# Database Systems

## Spring 2013

1. Exam and syllabus
2. DBS courses at IfI
3. BSc theses, etc.
4. Outlook and Buzzwords
5. Course evaluation and feedback

# Exam and Syllabus


# The Exam

- The final exam is written and takes place Tuesday, June 18, 10:15 - 12:00 in BIN 1.B.01 (see VVZ web page for details).
- Auxiliary material during exam: 1 A4 sheet with notes and pocket calculator.

- Course web page: http://www.ifi.uzh.ch/dbtg/
- The textbook is Database Systems by Elmasri and Navathe, 6th edition.

# What is Important

- **Being precise** is important.
- **Solving relevant examples** is important.
- Understanding material in detail; **apply** to new examples.

- It is not sufficient to "know about it" or to "reproduce".
- You must understand and apply techniques learned throughout the course.
- Exercises are representative for exam and are the best preparation.
- Easy and readable solutions are important.

# Preparation Material

- Exercise with solutions (practice to do exercises yourself before you look at solution)
- Lectures with reviews
- Slides
- Textbook
- Exam 2012, 2011 and 2010 (note that the material has evolved; e.g., definition of B+ tree; mapping from ER to relational DB)
- Open door policy in the database group

# Syllabus/1

- **Relational model, algebra, and calculus**
  - Elmasri and Navathe: chapters 3 and 6
  - relational model
  - relational algebra (RA):
    $\sigma$, $\pi$, $\cup$, $-$, $\times$, $\rho$, $\bowtie$, $\bowtie_\theta$, $\div$, $\leftarrow$, $\vartheta$, $\rtimes$, $\ltimes$, $\rtimes$
  - domain relational calculus (DRC), FOPL
  - tuple relational calculus (TRC)

  - practice Cartesian product
  - practice quantifiers
  - move between RA, DRC, TRC, natural language
  - be precise (e.g., qualified names do not exist in RA; use renaming)

# Syllabus/2

- **SQL**
  - Elmasri and Navathe: chapters 4 and 5
  - data definition language, data manipulation language
  - query expressions, query specifications, orthogonality
  - subqueries
  - duplicates
  - null values
  - logical update semantics

  - practice formulation of declarative queries
  - consider effects of duplicates and NULL values
  - solve and try out SQL solutions with PostgreSQL
    - important is systematic plan: input, output, modular SQL code
    - avoid trial and error
  - be conservative with SQL features

# Syllabus/3

- **Constraints, triggers, views, DB programming**
  - Elmasri and Navathe: chapters 5, 12 and 25
  - views, with clause
  - column constraints, table constraints, assertions, referential integrity
  - functions, triggers, stored procedures
  - expressiveness, recursion

  - know key concepts and their properties
  - know when and how to use these concepts; help to solve specific problems or break down solutions into smaller parts
  - details of extended SQL syntax are not the crucial part

# Syllabus/4

- **Relational database design**
  - Elmasri and Navathe: chapters 14 and 15
  - design goals, redundancy, keys
  - functional dependencies, Armstrong's inference rules
  - 1NF, 2NF, 3NF, BCNF, 4NF
  - normalization algorithm
  - dependency preservation, lossless join decompositions
  - closure, equivalence, minimal cover

  - definition of FDs and MVDs
  - definition of normal forms, dependency preservation and losless join decomposition
  - apply inference rules and normalization algorithm

# Syllabus/5

- **Conceptual database design**
  - Elmasri and Navathe: chapters 7 and 8
  - conceptual design process: ER model, entities, attributes, relationships
  - weak entities, specializations
  - 8 step ER-to-relational mapping

  - ER diagrams: construct ER diagrams from real world descriptions
  - no unique solution; make clarifying assumptions during design
  - analyze strengths and weaknesses of an ER diagram
  - use and apply 8 step mapping algorithm

# Syllabus/6

- **Physical database design**
  - Elmasri and Navathe: chapters 16 and 17
  - seek time, latency, block read time
  - file and buffer manager
  - indexing: secondary and primary index
  - B+ tree
  - extendable hashing

  - compute basic characteristics (nr of blocks, nr of IOs, etc)
  - know definitions of B+ tree and extendable hashing
  - use B+ tree definition and algorithms from slides (other solutions are not valid)

