

Develop your own Database 2018/2019

Week 1

Outlook

1. High-Level Overview
2. First Work Package
3. Organizational Stuff

What can you expect?

- Better understand how in-memory databases work
- Learn how to familiarize yourself with a larger code base
- Gain experience in system development
- Improve your C++ skills
- Work in small teams on a larger project

If this sounds interesting to you, you are in the right room.

Timeline

Weekly Meetings

1. Sprint

Simple Table Functionality (Segments, Chunks, Types)

2. Sprint

Dictionary Compression

3. Sprint

First Operators (Scan, Sort, ...)

Group Phase

It's all about performance

- Optimizer Rules
- Self-Tuning
- and more...

Final Presentation

Timeline

- In addition to introducing you to the architecture, the first two sprints aim at
 - refreshing your C++ knowledge
 - getting you up to speed with our code style, test setup, and expectations
- If you and C++ are on a first-name basis, this might appear a bit slow - please bear with us

What do we expect?

- Fruitful discussions about why we do things the way we do
- Active participation in the group work and our meetings

What do we hope for?

1. Generate interest in our research
2. Continue to work with you in Master's theses, Hiwi jobs, ...

If anyone is interested right away, please contact us.

Who are we?



Martin Boissier

- Data Aging and Tiering
- Pricing



Stefan Klauck

- Replication
- SSI-CLOPS



Markus Dreseler

- New Hardware
 - NVRAM
 - SGI



Jan Kossmann

- Self-Driving Databases
- Query Optimization

What has changed since last year?

- Hyrise has grown significantly and can slowly be considered a real database
 - Just as in industry, you will have to work your way into a grown (but well maintained) code base
 - We will help you by proposing group projects that are digestible chunks

What has changed since last year?

- In the first year, we built fundamental components like a scheduler or joins
- In the second year, we built more components, including networking and subselects
- Now, we can focus on performance and build optimizer rules, or automatic tuning components
 - You will be able to see your results right away!

What has changed since last year?

- We have incorporated the in-class and EvaP feedback and
 - will not have an intermediary presentation
 - will start the group project earlier
 - will coordinate the communication between seminar students and other people working on Hyrise better

