HPI Hasso Plattner Institut
Digital Engineering · Universität Potsdam

WEEK 1

DYOD 2023

AGENDA

- **Course Overview**
- First Work Package
- Administration

WHAT CAN YOU EXPECT?

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

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

TIMELINE

WEEKLY MEETINGS

Sprint 1: Simple table functionality (segments, chunks, types)

Sprint 2: Dictionary encoding

Sprint 3: First operators (scan, print, …)

Sprint 4: Familiarization with Hyrise – write a test case

Project Phase

Work on more complex concepts, e.g.,

- performance optimization for multi-socket systems,
- integrating data profiling into query optimization,
- efficient parallelization of database operators, …

Project 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?

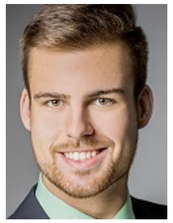
1. That you learn a lot about IMDBs & system's development
2. Generate interest in our research
3. Continue to work with you in master's theses, student assistant jobs, …

   If anyone is interested right away, please contact us.

WHO ARE WE?

Daniel Lindner

Query Optimization using Data Dependencies



Marcel Weisgut

Memory Disaggregation for Database Systems

Martin Boissier

Memory Footprint Reduction of In-Memory Database Systems

Thomas Bodner

Elastic Query Processing on Serverless Infrastructure

Stefan Halfpap

ILP-based Heuristics for Data Placement and Selecting Indexes

Prof. Felix Naumann

Professor of Information Systems Group

Prof. Tilmann Rabl

Professor of Data Engineering Systems Group

INTRODUCING OPOSSUM

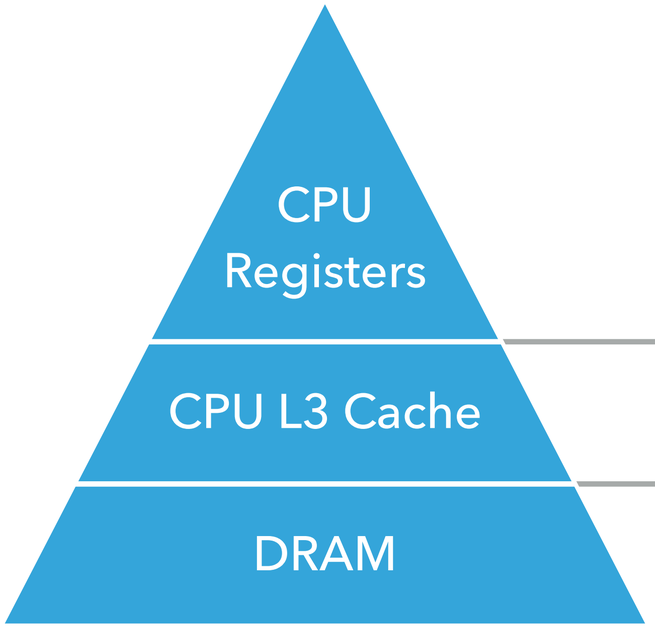
https://commons.wikimedia.org/wiki/File:Fam%C3%ADlia_de_Gamb%C3%A1s-de-orelha-branca_(Didelphis_albiventris).jpg

INTRODUCING OPOSSUM

- Opossum is the (i) prototypical, (ii) columnar (iii) 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

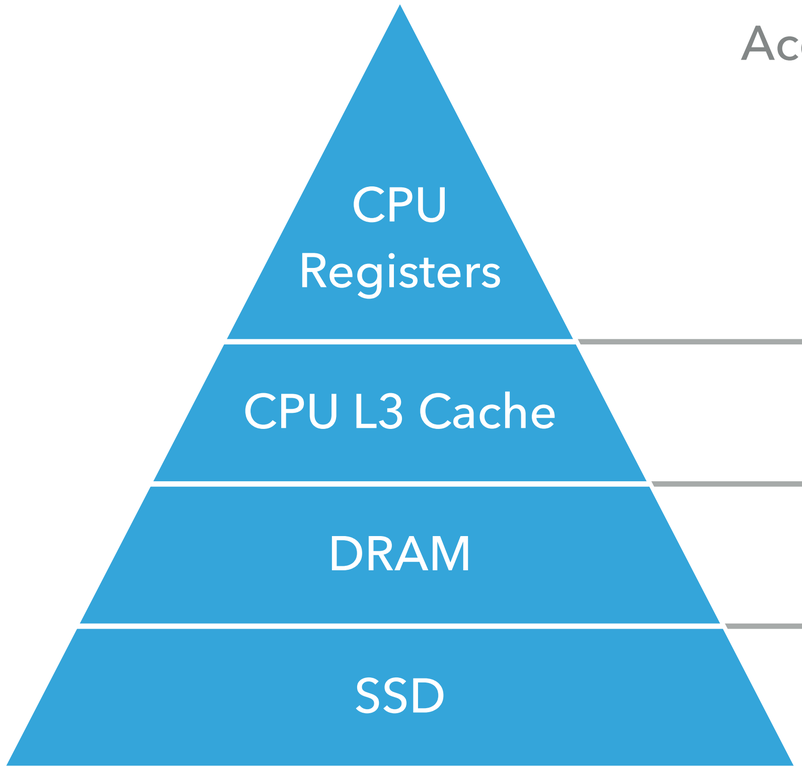
WHY IN-MEMORY?