# Syllabus/7

- **Query processing and optimization**
  - Elmasri and Navathe: chapters 18
  - measures of query cost
  - sorting (external sort merge)
  - selection (scan, binary search, index)
  - join (nested loop, block nested loop, sort merge, hash join)
  - algebra trees, evaluation plans
  - heuristic and cost-based query optimization

  - compute cost of operations
  - transformation (rewriting) of relational algebra expressions
  - interpretation of query plans

# Database System Courses at IfI

# Database Systems Courses @IfI

- Database Systems, Spring (sem4)
- Praktikum Datenbanksysteme, Fall (sem5)
- Distributed Databases, Fall (sem5; not HS13)
- Database Management and Performance Tuning, Fall (sem5, sem7)
- Seminar in Database Systems, Spring (sem6, sem8, sem12)
- XML Databases, Spring (sem6, sem8)
- Data Warehousing, Spring (sem6, sem8; even years)
- Nonstandard Databases, Fall (sem7, sem9; not HS13)

- Vertiefung, BSc Arbeit, MSc Arbeit

# Andreas Geppert: Praktikum Datenbanksysteme

- Since 2001 Andreas Geppert has been with Credit Suisse.
- Before this he was a senior researcher in the Database Technology Research Group.
- He received his diploma in Computer Science from the University of Karlsruhe (Germany) in 1989 and his PhD in computer science from the University of Zurich (1994).
- From August 1998 to August 1999 he was a visiting scientist at the IBM Almaden Research Center.
- Praktikum Datenbanksysteme
  - Apply/practice your SQL knowledge on a case study
  - Dienstag 16:00 - 18:00

# Praktikum Datenbanksysteme

- Inhalt:
  - relationale Datenbanksysteme
  - konzeptueller und logischer Entwurf
  - Anfragesprachen
  - Trigger, Stored Procedures
  - Anwendungsentwicklung in Java
- Datenbankverwaltungssystem:
  - PostgresQL (wahrscheinlich)
- Anwendung:
  - Reservationssystem für eine Car-Sharing-Genossenschaft (von DB bis Web)
  - Bewirtschaftung des Fuhrparks, der Mitglieder und der Stationen

CREDIT SUISSE

Produced by: Andreas Geppert, KIVB
Date: May 2010 Slide 2

# Pei Li: Database Management and Performance Tuning

- Pei Li is a senior researcher in Database Technology Group at IfI. She got her PhD degree from the University of Milano - Bicocca in 2013.
- Lecture type: Lecture with exercises
- Course content: The course aim to give an in-depth understanding of the features that off-the-shelf database management systems offer, in particular with respect to system performance. This knowledge is used to tune the database system and its environment: dimension the hardware for the database system, write efficient queries, set effective indexes, communicate with the database efficiently, and diagnose performance problems.