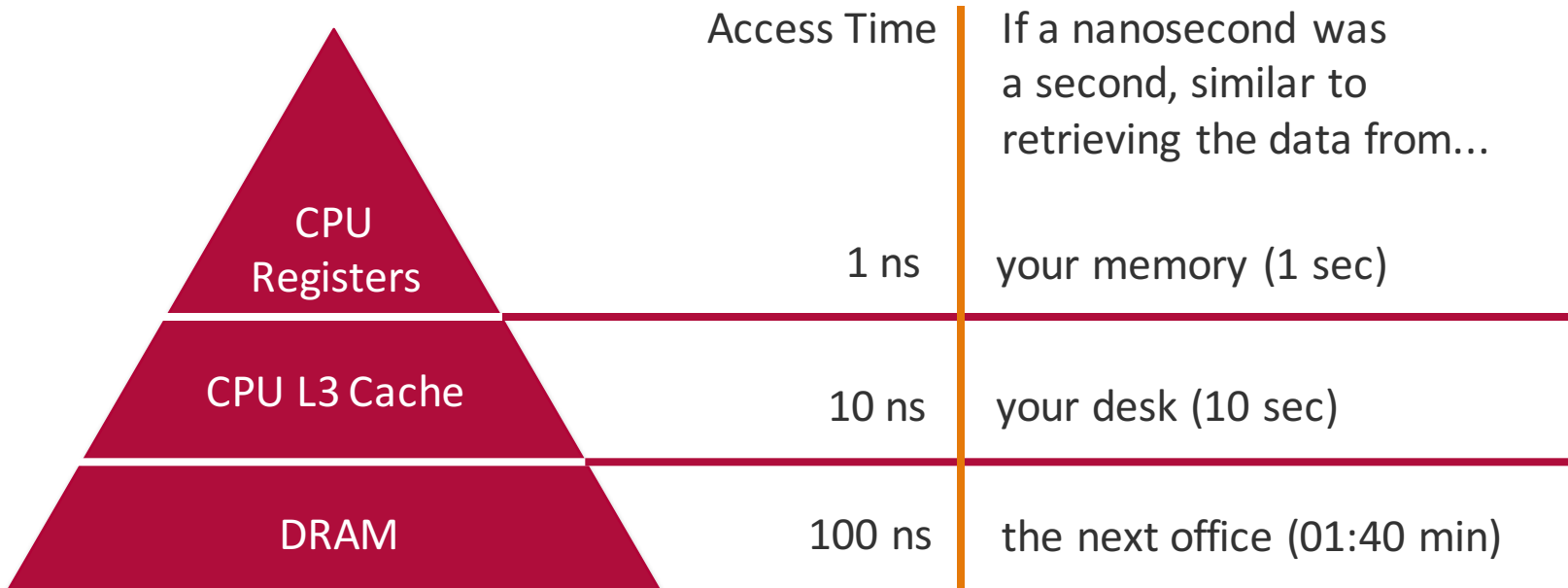
Introducing Opossum



Introducing Opossum

- Opossum is the (1) prototypical, (2) columnar (3) in-memory database that we will build during the first three sprints
- Prototypical: We do not plan for Opossum to be used in a productive environment
- Columnar: We exclusively use columnar orientation for data
- In-Memory: All data that we work with is stored in RAM

Why In-Memory?

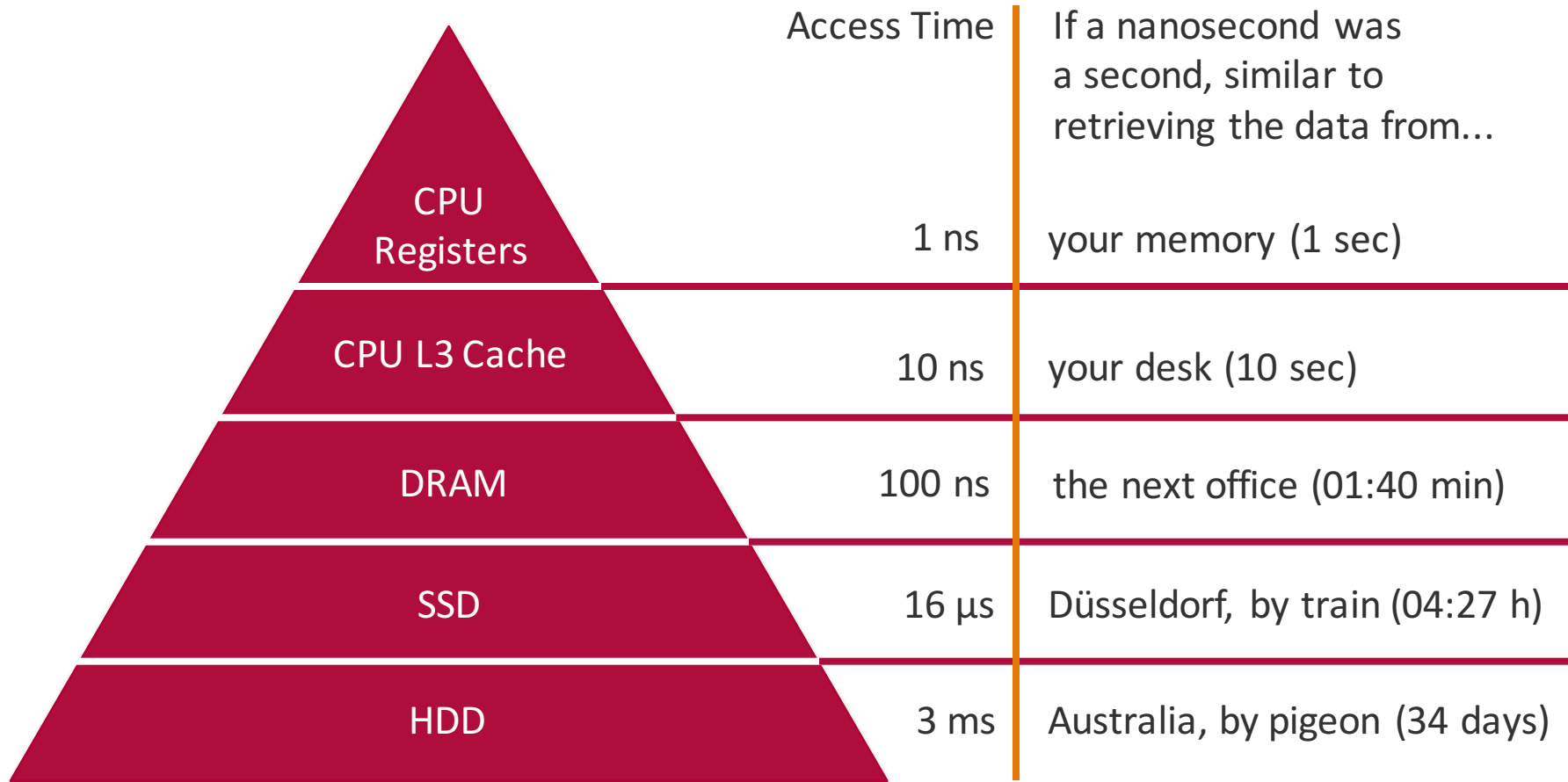


	Access Time	If a nanosecond was a second, similar to retrieving the data from...
CPU Registers	1 ns	your memory (1 sec)
CPU L3 Cache	10 ns	your desk (10 sec)
DRAM	100 ns	the next office (01:40 min)

Why In-Memory?

