HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam


# Trends and Concepts in Software Industry I

# Goals

Deep technical understanding of trends and concepts in enterprise computing, esp. main-memory-centric data management on modern hardware and its impact on applications

- Foundations of database storage techniques and operators
- Characteristics of enterprise applications and systems
- Trends in enterprise computing (e.g., cloud platforms)
- Hands-on exercises and experiments

# Block Week

- General information
  - 8th of July to 11th (?) of July
  - Lectures given by Prof. Plattner
  - Discussions about open questions in enterprise computing are a vital part of the lecture!

- Focus areas
  - Basic principles of in-memory databases
  - Characteristics of modern enterprise systems
  - Advanced data structures for in-memory databases
  - Trends in enterprise computing

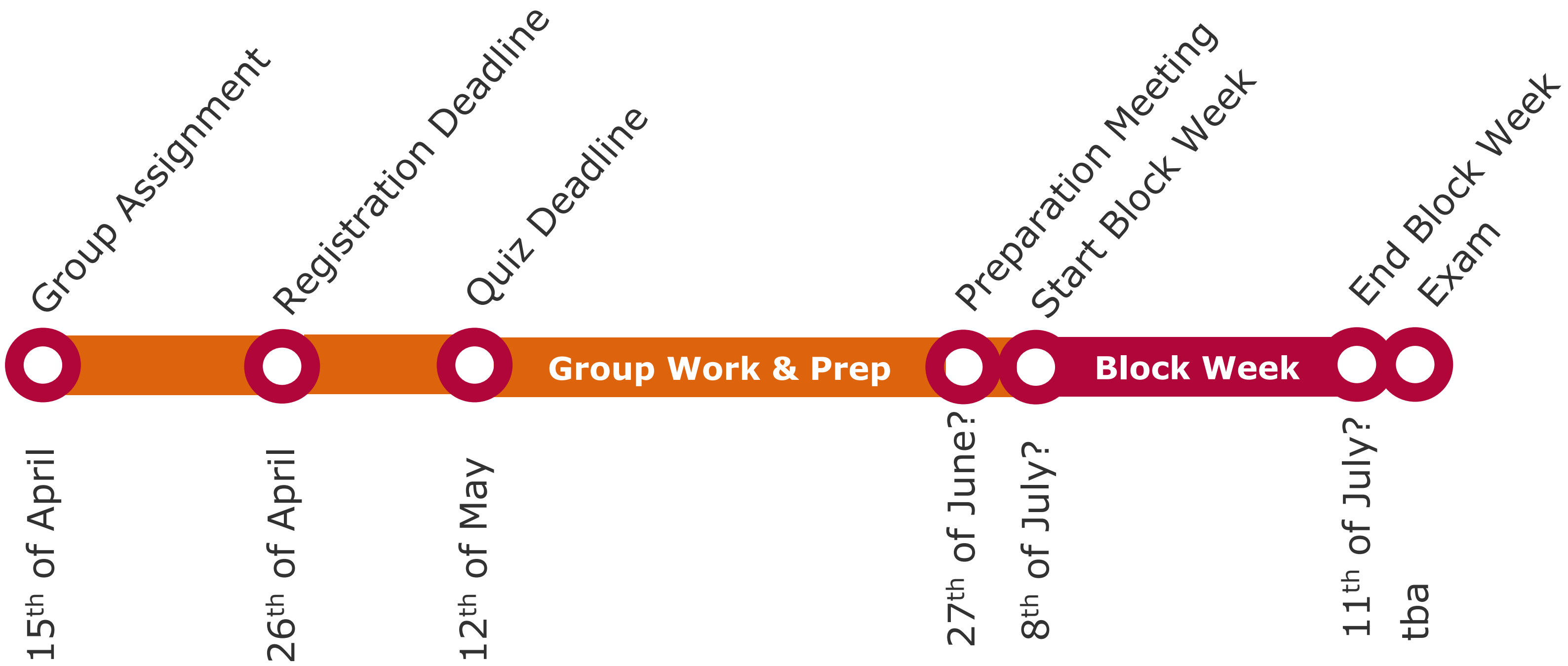
# General Information

- 6 ECTS points
- Latest enrollment: 26th of April 2019
- Modules
  - IT-Systems Engineering MA
    - ITSE-Analyse
    - ITSE-Entwurf
    - ITSE-Konstruktion
    - ITSE-Maintenance
    - BPET-Konzepte und Methoden
    - BPET-Spezialisierung
    - BPET-Techniken und Werkzeuge
    - SAMT-Konzepte und Methoden
    - SAMT-Spezialisierung
    - SAMT-Techniken und Werkzeuge
    - OSIS-Konzepte und Methoden
    - OSIS-Spezialisierung
    - OSIS-Techniken und Werkzeuge
  - Data Engineering MA
    - DATA-Konzepte und Methoden
    - DATA-Techniken und Werkzeuge
    - DATA-Spezialisierung
    - SCAL-Konzepte und Methode
    - SCAL-Techniken und Werkzeuge
    - SCAL-Spezialisierung