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

http://people.eecs.berkeley.edu/~rcs/research/interactive_latency.html
http://www.montana.edu/cpa/news/wwwpb-archives/yuth/pigeon.html

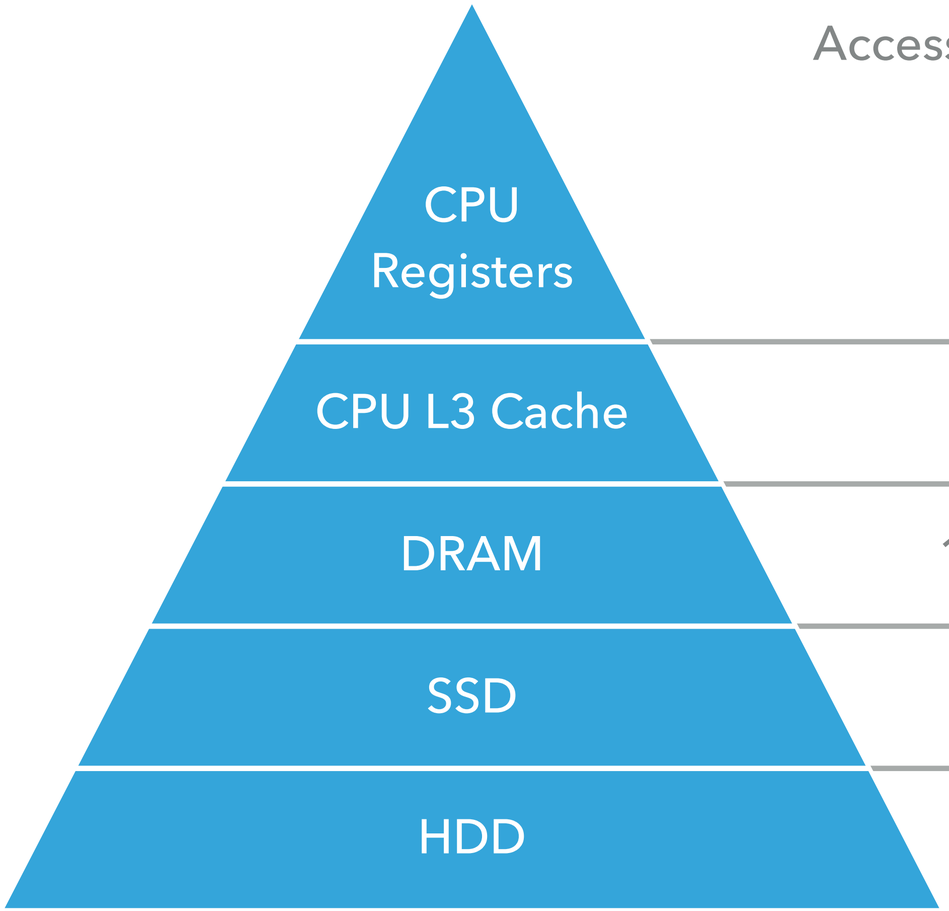
WHY IN-MEMORY?