	Access Time	If a nanosecond was a second, similar to retrieving the data from...
CPU Registers	1 ns	your memory (1 sec)
CPU L3 Cache	10 ns	your desk (10 sec)
DRAM	100 ns	the next office (01:40 min)
SSD	16 μ s	Düsseldorf, by train (04:27 h)

Why In-Memory?



Build your own Database – Week 1

First Work Package

Description

- You can find the description of the work package online:
 - <https://hpi.de/plattner/teaching/winter-term-201819/develop-your-own-database-ws18.html>

First tasks

1. Set up your build environment
2. Implement a single segment
3. Group segments into a chunk
4. Append data to a chunk
5. Group chunks into a table
6. Store tables in a StorageManager

Setting up your Environment

- Demo (git clone, install, cmake, make test -j)

Up-to-Date Build Setup

- Why do we require current compiler and library versions?
- First reason: New C++17 features are great, but building up technical debt for workarounds is not:

```
-#if __has_include(<optional>)
-#include <optional>
-#else
-#include <experimental/optional>
-#endif

-#if __has_include(<nullopt>)
-#include <nullopt>
-#else
-#include <experimental/nullopt>
-#endif

-#if __has_include(<optional>)
-#include <optional>
-#else
-#include <experimental/optional>
-#endif

-#if __has_include(<nullopt>)
-#include <nullopt>
-#else
-#include <experimental/nullopt>
-#endif
```

Up-to-Date Build Setup

- Second reason: Even compilers are not infallible

Bug 79180 - Nested lambda-capture causes segfault for parameter pack

Status: RESOLVED FIXED

Alias: None

Product: gcc

Component: c++ ([show other bugs](#))

Version: 6.3.0

Importance: P3 normal

Target Milestone: 8.0

Assignee: Not yet assigned to anyone

Reported: 2017-01-22 08:25 UTC by Markus Dreseler

Modified: 2017-10-02 12:48 UTC ([History](#))

CC List: 5 users ([show](#))

See Also:

Host:

Target:

Build:

Known to work:

Known to fail:

Last reconfirmed: 2017-01-23 00:00:00

Up-to-Date Build Setup

- Now that we have gcc 8...

Bug 86740 - [8/9 Regression] ICE with hana and nested lambdas (likely a regression, tsubst_copy, at cp/pt.c:15325) ([edit](#))

Save Changes

Status: NEW ([edit](#))

Alias: None ([edit](#))

Product: gcc

Component: C++ ([show other bugs](#))

Version: 8.0

Reported: 2018-07-30 14:26 UTC by [Markus Dreseler](#)

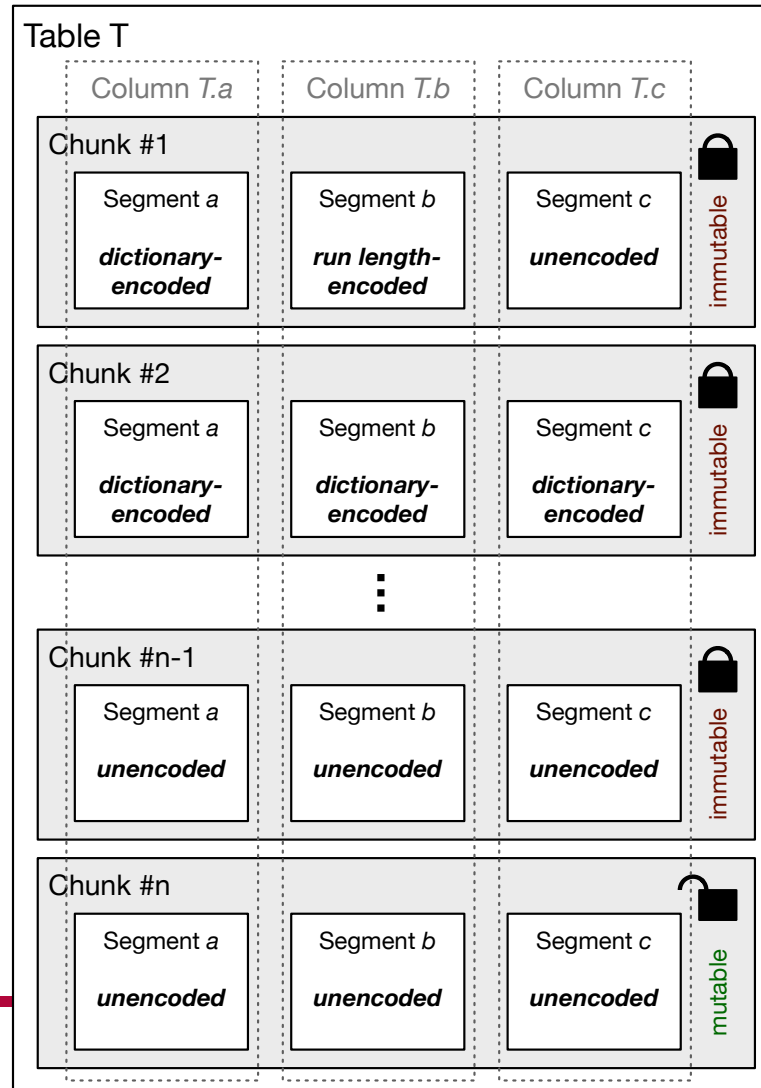
Modified: 2018-10-09 09:13 UTC ([History](#))