# Pei Li: Database Management and Performance Tuning

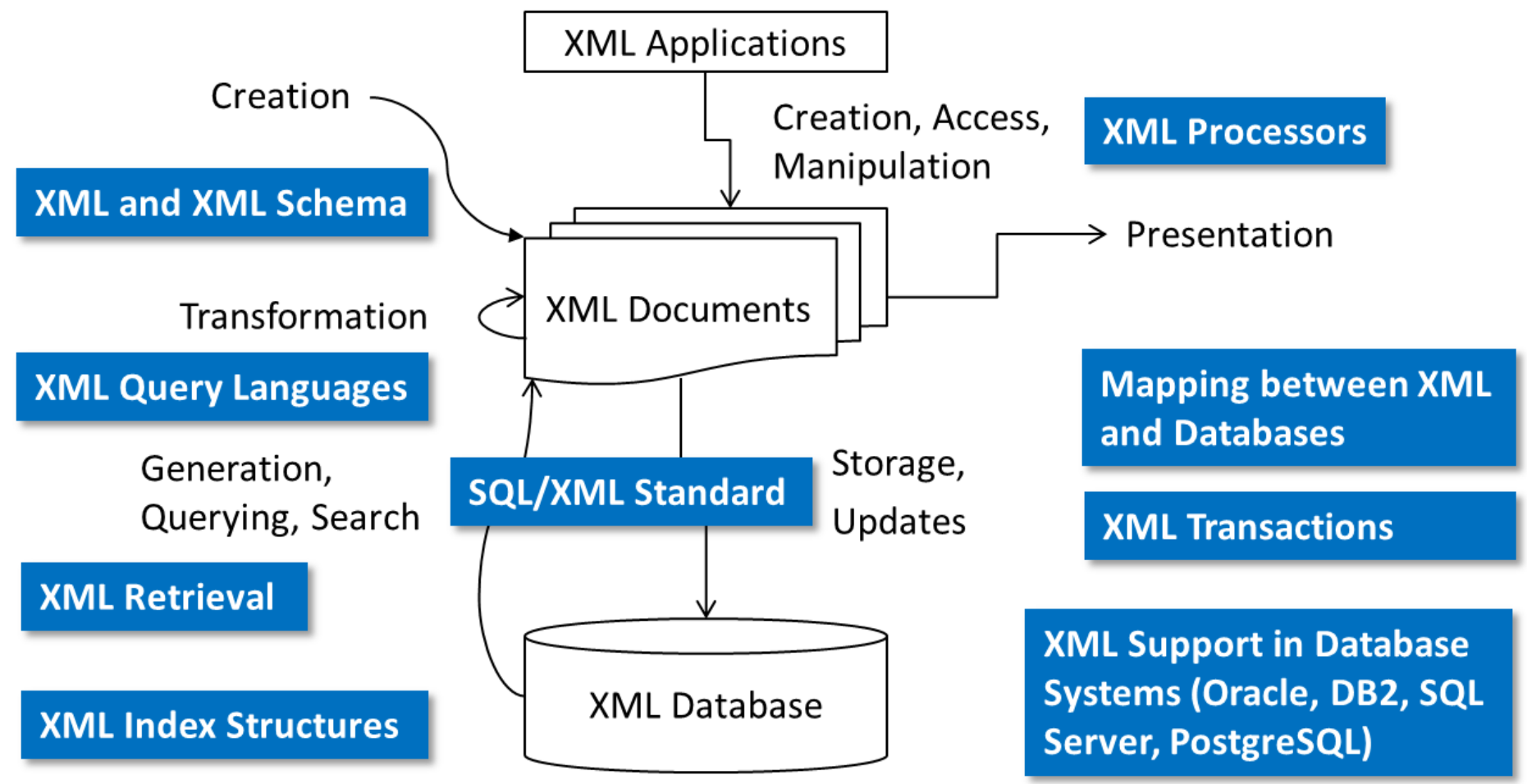
- When?
  - Fall semester 2013
  - Friday 14:00 - 16:00 (tentative)
- What you will learn?
  - Basic principles of how to tune database applications
  - Performance criteria for choosing a database management system
  - Testing particular aspects of systems with experimental data and scripts
- You are expected to:
  - Have background knowledge in programming, database system, data structures and algorithms

