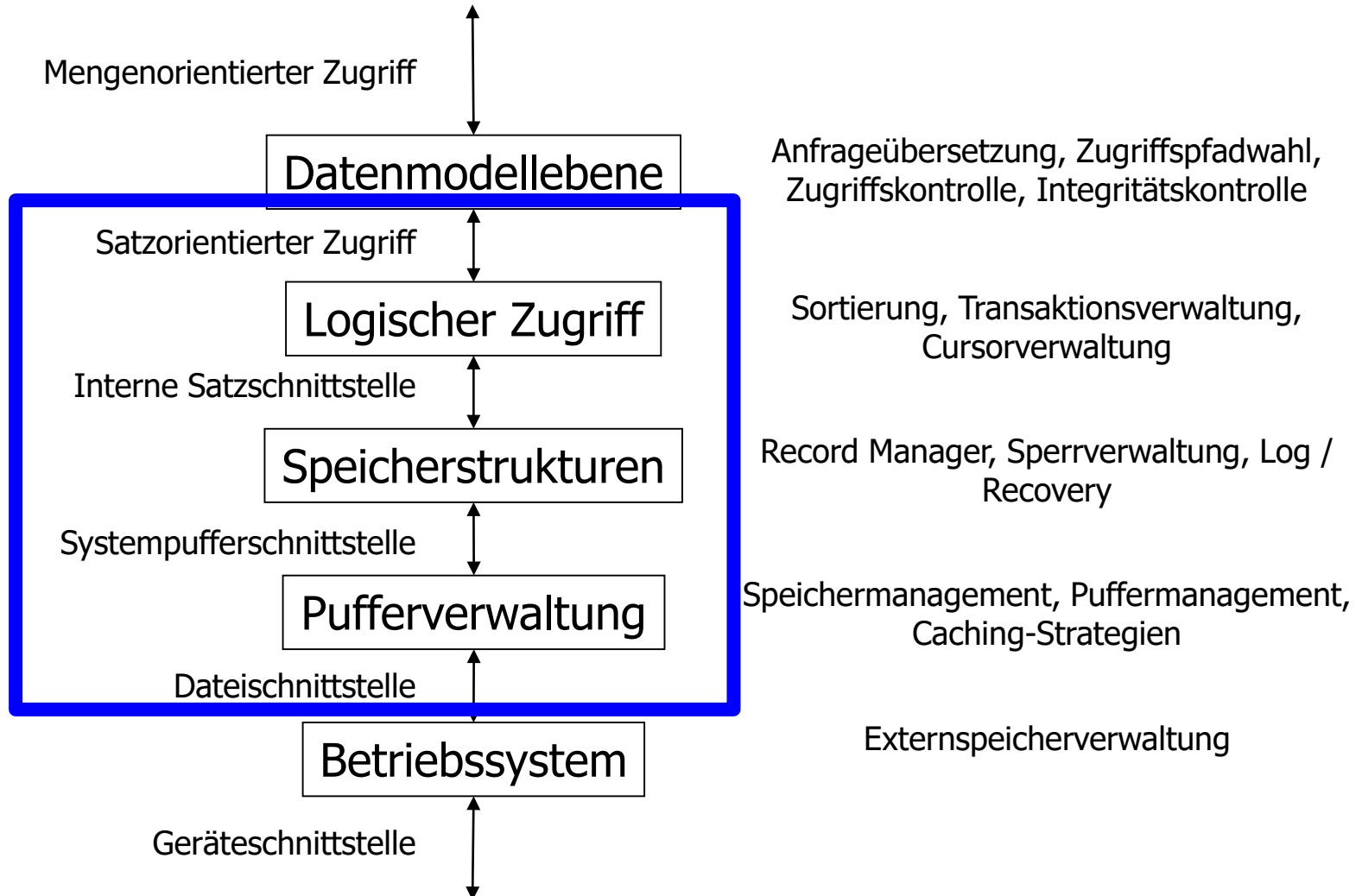
Überblick



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

5 Schichten Architektur



Guests

- TBA ...

