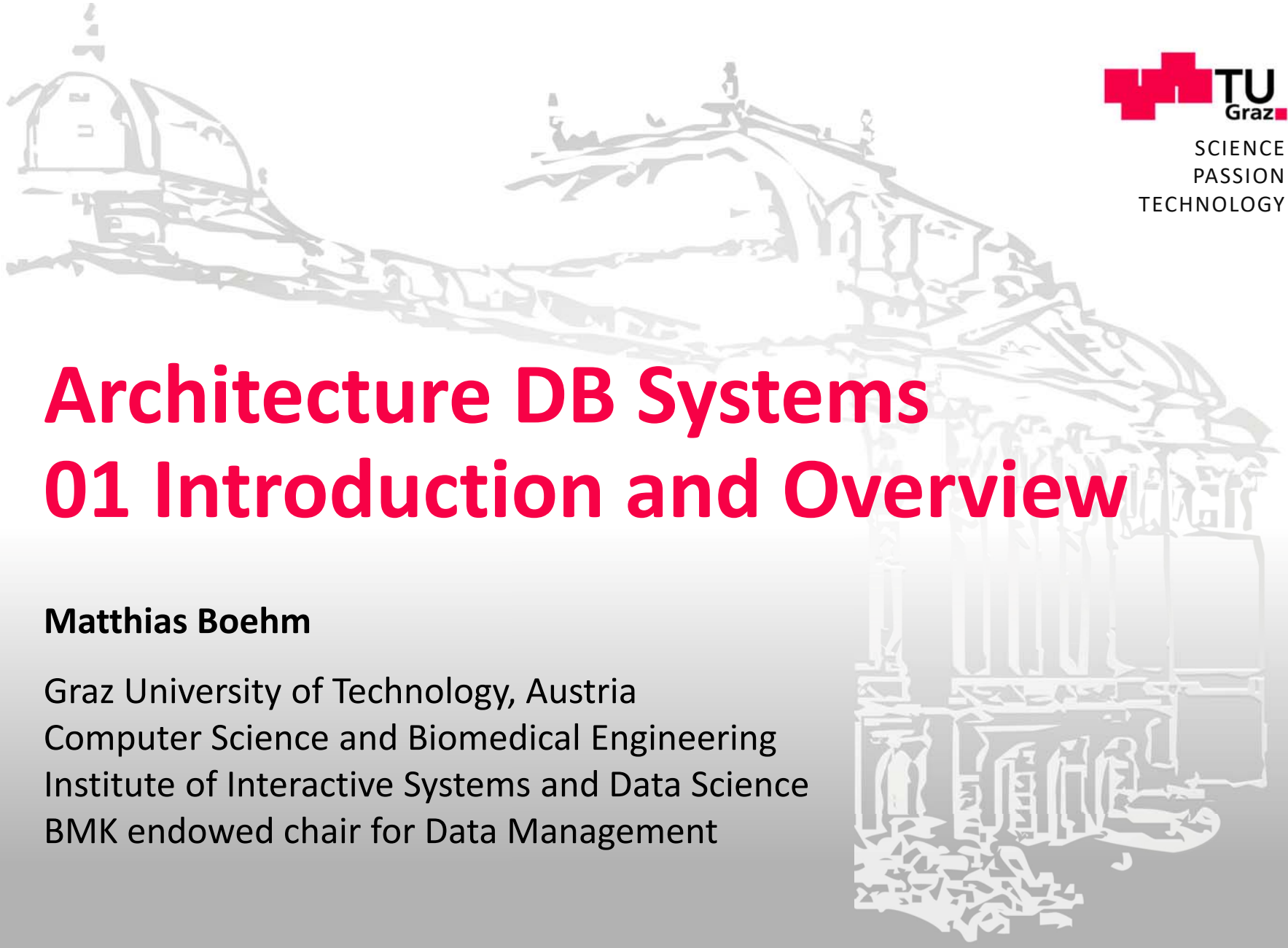


SCIENCE
PASSION
TECHNOLOGY

Architecture DB Systems
01 Introduction and Overview

Matthias Boehm

Graz University of Technology, Austria
Computer Science and Biomedical Engineering
Institute of Interactive Systems and Data Science
BMK endowed chair for Data Management

Last update: Oct 06, 2020

ISDS

Announcements/Org

- **#1 Video Recording**
  - Link in TeachCenter & TUbe (lectures will be public)
  - Optional attendance (independent of COVID)

- **#2 Course Registration** (as of Oct 06)
  - Architecture of Database Systems (ADBS)

73 (0)

- **#3 COVID-19 Restrictions** (HS i5)
  - Max 25% room capacity (TC registrations)
- **#4 Startup Incubator "Gründungsgarage"**
  - 5min-overview by Julia Harrer in DM (see recording)
  - If interested, apply by **Oct 11**

Gründungs garage

Agenda

- **Data Management Group**
- **Course Organization**
- **Course Motivation and Goals**
- **Course Outline and Projects**
- **Excursus: DEXTER Project**

Data Management Group

https://damslab.github.io/

About Me

- **09/2018 TU Graz**, Austria
  - BMK endowed chair for data management
  - **Data management for data science**
    (ML systems internals, end-to-end data science lifecycle)