	Access Time	If a nanosecond was a second, similar to retrieving the data from…
CPU Registers	1 ns	your memory (1 s)
CPU L3 Cache	10 ns	your desk (10 s)
DRAM	100 ns	the next office (1 min 40 s)
SSD	16 µs	Düsseldorf, by train (4 h 27 min)

http://people.eecs.berkeley.edu/~rcs/research/interactive_latency.html
http://www.montana.edu/cpa/news/wwwpb-archives/yuth/pigeon.html

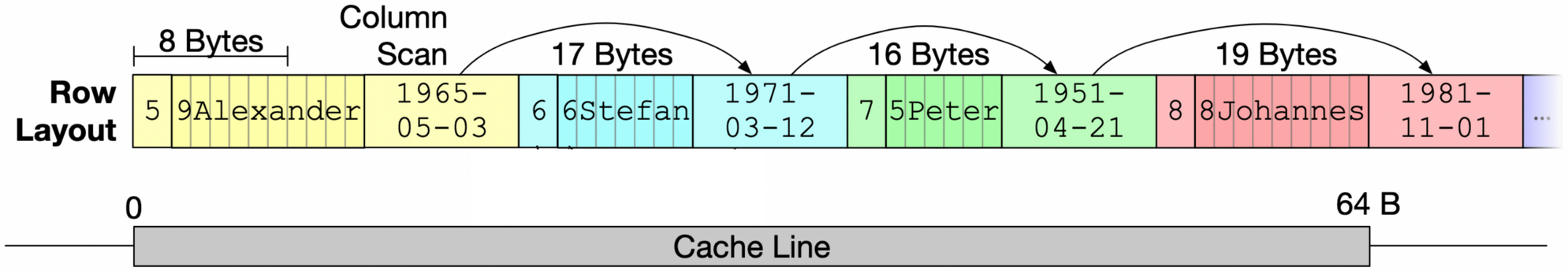
WHY IN-MEMORY?



	Access Time	If a nanosecond was a second, similar to retrieving the data from…
CPU Registers	1 ns	your memory (1 s)
CPU L3 Cache	10 ns	your desk (10 s)
DRAM	100 ns	the next office (1 min 40 s)
SSD	16 µs	Düsseldorf, by train (4 h 27 min)
HDD	3 ms	Australia, by pigeon (34 days)

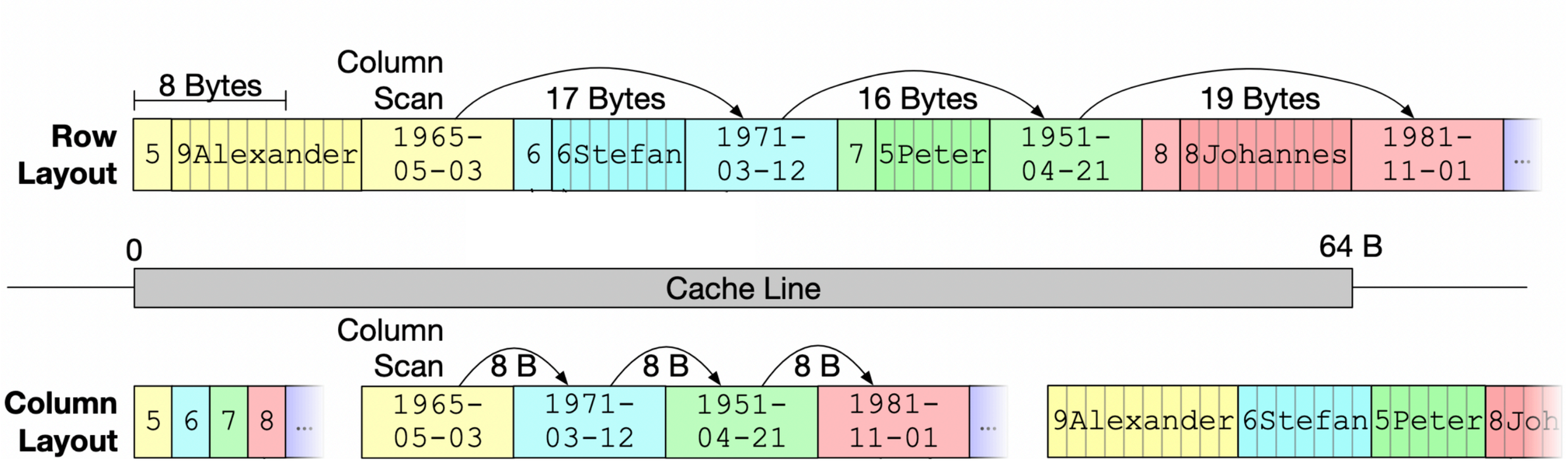
http://people.eecs.berkeley.edu/~rcs/research/interactive_latency.html
http://www.montana.edu/cpa/news/wwwpb-archives/yuth/pigeon.html

WHY COLUMNAR?