Feedback 2012 / 2013

Alter	Geschlecht	Gefehlt	Teilnehmerzahl	Warum kommen?	Studiengang	Fachsemester	Freundlich	Fragen	Sprache	Präsentation	Beispiele	Konzeption	Überblick	Viel neues	Kritische Auseinandersetzung	Nützlich	Lernziele	Materialien	Tempo	Schwierigkeit	Arbeitsaufwand	Dozent	Vorlesung	Abweichung vom Optimum
21	M	1	3	2	BA	5	6	5	6	5	6	5	5	6	4	6	5	5	3	3	3			8
28	M	0	3	2	?	5	6	6	4	6	6	6	6	6	5			5	3	3	3	1	1	4
35	M	4	3	2	DI	8	6	6	6	4	6	6	6	6	5	6	6	5	3	3	4	1	1	4
25	M	5	3	2	DI	11	6	6	6	4	6	5	5	6	4	5	5	5	3	3	3	1	1	9
35	M	2	3	2	?	6	6	6	6	6	6	6	6	5	6	5	5	5	2	2	3	1	1	5
24	M	0	3	1,2,3	DI	9	6	6	6	6	6	6	6	6	6	6	6	6	3	3	3	1	1	0
25	M	1	3	2,3	MW	3	6	6		5	6	5	6	6	6	6	6	5	3	3	3	1	1	3
22	M	4	3	2	MI	1	6	5	6	5	5	6	6	6	4	4	5		3	4	3	1	2	10
24		5	2	2	MI	1	6	6	6	5	5	5	6	6	5	5	6		3	4	3	1	1	6
22	M	0	2	2	MI	1	6	6	5	6	6	6	6	5		6	6	4	3	4	3	1	1	5
24	M	0	3	2	MI	1	6	5	6	5	5	6	6	5	4	5	6	4	2	3	3	1	2	11
25,9		2,0	2,9			4,6	6,00	5,73	5,70	5,18	5,67	5,67	5,82	5,55	5,14	5,44	5,63	4,86	2,82	3,18	3,09	1,00	1,20	
			3,0				6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	3,00	3,00	3,00	1,00	1,00	
			0,10				0,00	0,27	0,30	0,82	0,33	0,33	0,18	0,45	0,86	0,56	0,38	1,14	0,18	-0,18	-0,09	0,00	-0,20	
							0,00	0,27	0,30	0,82	0,33	0,33	0,18	0,45	0,86	0,56	0,38	1,14	0,18	0,18	0,09	0,00	0,20	6,28

Feedback „016/2017

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1		Konzeption	Viel neues	Lernziele	Materialien	Rhetorik	Beispiele	Klare Struktur	Verbindungen	Klar verständlich	Übung hilft	Verständliche Antworten	Kritische Auseinandersetzung	Gute Atmosphäre	Tempo	Schwierigkeit	Arbeitsaufwand	Dozent	Vorlesung	Abweichung vom Optimum	Abweichung pro Frage	Gefehlt	Warum kommen?	Studiengang	Fachsemester
2		6	6	6	5	6	6	6	6	5	5	6	6	6	3	3	3	1	1	4	0,22	2	4	M	1
3		5	4	4	4	5	5	5	5	4	5	6	6	6	3	4	3	2	2	19	1,12	2	3,4	M	5
4		6	6	6	5	6	5	6	6,0	6	5	5,0	6	6	3	3	3	1	1	4	0,22	2	3,4	M	1
5		6	6	6	6	6	6	6	6	4	6	6	6	6	3	3	3	1	1	2	0,13	0	3,4	M	9
6		6	6	5	6	6	5	5	5	6	6	6	6	6	3	3	3	1	1	4	0,25	4	4	BA	9
7		5	6	6	5	6	6	6	6	6	6	6	6	6	3	3	3	1	1	2	0,12	1	1,4	M	1
8		5	5	5	5	6	5	6	6	6	4	6	6	6	3	3	3	2	2	9	0,50	4	4	M	3
9		6	5	5	6	6	6	6	5	6	6	5	5	6	3	3	3	1	1	5	0,29	4	3,4	M	
10		5	4	5	1	6	6	5	4	6	1	3	2	6	3	2	3	2	3	28	1,56	1	2,3,4	BA	9
11	Durchschnitt	5,56	5,33	5,33	4,63	5,89	5,56	5,67	5,38	5,67	3,80	4,88	5,44	6,00	3,00	3,00	3,00	1,33	1,44			1,7			4,8
12	Wunschzahl	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	6,00	3,00	3,00	3,00	1,00	1,00						
13	Abweichung	0,44	0,67	0,67	1,38	0,11	0,44	0,33	0,63	0,33	2,20	1,13	0,56	0,00	0,00	0,00	0,00	-0,33	-0,44						
14		0,44	0,67	0,67	1,38	0,11	0,44	0,33	0,63	0,33	2,20	1,13	0,56	0,00	0,00	0,00	0,00	0,33	0,44	9,66					

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
- ...