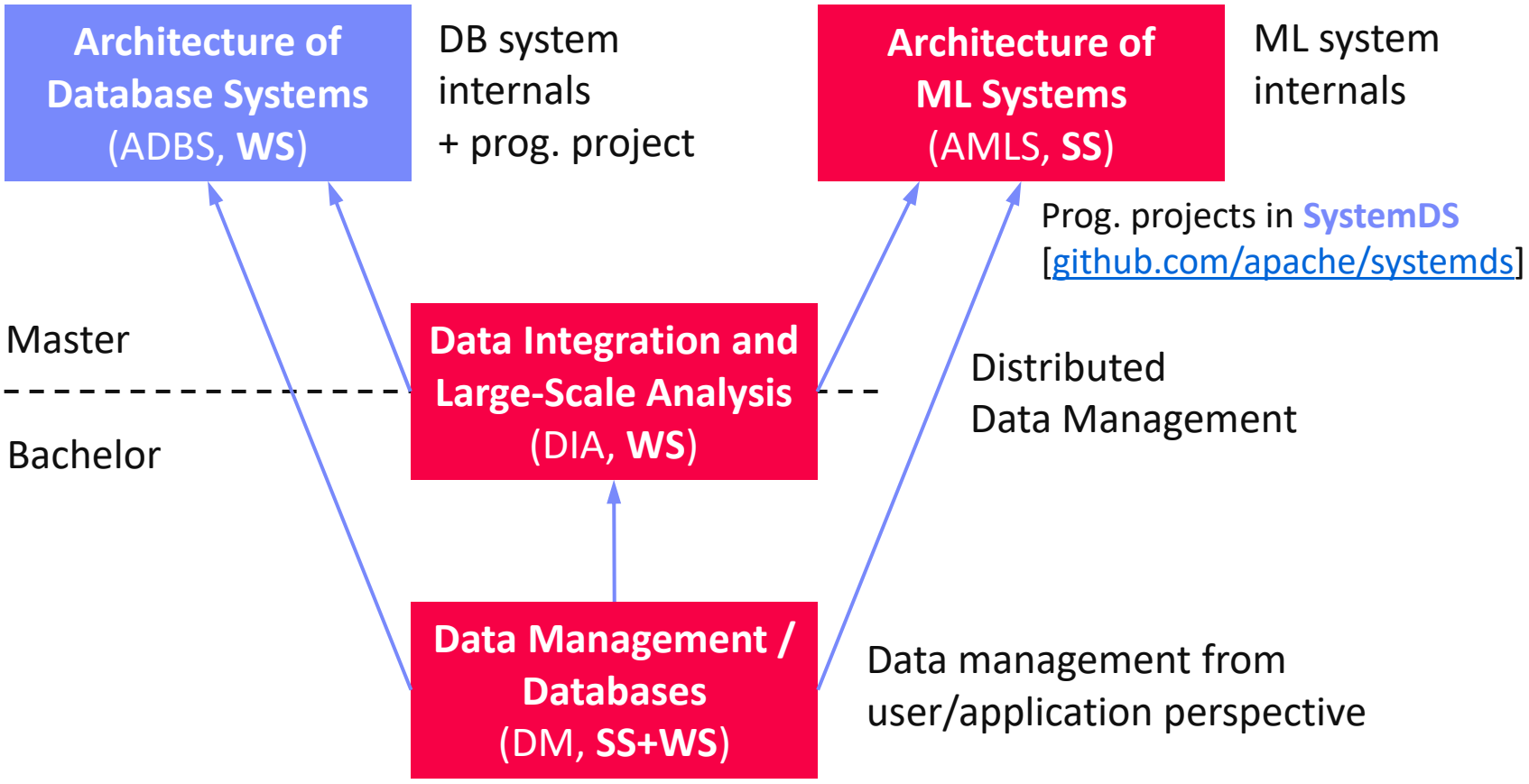
https://github.com/apache/systemds

- **2012-2018 IBM Research – Almaden**, USA
  - Declarative large-scale machine learning
  - Optimizer and runtime of **Apache SystemML**

- **2011 PhD TU Dresden**, Germany
  - Cost-based optimization of integration flows
  - Systems support for time series forecasting
  - In-memory indexing and query processing

TECHNISCHE UNIVERSITÄT DRESDEN
DB group

Data Management Courses



Course Organization

Basic Course Organization

- **Staff**
  - Lecturer: Univ.-Prof. Dr.-Ing. Matthias Boehm, ISDS
  - Assistant: M.Tech. Arnab Phani, ISDS
- **Language**
  - Lectures and slides: **English**
  - Communication and examination: **English**/**German**
- **Course Format**
  - VU 2/1, **5 ECTS** (2x 1.5 ECTS + 1x 2 ECTS), bachelor/master
  - **Weekly lectures** (**Wed 6.15pm**, including **Q&A**), **attendance optional**
  - **Mandatory programming project** (2 ECTS)
  - **Recommended papers** for additional reading on your own
- **Prerequisites**
  - **Preferred:** course Data Management / Databases is very good start
  - **Sufficient:** basic understanding of SQL / RA (or willingness to fill gaps)
  - Basic programming skills in low-level language (C, C++)

Course Logistics

- **Website**
  - https://mboehm7.github.io/teaching/ws2021_adbs/index.htm
  - All course material (lecture slides) and dates
- **Video Recording Lectures (TUbe)?**
- **Communication**
  - **Informal language** (first name is fine)
  - Please, **immediate feedback** (unclear content, missing background)
  - Newsgroup: N/A – email is fine, summarized in following lectures
  - **Office hours:** by appointment or after lecture
- **Exam**
  - **Completed programming project** (checked by me/staff)
  - **Final written exam** (oral exam if <15 students take the exam)
  - **Grading** (40% project/exercises completion, 60% exam)

Course Logistics, cont.

- **Course Applicability**
  - **Master** programs computer science (CS), as well as software engineering and management (SEM)
    - Catalog Data Science (elective course in major/minor)
    - Catalog Software Technology (elective course in major/minor)
  - **Free subject course** in any other study program or university

Course Motivation and Goals

History 1970/80s (relational)

Ingres @ UC Berkeley (Stonebraker et al., Turing Award '14)

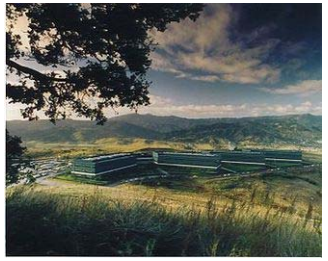
QUEL

SQL Standard (SQL-**86**)

SEQUEL

Oracle, IBM DB2, Informix, Sybase → MS SQL

System R @ IBM Research – Almaden (Jim Gray et al., Turing Award '98)



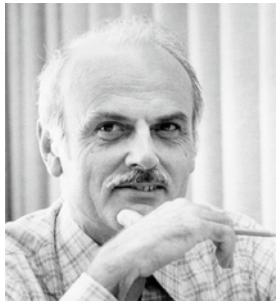
Tuple Calculus

Relational Algebra

Relational Model

Goal: Data Independence (physical data independence)

- Ordering Dependence
- Indexing Dependence
- Access Path Depend.



Edgar F. "Ted" Codd @ IBM Research (Turing Award '81)

[E. F. Codd: A Relational Model of Data for Large Shared Data Banks. Comm. ACM 13(6), **1970**]