# Can Türker: XML and Databases

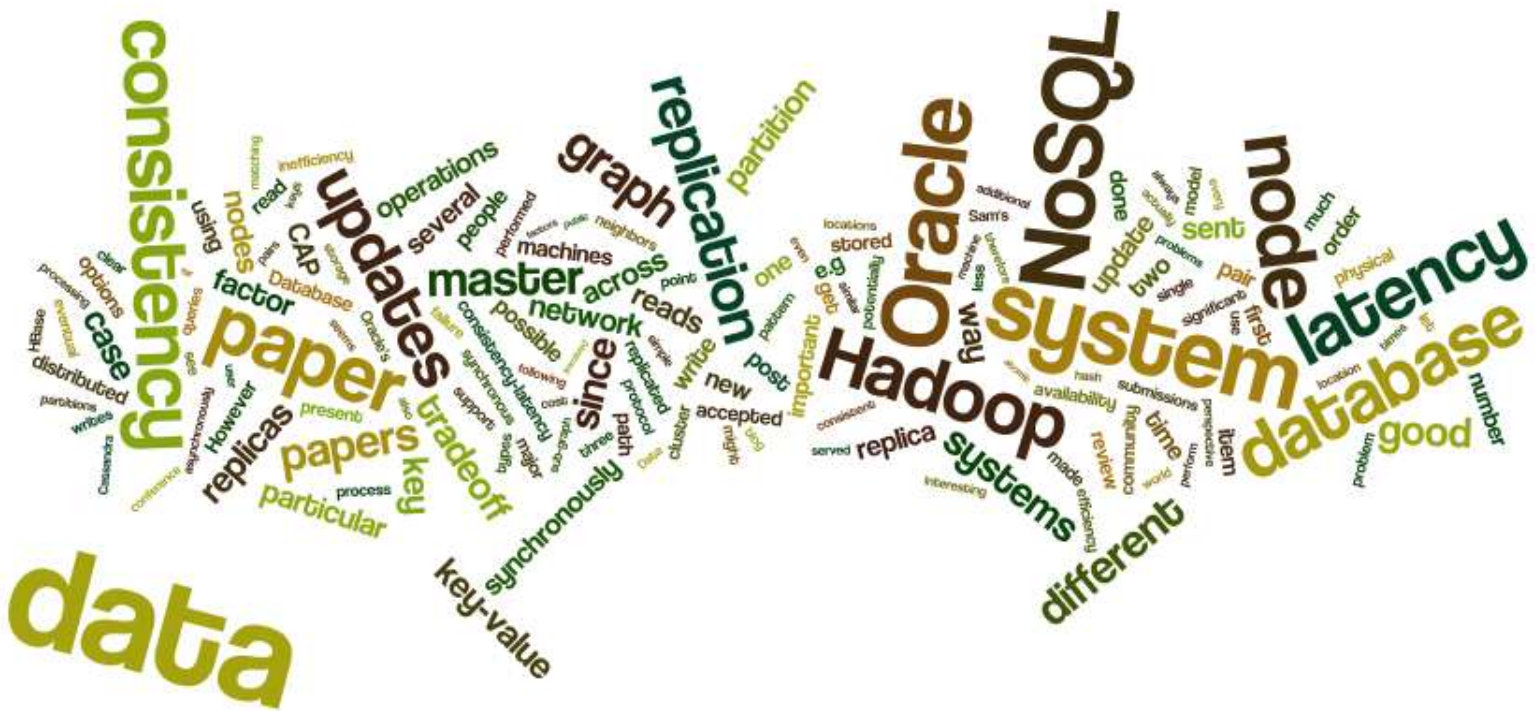
- Dr. Can Türker heads the Data Integration Group of the Functional Genomics Center Zurich (FGCZ). He holds a Ph.D. degree in computer science (1999) and is (co-)author of the several lecture books in the area of databases, among others 'Object-Relational Databases' and 'SQL:1999 & SQL:2003'.
- Lecture Hours: Thursday 8-10am
- Course content: XML is introduced with related technologies and it is shown how XML can be used for storing, accessing, querying, and updating data. The mapping between XML and databases as well as specific requirements arising from the usage of XML for data management are elaborated not only conceptually but are also demonstrated practically using todays major database systems.
- Prerequisite: The course expects background knowledge in database systems (especially in SQL).

# Can Türker: XML and Databases

Goal: This lecture deals with the interplay of two essential technologies, namely XML and databases.