# Grading

- Final grade consists of
  - Preparation quiz (mandatory)
  - Group work, presentation, and participation during the block week (40%)
  - Written or oral exam, depending on #participants (60%)

# Schedule

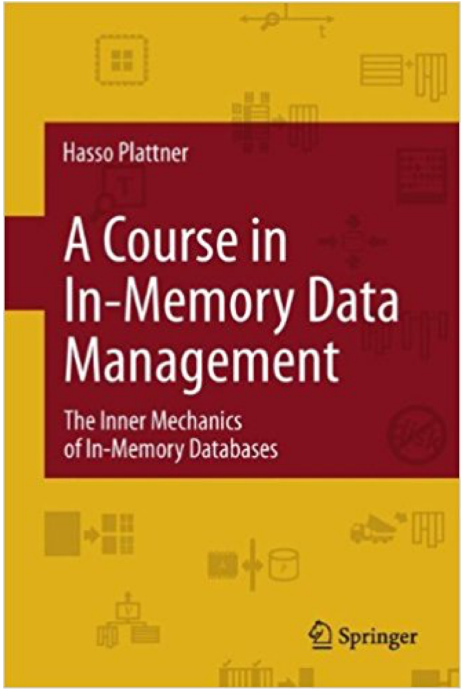

Group Assignment
15th of April
Registration Deadline
26th of April
Quiz Deadline
12th of May
Group Work & Prep
Preparation Meeting
27th of June?
Start Block Week
8th of July?
Block Week
End Block Week
11th of July?
Exam
tba

# Preparation Quiz

- Get a solid understanding of the fundamentals

- Materials
  - Course book (given out by Marilena Davis, V-2.11)
  - openHPI course
    https://open.hpi.de/courses/tuk2019

- Mandatory quiz
  - Start: 26th of April
  - Deadline: 12th of May



# Group Work

Hasso Plattner Institut

- Preparation of interactive group part
  - Teams of 6 to 8 students
  - Regular meetings
  - Team assignment: 15th of April
- Hands-on experiments
  - Familiarization with existing research
  - Implementation part in C/C++
  - Evaluation of the results
  - Presentation in the block week (~30 minutes)
- Tell us your topic preference: [https://forms.gle/iQMHLfUjkZfUN5Ab9](https://forms.gle/iQMHLfUjkZfUN5Ab9)

# Compression – Trading CPU cycles against I/O in main-memory column-oriented databases

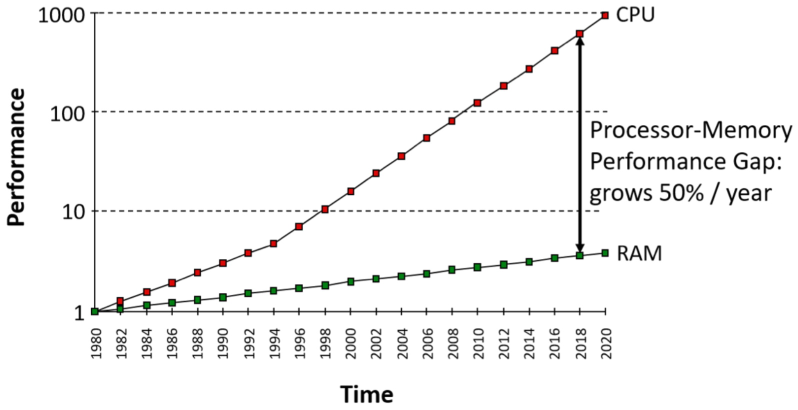
HPI Hasso Plattner Institut

**Motivation**

The main database bottleneck is memory bandwidth. And, the performance gap between the memory hierarchy and the CPU continues to grow. At the same time main memory is a scarce resource in today's systems. Fortunately, database compression can decrease utilized bandwidth as well as memory footprint.

**Experiments and Tasks**

- Get to know the characteristics of enterprise data
- Implement various compression techniques
- Measure the upfront cost of compression
- Measure the impact of decompression on query processing

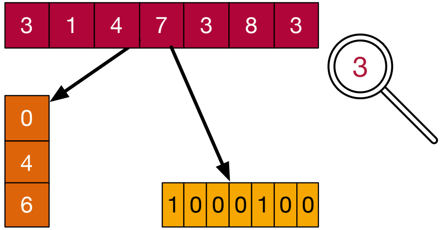


Rappoport and Yoaz: Computer Structure – Cache Memory

# Influence of CPU characteristics on query performance

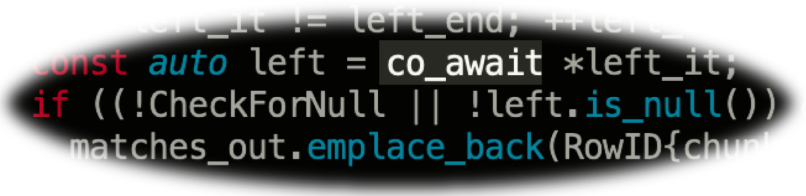
### Motivation

Besides clock frequency, DRAM latency and throughput, the query performance is influenced by more obscure CPU features, such as prefetching, branch prediction, and data dependencies. We will look into how careful programming affects the speed of database operations.