Success of SQL / Relational Model

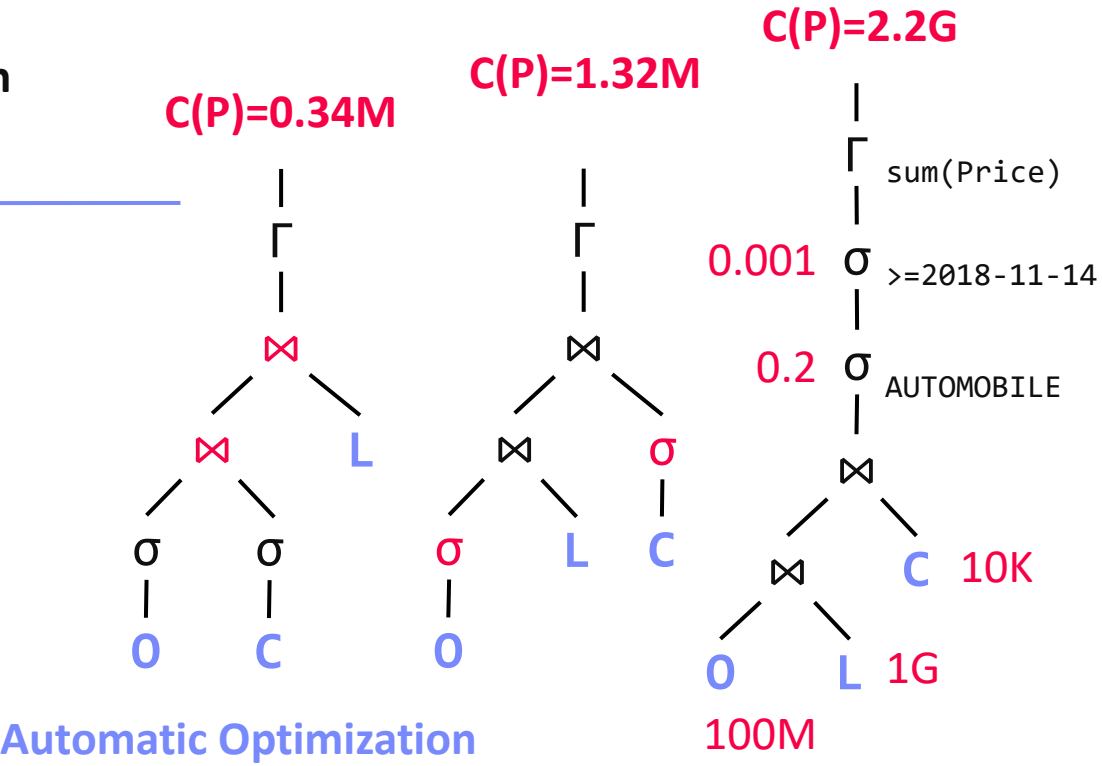
Query:

```
SELECT O_OID, sum(L_Price)
FROM Orders, Lineitem, Customer
WHERE O_OID = L_OID AND O_CID = C_CID
   AND O_Odate >= '2018-11-14'
   AND C_Msegment = 'AUTOMOBILE'
GROUP BY O_OID
```

#1 Declarative:
what not how

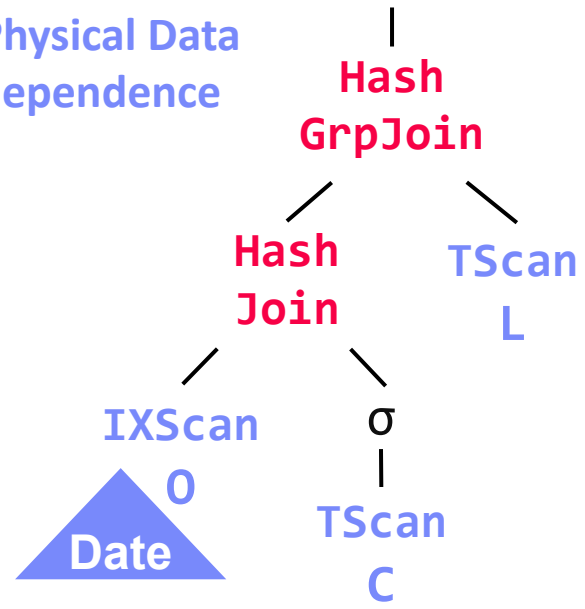
#2 Flexibility:
closure property
→ composability