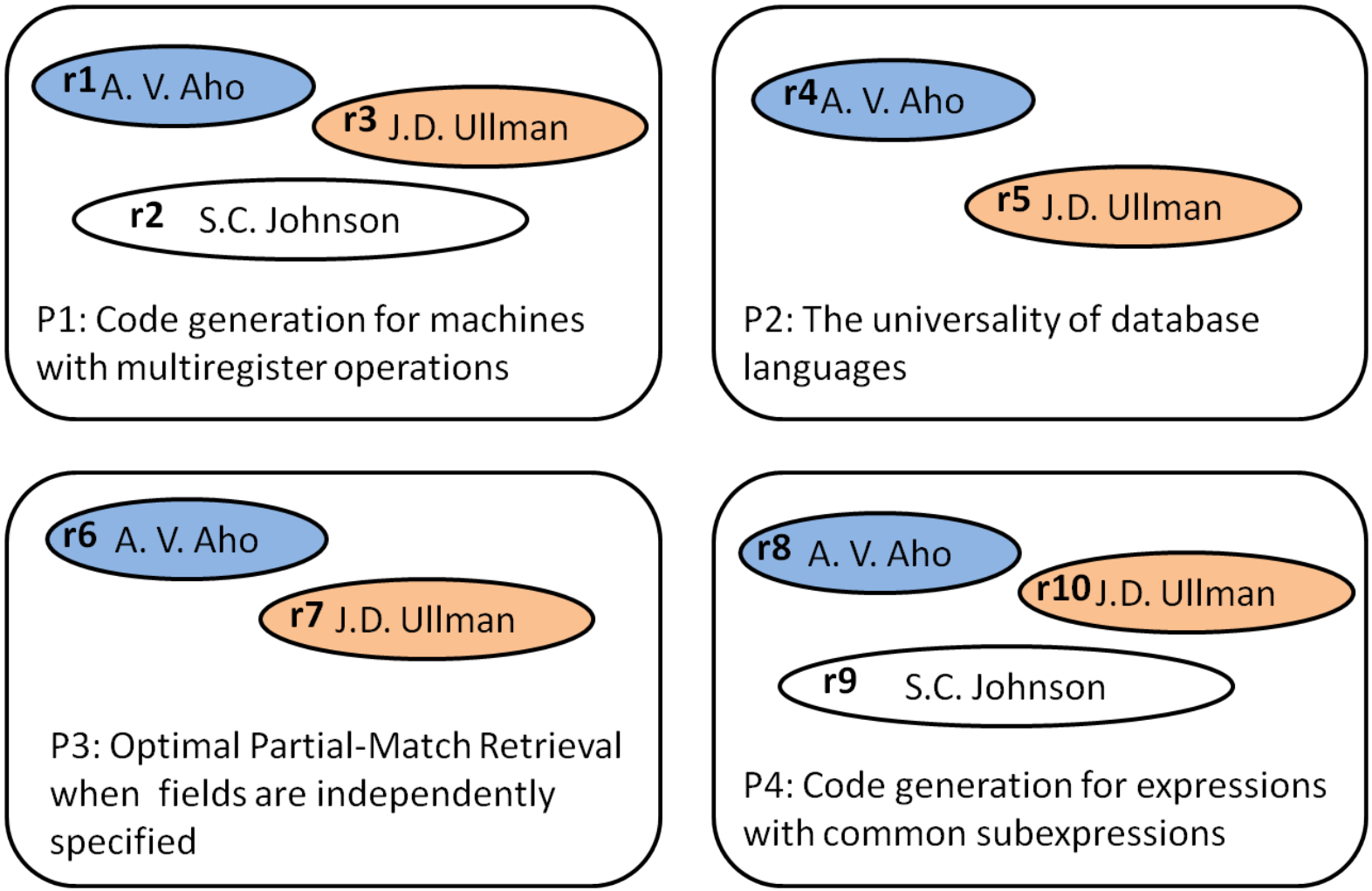


# Topics for BSc theses, MSc theses, etc in the DBS Area



# Pei Li: Large-scale Entity Resolution Algorithms with MapReduce

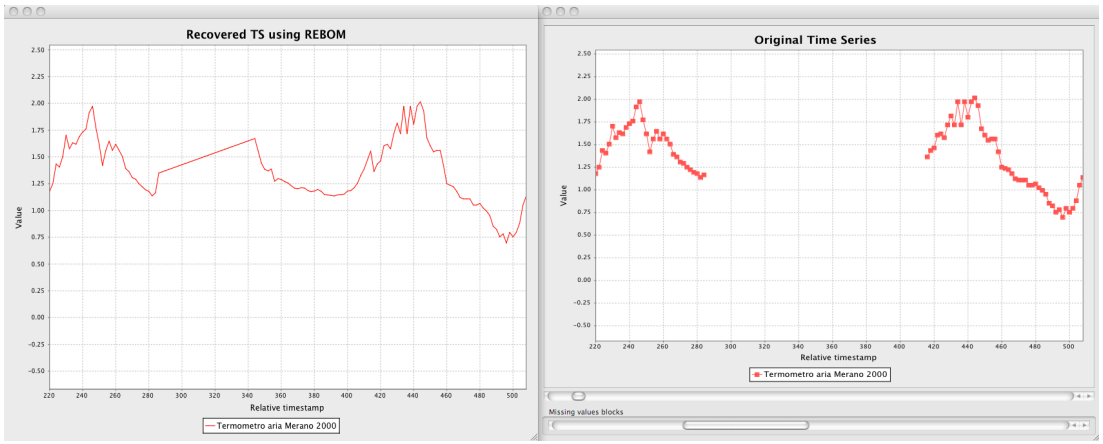
- Large-scale entity resolution (ER) algorithms in MapReduce