### Experiments and Tasks

- Implement a table scan on integers as an exemplary database operators
- Compare materialized result lists to binary hit masks
- Measure the influence of prefetching on sequential and random scans of data
- Add software prefetching and, if time permits, C++ coroutines to reduce the number of data stalls when scanning a filtered table



# Execution Models – Vectorized vs. Compiled Query Processing

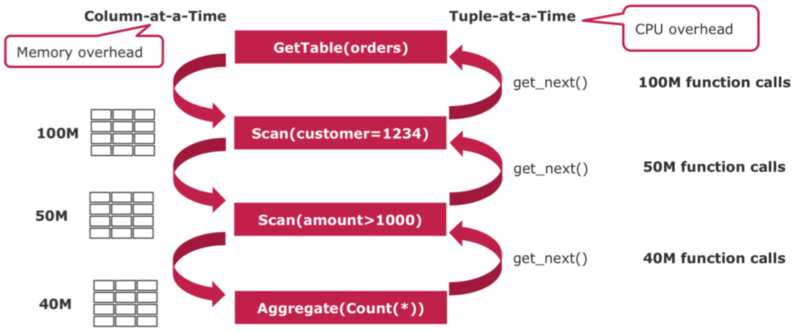
### Motivation

Traditionally, databases use one out of two execution models: Either operators execute bottom-up, producing intermediary result tables that are consumed by the following operator, or they work in a pull-based fashion and query the previous operator using iterators. The first concept is limited by the memory bandwidth when writing the intermediary results, the second is limited by the virtual method calls required by the iterators. Modern JIT-based approaches aim at building an iterator-based implementation without any dependence on virtual method calls.

### Experiments and Tasks

- Implement the three execution models for a selected number of database operators (focus on scans and aggregates).
- Compare the performance of your implemented approaches using well known queries from industry standard benchmarks (focus on the effects of execution models).
- Discuss how well the execution models work for other database operators (When does which model excel? What are their limits?).

```
SELECT COUNT(*) FROM orders WHERE customer = 1234 AND amount > 1000
```




**JIT Pipeline:**
GetTable(orders)
Scan(customer=1234)
Scan(amount>1000)
Aggregate(Count(*))

# Indexes – Are there still use cases for main-memory column stores?

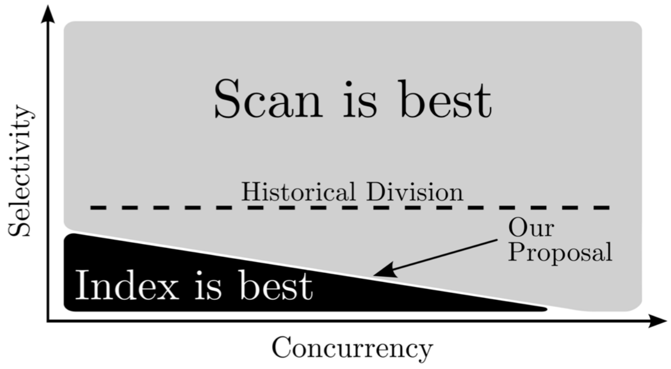


## Motivation

- Sequential scans are becoming faster and faster because of many-core systems, SIMD, and further hardware optimizations
- Alternatively, auxiliary data structures, e.g., indexes can be used
- In theory index scans have a complexity of $\mathcal{O}(\log n)$ vs. $\mathcal{O}(n)$ for sequential scans
- We want to evaluate whether index scans are still advantageous over highly-optimized in-memory scans in certain scenarios

## Experiments and Tasks

- Implement an optimized, efficient scan operator
- Utilize a third party index data structure to implement an efficient index scan
- Compare the performance taking several dimensions, e.g., data types, selectivity, and concurrency into account to identify the break-even point of both techniques




Kester et al.: Access Path Selection in Main-Memory Optimized Data Systems: Should I Scan or Should I Probe? (SIGMOD 2017)

# Contacts


- Guenter Hesse
  - Room: V-2.03
  - Email: guenter.hesse@hpi.de
  - Phone: 1381

- Ralf Teusner
  - Room: V-2.01
  - Email: ralf.teusner@hpi.de
  - Phone: 1301


Tell us your topic preference:

https://forms.gle/iQMHLfUjkZfUN5Ab9