Logical Query Plans



#3 Automatic Optimization

Physical Query Plan

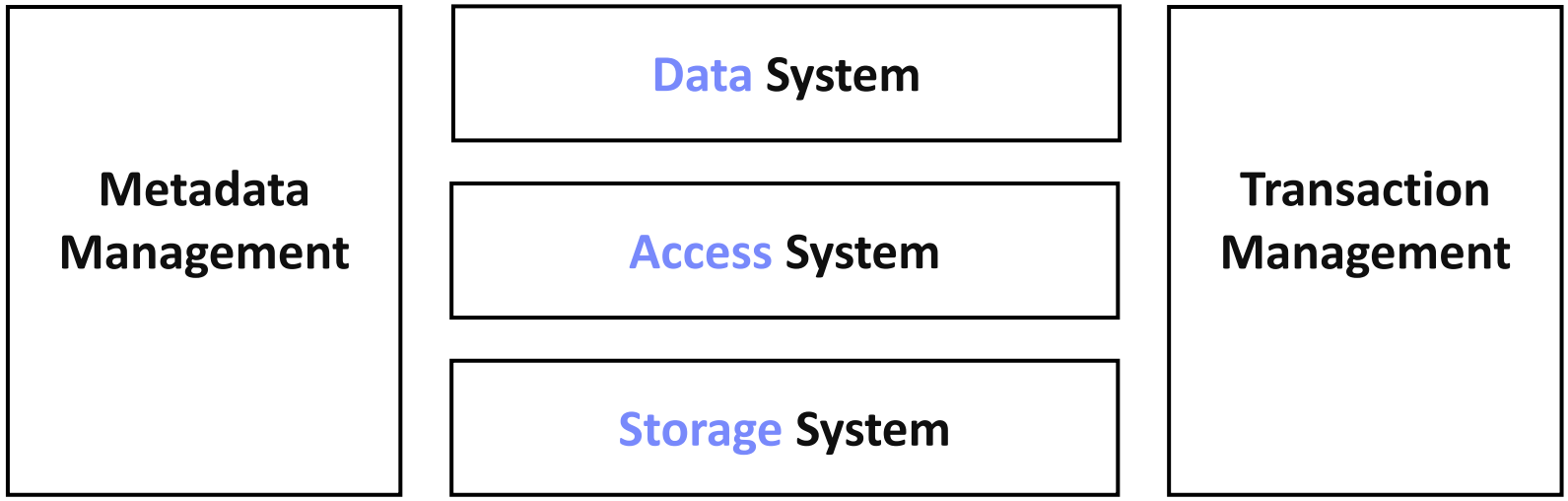


#4 Physical Data Independence

DBMS Architecture

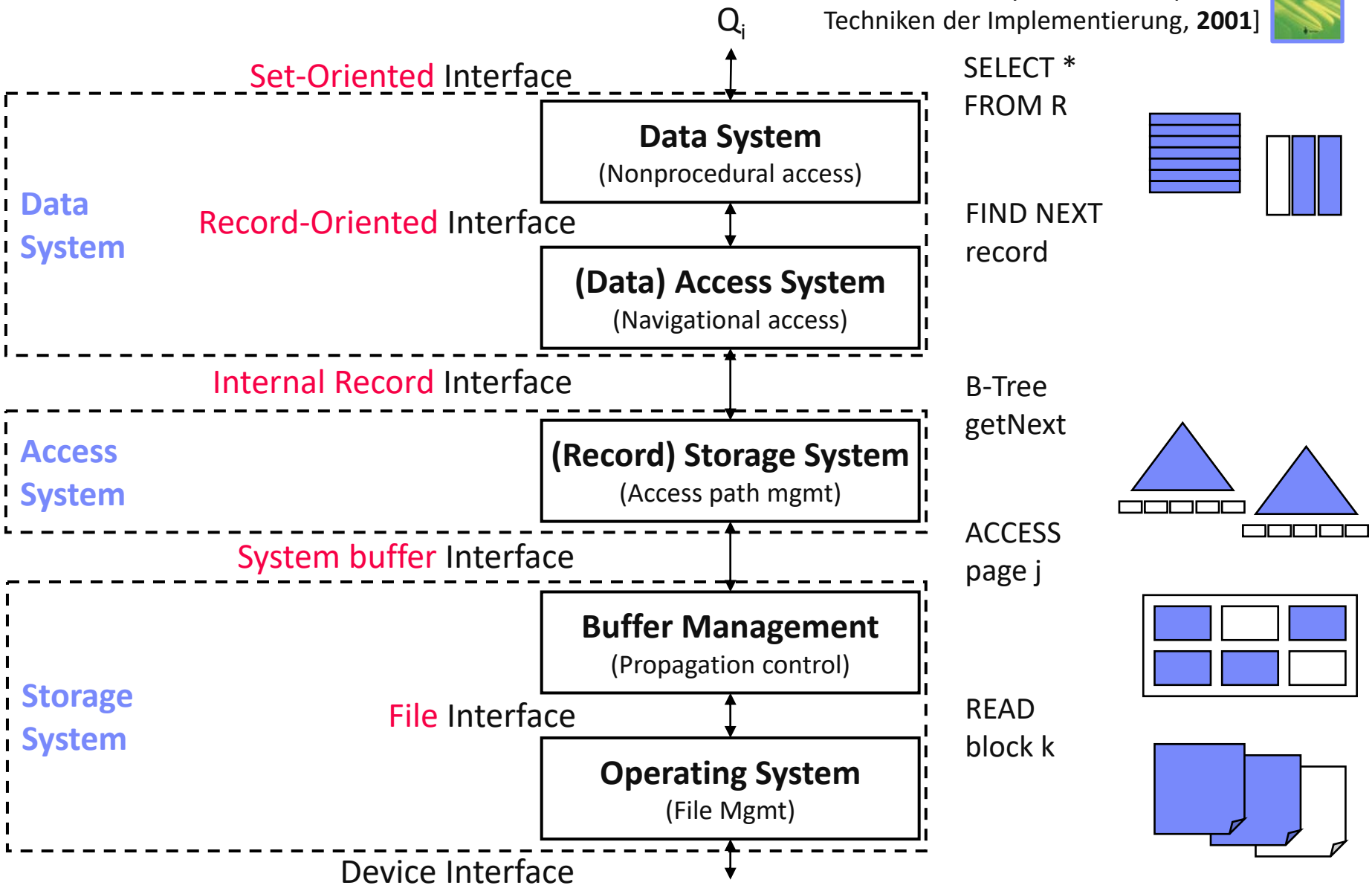
[Theo Härder, Erhard Rahm: Datenbanksysteme: Konzepte und Techniken der Implementierung, **2001**]

- **Coarse-grained System Architecture**



DBMS Architecture, cont.

[Theo Härder, Erhard Rahm: Datenbanksysteme: Konzepte und Techniken der Implementierung, **2001**]



Course Goals

- **Constantly Changing Environment**
  - Heterogeneous and changing **hardware characteristics**
  - New application and **data analysis workloads**
- **#1 Architecture and internals of traditional/modern DB systems**
- **#2 Understanding of DB characteristics → better evaluation / usage**
- **#3 Understanding of effective techniques → build/extend DB systems**
  (these fundamental techniques are **broadly applicable** in other systems)

Course Outline and Projects

Course Outline

A: System Architecture and Data Access

- **01 Introduction and Overview** [Oct 07]
- **02 DB System Architectures** [Oct 14]
- **03 Data Layouts and Bufferpool Management** [Oct 21]
- **04 Index Structures and Partitioning** [Oct 28]
- **05 Compression Techniques** [Nov 04]

B: Query Processing and Optimization

- **06 Query Processing (operators, execution models)** [Nov 11]
- **07 Query Compilation and Parallelization** [Nov 18]
- **08 Query Optimization I (normalization, rewrites, unnesting)** [Nov 25]
- **09 Query Optimization II (cost models, join ordering)** [Dec 02]
- **10 Adaptive Query Processing** [Dec 09]

Course Outline, cont.

C: Emerging Topics

- **11 Cloud Database Systems** [Dec 16]
- **12 Modern Concurrency Control** [Jan 13]
- **13 Modern Storage Management (SSD, NVM)** [Jan 20]
- **14 DBMS on HW Accelerators** [Jan 27]

Overview Programming Project

- **Team**
  - 1-3 person teams (w/ clearly separated responsibilities)
- **Task: SIGMOD'09 Programming Contest**

First Annual SIGMOD Programming Contest
Main Memory Transactional Index

http://db.csail.mit.edu/sigmod09contest/

  - Transactional, in-memory index for VARCHAR128, INT32, INT64 w/ duplicates
  - C test / performance suites, multi-threaded lookup/scan/insert/delete ops
  - Programming language: no restrictions, but **C or C++** recommended
- **Timeline**
  - **Oct 14:** Test drivers, reference implementation available
  - **Jan 20, 11.59pm:** Final programming project deadline

DEXTER Project

(Dresden Index for Transactional Access on Emerging Technologies)

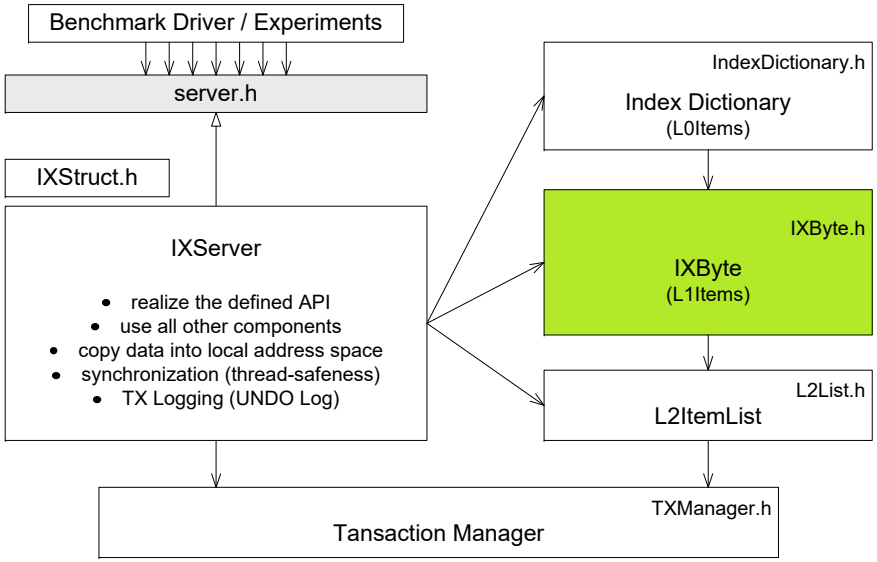
M. Boehm, B. Schlegel, P. B. Volk, U. Fischer, D. Habich, W. Lehner: Efficient In-Memory Indexing with Generalized Prefix Trees. **BTW 2011**
https://subs.emis.de/LNI/Proceedings/Proceedings180/227.pdf