# Pei Li: Large-scale Entity Resolution Algorithms with MapReduce

- Outset
  - Similarity relationships of the data pose challenges for conventional "loosely-coupled" shared-nothing programming paradigms;
  - Efficient entity resolution solutions require to minimize I/O and network costs;
  - Large-scale datasets require adaptive load balancing strategies;
  - Iterative entity resolution algorithms pose challenges for state-of-the-art parallel systems that are based on acyclic data flows.
- $\Rightarrow$ use Hadoop with MapReduce techniques for entity resolution

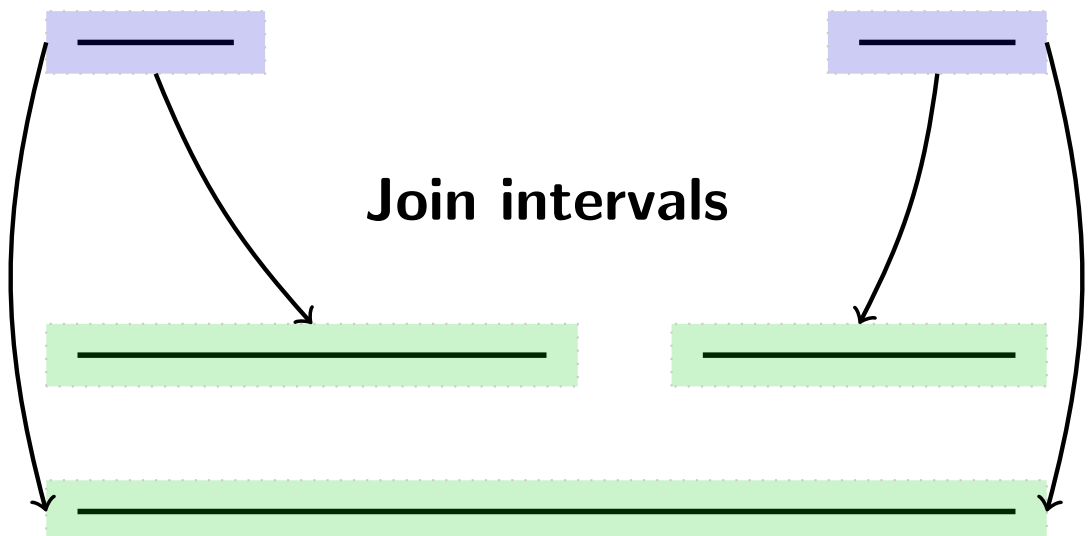
# Mourad Khayati: Recovery of Missing Values in Time Series




- Implement matrix decomposition techniques
- Apply the implementation for the recovery of missing values in real world time series data
- SQL-based implementation of the Centroid Decomposition method
- Memory efficient implementation (Java, C, or C++) of the Incremental Centroid Decomposition method
- Large scale empirical evaluation on real world hydrological time series (Oracle database)

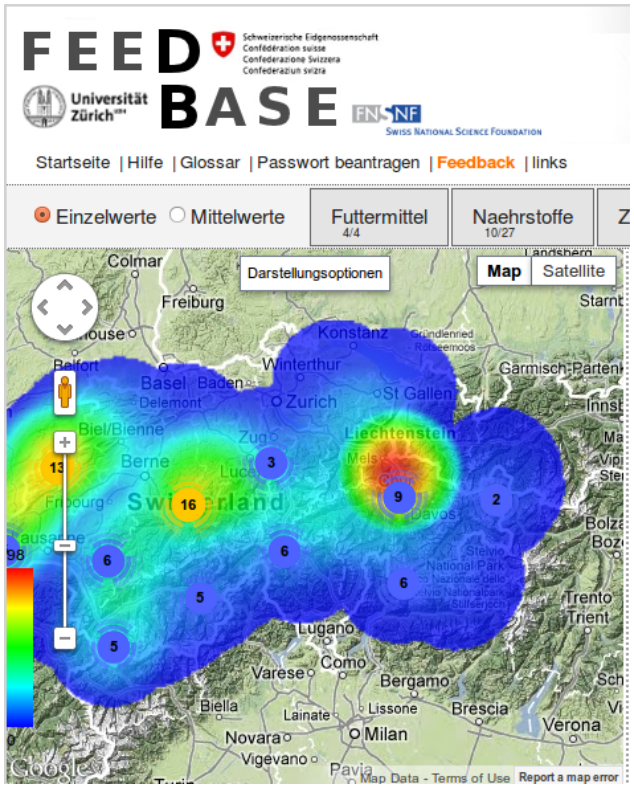
# Anton Dignös: Parallel Join Computation using Interval Partitioning