CC List: 6 users including you ([edit](#))

Ignore Bug Mail: (never email me about this bug)

See Also: ([add](#))

The Opossum Table Model



Document Walkthrough

Build your own Database – Week 1

Organizational Stuff

About Correctness

- For the sprints, we are using a stripped Hyrise code base
- Some things look slightly different in the master, but we believe that this is a better start
- We have tested that everything works the way we expect it to, but this does not mean that everything is perfect
- If something looks wrong, or if you have any issues about the course itself, please do not hesitate to talk to us

Einschreibung und -fristen, Leistungserfassungsprozess, Vertiefungsgebieteinordnung

Allgemeine Information

- > Semesterwochenstunden: 4
- > ECTS: 6
- > Benotet: Ja
- > Einschreibefrist: 21.10.2018
- > Lehrform: Projekt / Seminar
- > Belegungsart: Wahlpflichtmodul
- > Maximale Teilnehmerzahl: 24

^ IT-Systems Engineering MA

- > ITSE-Analyse
- > ITSE-Entwurf
- > ITSE-Konstruktion
- > ITSE-Maintenance
- > BPET-Konzepte und Methoden
- > BPET-Spezialisierung
- > BPET-Techniken und Werkzeuge
- > OSIS-Konzepte und Methoden
- > OSIS-Spezialisierung
- > OSIS-Techniken und Werkzeuge
- > SAMT-Konzepte und Methoden
- > SAMT-Spezialisierung
- > SAMT-Techniken und Werkzeuge

^ Data Engineering MA

- > PREP-Konzepte und Methoden
- > PREP-Techniken und Werkzeuge
- > PREP-Spezialisierung
- > SCAL-Konzepte und Methode
- > SCAL-Techniken und Werkzeuge
- > SCAL-Spezialisierung

Einschreibung und -fristen, Leistungserfassungsprozess, Vertiefungsgebieteinordnung

Kriterium	Gewichtung
Sprint 1-3	30 %
Gruppenphase	60 %
Aktive Mitarbeit	10 %

Piazza

- Most likely, there will be remaining questions about the architecture or the implementation
- Waiting for a week is not an option
- Your classmates may have the same question or be able to help you

Piazza

- We use Piazza to answer questions, communicate, and organize the class:
- <https://piazza.com/hpi.uni-potsdam.de/fall2018/dyod>
- Please use common sense in how much of your implementation you should share

Groups

- We would like for you to work in groups of three
- Feel free to start working on the first sprint now
- Please wait with forming groups until you have received your confirmation by the Studienreferat (Monday?)
- You can also use Piazza to find team members
- For your submission, please send us an email with the names of your group members, a link to your repository, and the SHA-1 hash of your final commit

Weekly Meetings

- We will use one of our two slots for presentations given (mostly) by the teaching team – attendance here is highly recommended
- The other slot can be used for your group work
- As needed, we might use it for further clarifications of the material – this will be announced in advance and is optional
- Which slot do you prefer? Vote on Piazza!

Deliverables

- 31 Oct Code Sprint 1
 - 7 Nov Review Sprint 1
-
- 14 Nov Code Sprint 2 (tbc)
 - 21 Nov Review Sprint 2
 - 28 Nov Code Sprint 3
 - 5 Dec Review Sprint 3
- (Group phase)
- 6 Feb First Code Group Phase
 - tbd Review and Final Code Group Phase

Next Week

- Deep Dive into some of the used C++ concepts and beyond
 - Templates
 - Smart Pointers
 - RAII