Follow-up work:
Other index structures, mobile/HW accelerators, query processing on prefix trees

DEXTER Overview

First Annual SIGMOD Programming Contest 2009

- **System Architecture**
  - Transactional main memory index server
  - Manages a set of indices of different data types
  - Supports point and range queries as well as inserts and deletes
  - Optimistic concurrency control
  - TX UNDO log
  - Implemented in C



- **Several Subprojects**
  - **Core indexing**
  - Query processing (HPI Future SOC Lab project)
  - Mobile devices, GPUs

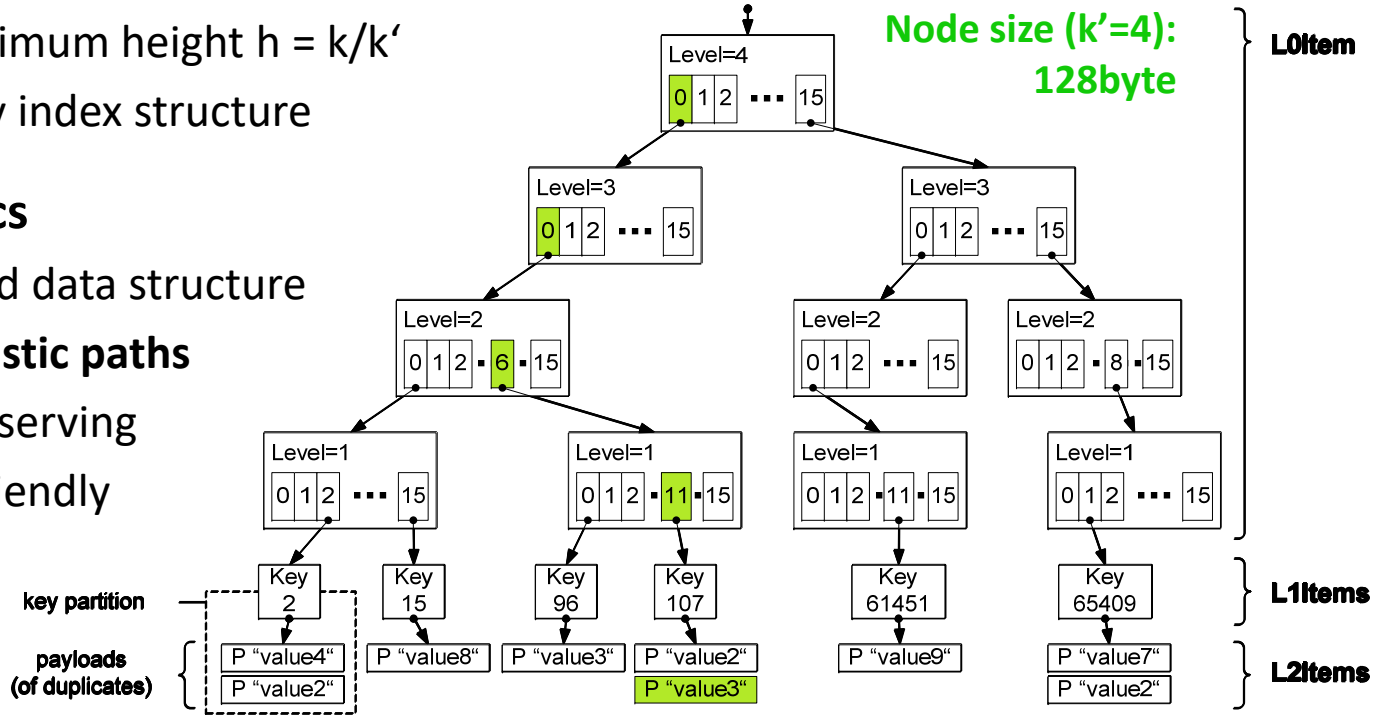
Core Index Structure

- **Generalized Prefix Tree (IXByte)**
  - **Arbitrary data types (byte sequences)**
  - **Variable prefix length k'**
  - Node size: s = $2^{k'}$ references
  - Fixed maximum height h = k/k'
  - Secondary index structure
- **Characteristics**
  - Partitioned data structure
  - **Deterministic paths**
  - Order-preserving
  - Update-friendly

INSERT key=107, payload="value3"

key = 107

0000	0000	0110	1011
0	0	6	11



Configuration Spectrum

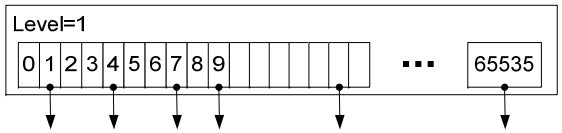
- **Definition (Generalized Trie)**
  - Variable prefix length k', fixed maximum height h = k/k'
  - Key observation: **read one pointer per node** (in contrast to balanced trees)