- Goal:
  - CPU and IO parallelization of join using interval partitioning
  - Implementation in C, C++, or Java
- What you will learn
  - Efficient partitioning of intervals for joins
  - Theory and practice of parallel computing
  - Limitations according to Amdahl's law



# Andrej Taliun: Swiss Feed Data Database

- The **Swiss Feed Data Database** contains temporal and spatial information about animals feeding in Switzerland.
- Feed rations for animals are optimized to meet nutrient requirements. The efficient search for feed types that best match these requirements has to rely on query options that allow multiple, user defined constraints in the nutrient selection step.
- So far, the search function is restricted to feed type selection and nutrient selection.
- A search additionally based on user defined nutrient ranges would help to solve many real world situations efficiently.



# Francesco Cafagna: The Feed Data Warehouse(1)

- **Implementing a Hash Nearest Neighbor Join** BScThesis - Facharbeit
  - The goal is to realize inside the DBMS of Postgres a hash indexed solution for computing nearest neighbor joins. [C++]

R

| Time |
|---|
| 03-02-2010 |
| 01-03-2010 |
| 12-04-2010 |

S

| | Time | Value |
|---|---|---|
| | 04-02-2010 | 3.4 |
| h(02) | 09-02-2010 | 3.3 |
| | 12-03-2010 | 3.3 |
| h(03) | 22-03-2010 | 3.1 |
| | 11-04-2010 | 3.1 |
| h(04) | 30-04-2010 | 2.9 |
| | 11-05-2010 | 3.1 |
| h(05) | 25-05-2010 | 2.9 |
| | 09-06-2010 | 3.1 |
| h(06) | 25-06-2010 | 3.0 |

$h(t)$ returns the month of timestamp $t$

For a given time $t \in$ R, the join matches in S are always in $h(t)$ or $h(t) \pm 1$. $\Rightarrow$ No need to make nested loop!

block fetched from disk  block in buffer mem.

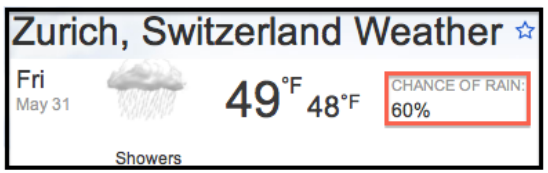
# Francesco Cafagna: The Feed Data Warehouse(2)

- **Lineage of derived nutrients** - Vertiefung - Facharbeit
  - A derived nutrient is a tuple computed using mathematical calculations on other tuples stored in the database. A derived nutrient on a timestamp $t$ is calculated using the tuples thate are the nearest neighbors of $t$.
  - The goal of this project is to find, for a given derived nutrient on time $t$, all the tuples that have originated it, i.e. the nearest neighbors of $t$. [SQL + PHP]

# Katerina Papaioannou: Temporal Probabilistic Databases with Lineage


**Apartments** $^{PT}$

| City | Rooms | T | P |
|---|---|---|---|
| Zurich | 3 | [1/7, 30/9) | 0.7 |




**Weather** $^{PT}$

| City | Weather | T | P |
|---|---|---|---|
| Zurich | Rain | [31/5, 1/6) | 0.6 |

- **Goal:** computation of the probability of the result tuples **correctly** using lineage, i.e. a boolean formula which intuitively captures **"how a tuple was derived"**

**a. Modification of algebra operators for lineage computation**
The creation of a set of reduction rules for algebra operators is expected so that the lineage of the result tuples of an algebra operator can be computed

**b. Probability Computation out of Lineage**
Minimization of the lineage and proper transformation into an algebraic expression whose result will match the probability of a tuple.