- **Spectrum**
  - Example k=16

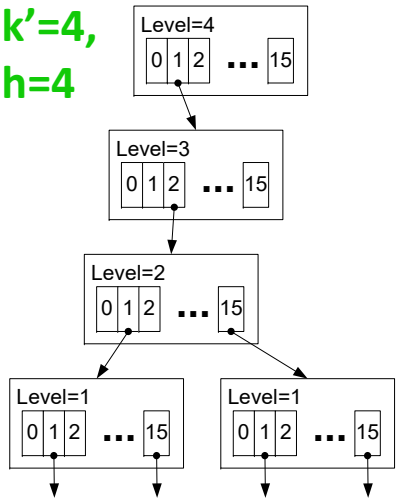
Trie Height ➔ # Memory Transfers

Space Requirements ➔ Memory Capacity / Allocation Time

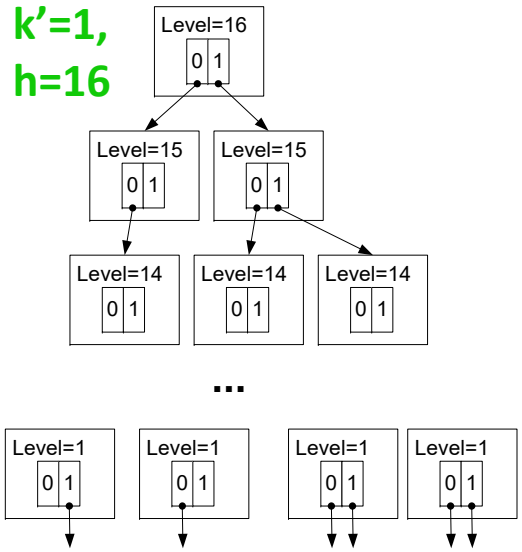
k'=16, h=1



k'=4, h=4



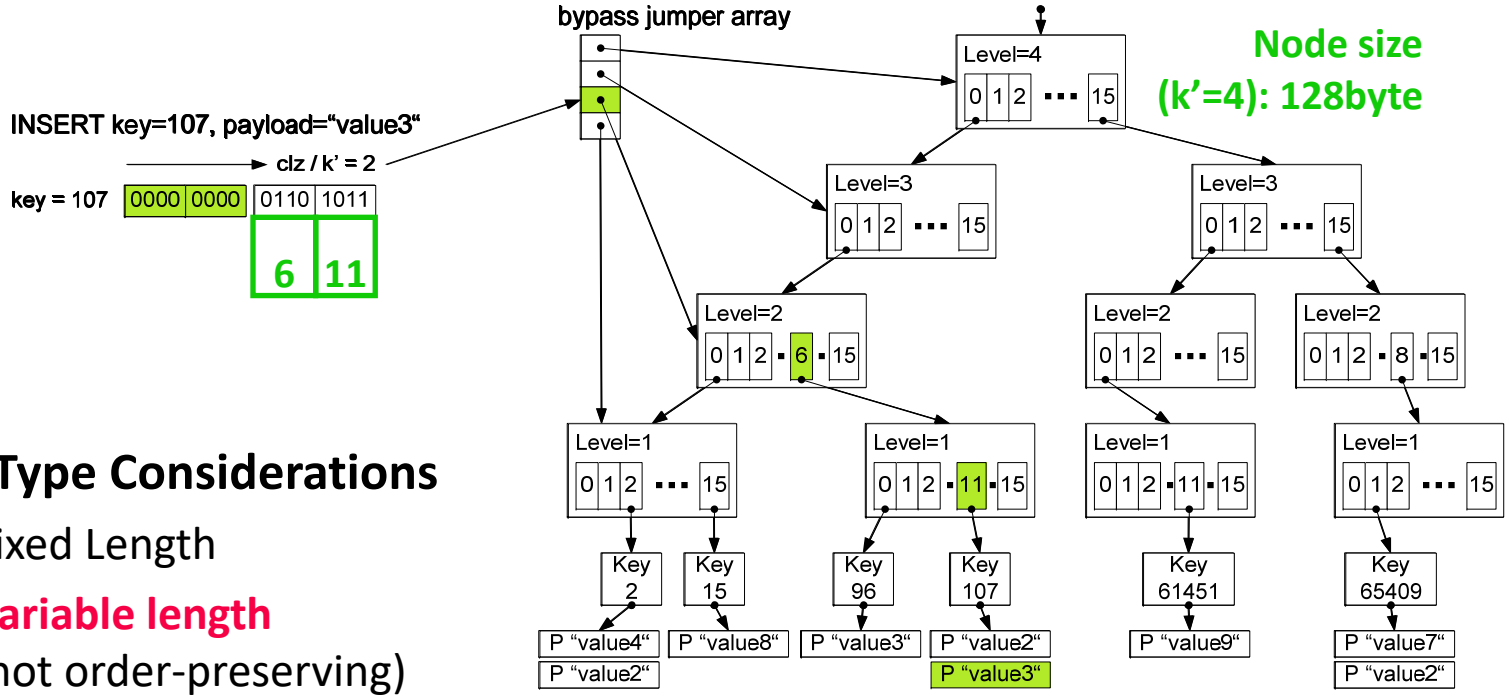
k'=1, h=16



- **Problems**
  - Trie Height
  - **Space Overhead**

Bypass Jumper Array

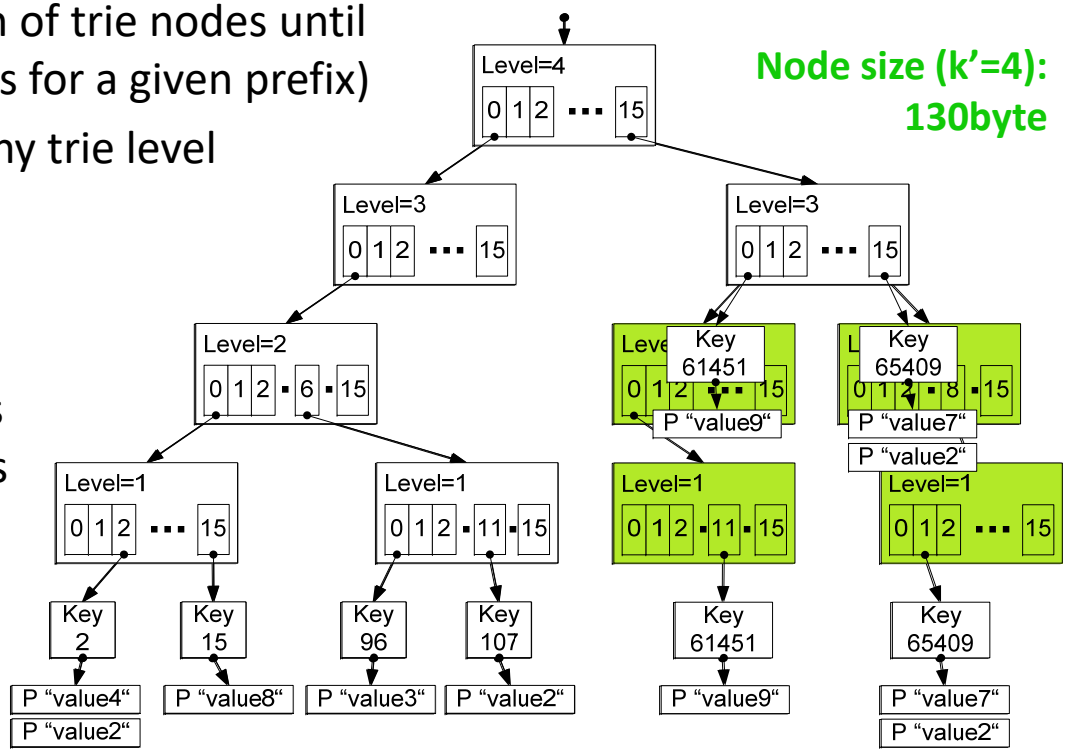
- **Core Concept**
  - Preallocate all direct 0-pointers (nodes)
  - Create a bypass jumper array of size h
  - Use the jumper array to bypass higher trie levels



- **Data Type Considerations**
  - Fixed Length
  - **Variable length**
    (not order-preserving)

Trie Expansion

- **Core Concept**
  - Defer access and creation of trie nodes until needed (two distinct keys for a given prefix)
  - References to tuples at any trie level
- **Dynamic Trie Expansion**
  - During Insert/Delete
  - **Algorithm:** Expand nodes until the two distinct keys exhibit a different prefix
  - Additional node flags required
- **Memory Optimization**
  - Preallocation, reduced pointers, node size alignment (64B cachelines)



Node size (k'=4): 130byte

Node size (k'=4): 64byte

Experimental Setting

- **Test Environment**
  - Quad core Intel Core-i7-920 processor (w/ HT) with 2.67 GHz, 6GB RAM
  - Fedora Core 14 (64 bit), GCC 4.4.5 (-O3 -fPIC -lpthread -combine)

- **Evaluation Parameters**
  - **Key data types:** INT32 (4 byte), INT64 (8 byte), VARCHAR (128byte)
  - **Operations:** Insert, Delete, Get, GetNext (Scan)
  - **Data sets:** key sequences (synthetical), uniform data (synthetical), real (DBLP)
  - **Index structures:** IXByte, B^+-Tree, T-Tree, HashMap
  - **Parameters:** Prefix length k‘ (default 4 bit), # Tuples N

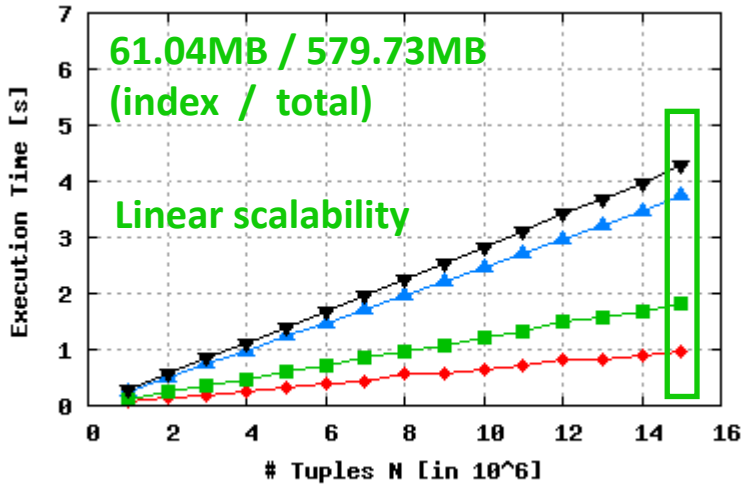
- **Experiments**
  - **Data:** Synthetically generated (S1, S2), real DBLP (R1, R2)
  - Comparison with B^+-Trees (C1, C2)
  - MIT main-memory benchmark (M1, M2, M3)

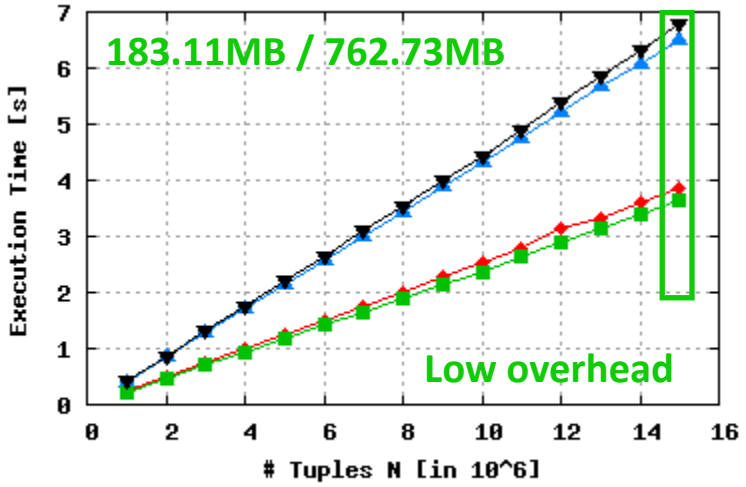
Synthetic Data

INT32, similar for INT64

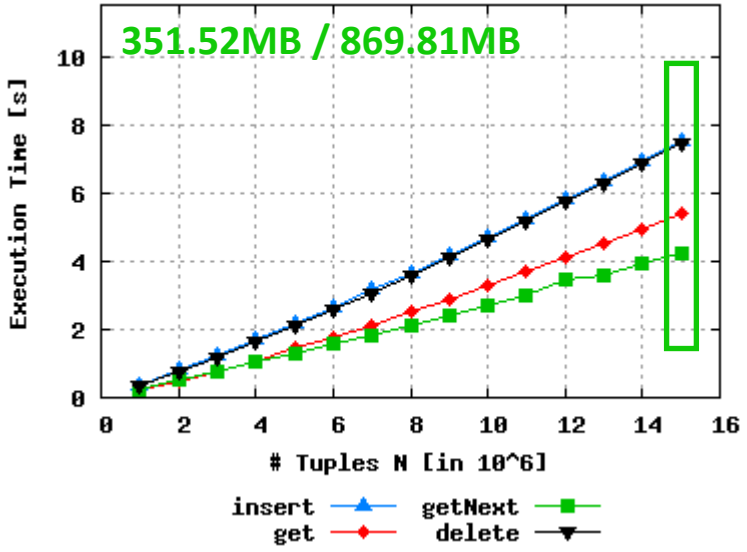
VARCHAR(128)

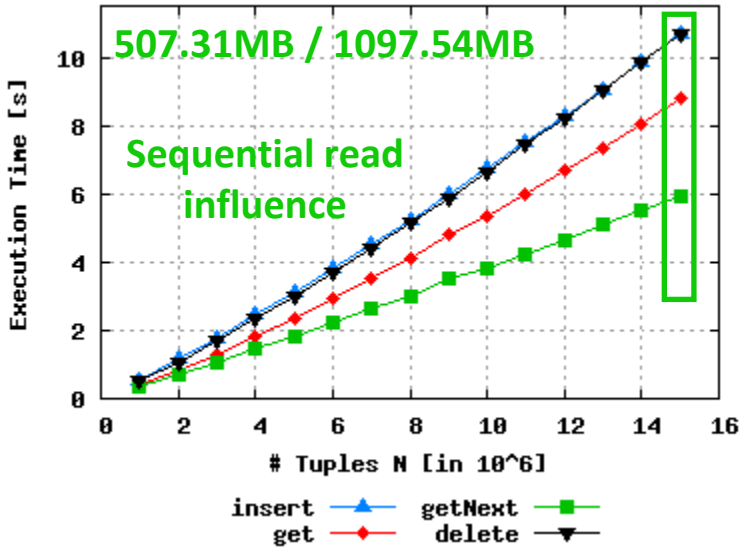
Sequence Data
(good case)





Uniform Data
(bad case)