# Amr Noureldin: Temporal Procedural Functions

- **Procedural Functions:** facilitate implementing complex functionalities inside a DBMS
- Modifying the behaviour of **PostgreSQL DBMS**, to evaluate such functions over time varying data
  - Requires determining **constant intervals** (intervals when no change in the data is observed)
- Use **Hadoop** (a MapReduce framework) to parallelize the computation of such intervals
  - Previous **Vertiefung** work: Constant interval extraction using Hadoop
  - Current **BSc thesis** in progress: Load-balancing clusters of data in Hadoop, to ensure even distribution of data

# Outlook and Buzzwords

# Outlook and Buzzwords

- Jennifer Widom's recipe for database research:
  1. pick a fundamental assumption underlying database systems
  2. drop it
  3. solve resulting problem

- **semistructured databases, XML databases**
  - drop: fixed structure (schema) of the data

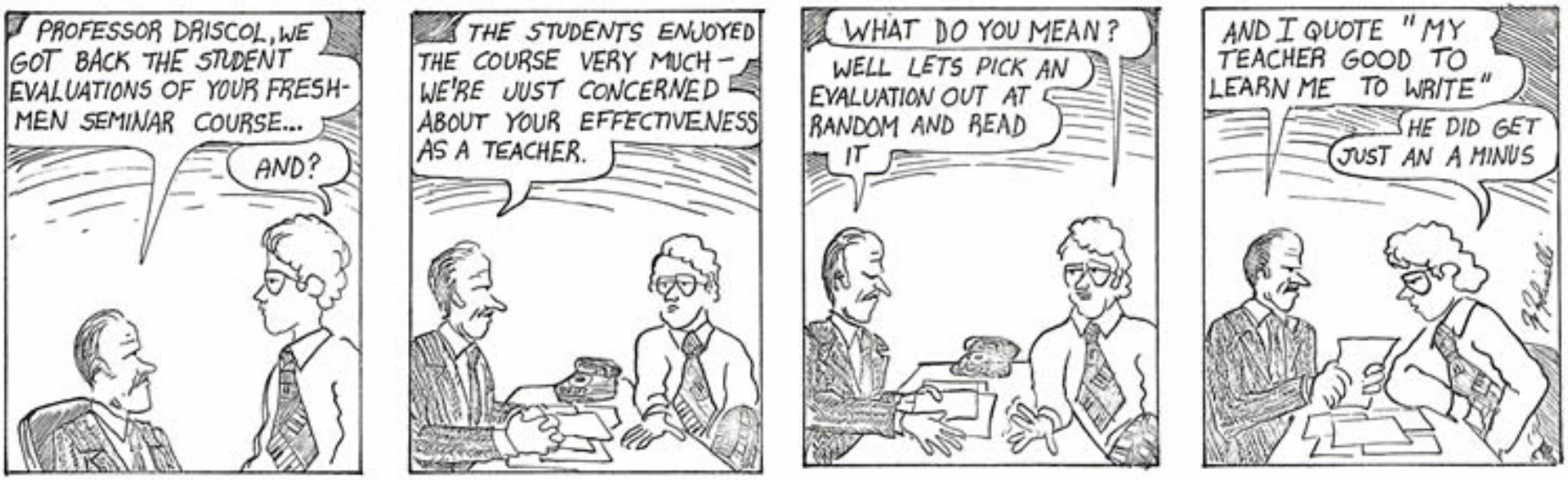
- **stream processing systems**
  - drop: data sets are persistent and disk-resident

- **probabilistic databases**
  - drop: tuple presence is certain

# Outlook and Buzzwords

- **data warehouses with complex analytical queries (OLAP)**
  - drop: banking applications with short update transactions
- **key values stores, NoSQL**
  - drop: sophisticated SQL queries
- **MapReduce**
  - drop: centralized processing of entire database
- **column stores**
  - drop: data is stored row by row

# Course Evaluation and Feedback



# Viel Erfolg!

# Danke.