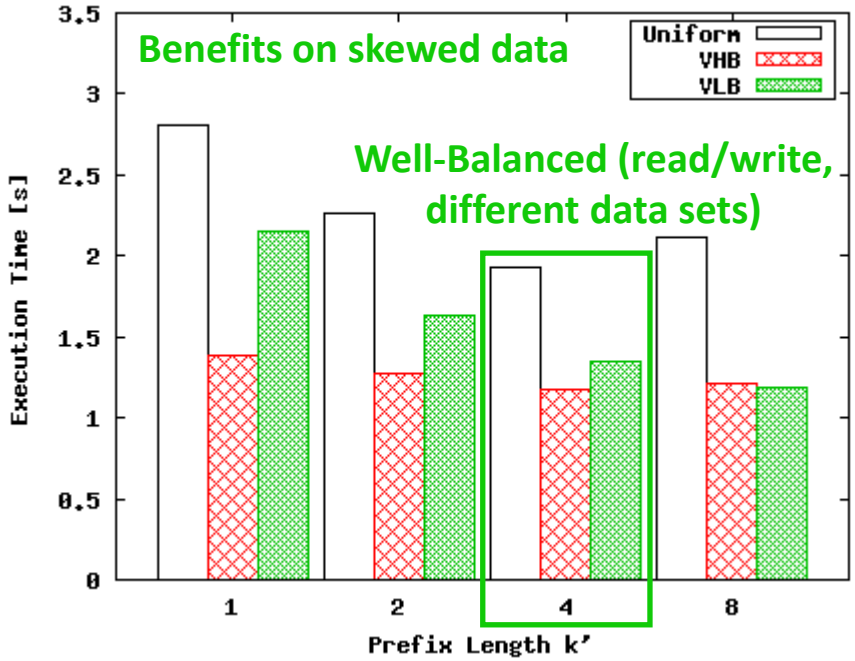


Varying Prefix Length k‘

[Elizabeth G. Reid: Design and Evaluation of a Benchmark for Main Memory Transaction Processing Systems. Master’s thesis, **MIT 2009**.]

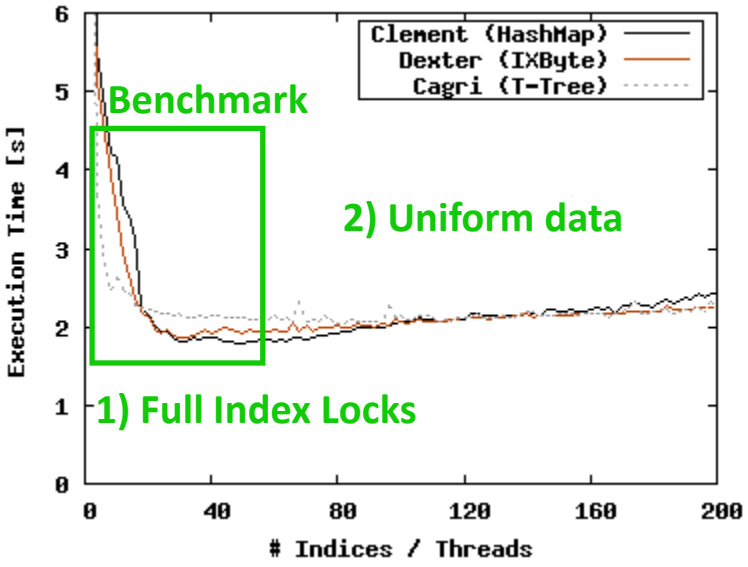
- **MIT Main-Memory Benchmark**
  - SIGMOD Programming Contest 2009
  - Transactional, multi-threaded main-memory index

- **Benchmark Workload**
  - 50 concurrent streams of transactions over multiple indices
  - Keys: INT32, INT64, VARCHAR
  - Payload: VARCHAR (10-100 byte)
  - Serializable TX execution
  - Standard: 800,000 TX, uniform
  - **(10%) Scan:** Get followed by 99-199 GetNext commands.
  - **(30%) Get:** 20 -30 Get operations.
  - **(60%) Update:** 5-10 (insert, delete) pairs



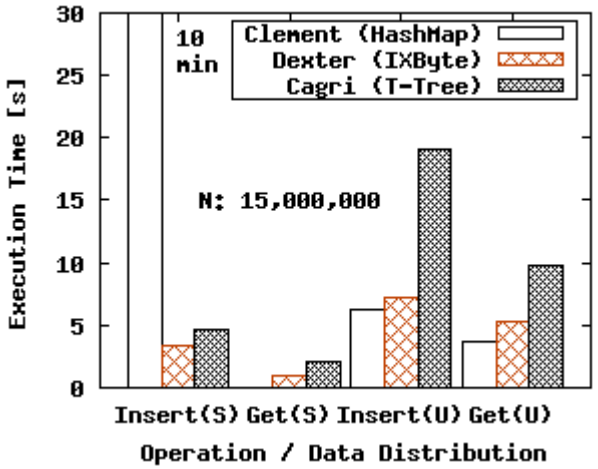
MIT Main-Memory Benchmark

- **Benchmark Comparison**
  - 5 finalists, 15 submissions, 90 users
  - **Final Ranking:** 1) Clement (HashMap), 2: Dexter (IXByte), 3: BCagri (T-Tree)
  - Other finalists: Ji (Trie/CSB-Tree), QBolec (Hash)
  - Other participants: Frame (CSS/CSB-Tree)



- **Individual Operations**

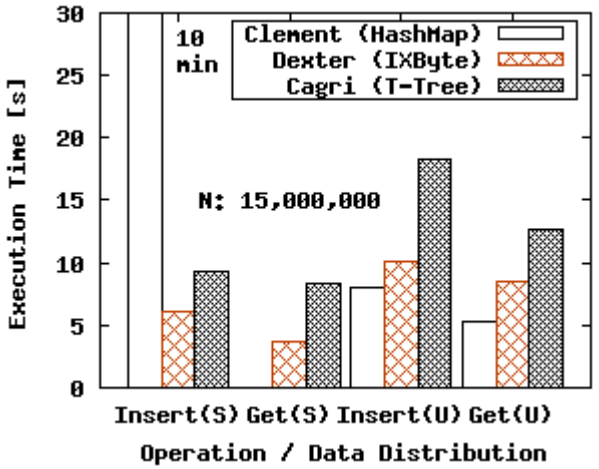
SHORT(4), similar for INTEGER(8)



Almost comparable to hash maps on uniform data

Especially beneficial on skewed data

VARCHAR(128)



Summary and Q&A

- **Course Goals**
  - **#1** Architecture and internals of traditional/modern DB systems
  - **#2** Understanding of DB characteristics → **better evaluation / usage**
  - **#3** Understanding of effective techniques → **build/extend DB systems**
    (these fundamental techniques are **broadly applicable** in other systems)
- **Programming Project**
  - **SIGMOD'09 Programming Contest** (in-memory, indexing, TX processing)
  - Transactional, in-memory index for VARCHAR128, INT32, INT64 w/ duplicates
  - Yearly contest, good preparation for next contest Spring'21
- **Next Lectures**
  - 02 **DB System Architectures** [Oct 14]
  - 03 **Data Layouts and Bufferpool Management** [Oct 21]
  - 04 **Index Structures and Partitioning** [Oct 28]
  - 05 **Compression Techniques** [Nov 04]