Markus Dreseler: *Automatic Tiering for In-Memory Database Systems*. University of Potsdam, Germany, 2022 (Ph.D. thesis).

WHY COLUMNAR?



Markus Dreseler: *Automatic Tiering for In-Memory Database Systems*. University of Potsdam, Germany, 2022 (Ph.D. thesis).

WHY WRITE OUR OWN DATABASE AT ALL?

- For research, we need a database system that has reasonable performance and is easy to modify
- Leaving out things like authentication and error handling makes the database leaner, thus easier to understand and maintain
- Re-building an academic system typically takes seconds, with a commercial database it comes close to an hour

WHY WRITE OUR OWN DATABASE AT ALL?

- We focus on the things we need, for example, fast benchmarking.

Item	Latency (ms/iter) old	Latency (ms/iter) new	Change	Throughput (iter/s) old	Throughput (iter/s) new	Change	p-value
TPC-H 01	5761.08	5656.80	-2%	0.17	0.18	+2%	0.4381
TPC-H 02	47.45	48.75	+3%^	21.07	20.51	-3%^	0.0334
TPC-H 03	2045.09	2262.73	+11%	0.49	0.44	-10%	0.0000
TPC-H 04	1538.91	1447.82	-6%	0.65	0.69	+6%	0.0000
TPC-H 05	3145.70	3134.69	-0%	0.32	0.32	+0%	0.7119
TPC-H 06	163.09	156.63	-4%^	6.13	6.38	+4%^	0.0000
TPC-H 07	989.69	1002.52	+1%	1.01	1.00	-1%	0.0010
TPC-H 08	786.71	722.89	-8%	1.27	1.38	+9%	0.0000
TPC-H 09	5743.37	5323.22	-7%	0.17	0.19	+8%	0.0000
TPC-H 10	4856.85	2985.67	-39%	0.21	0.33	+63%	0.0000
TPC-H 11	95.06	78.29	-18%^	10.52	12.77	+21%^	0.0000
TPC-H 12	1007.95	993.16	-1%	0.99	1.01	+1%	0.0000
TPC-H 13	4671.15	4732.37	+1%	0.21	0.21	-1%	0.0149
TPC-H 14	427.40	430.29	+1%^	2.34	2.32	-1%^	0.0178
TPC-H 15	197.99	197.50	-0%^	5.05	5.06	+0%^	0.0348
TPC-H 16	635.70	640.27	+1%	1.57	1.56	-1%	0.0001
TPC-H 17	223.52	224.82	+1%^	4.47	4.45	-1%^	0.0000
TPC-H 18	1556.13	1671.13	+7%	0.64	0.60	-7%	0.0000
TPC-H 19	278.04	277.39	-0%^	3.60	3.60	+0%^	0.6359
TPC-H 20	416.96	419.59	+1%^	2.40	2.38	-1%^	0.2252
TPC-H 21	4838.83	4883.32	+1%	0.21	0.20	-1%	0.2279
TPC-H 22	428.20	428.99	+0%^	2.34	2.33	-0%^	0.2577
Sum	39854.84	37718.84	-5%				
Geomean						+3%	
Notes	^ Execution stopped due to max runs reached						

STATUS QUO – HYRISE

- 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
- Compared to commercial databases, our query latency is within 5x; sometimes, we are actually faster

https://git.io/hyrise

AGENDA

- Course Overview
- **First Work Package**
- Administration

DESCRIPTION

- You can find the description of the work package at Moodle:

Enrolment key:
`namespace opossum`

https://moodle.hpi.de/course/view.php?id=429

FIRST TASKS

1. Form groups of three students **until Apr. 26, 23:59**
2. Set up your build environment
3. Implement a single segment
4. Group segments into a chunk
5. Append data to a chunk
6. Group chunks into a table
7. Store tables in a storage manager

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++ features are great, but building up technical debt for workarounds is not:

```
-#if __has_include(<optional>)
-template <class T>
-using optional = ::std::optional<T>;
-static auto nullopt = ::std::nullopt;
-#else
-template <class T>
-using optional = ::std::experimental::optional<T>;
-static auto nullopt = ::std::experimental::nullopt;
-#endif
```

UP-TO-DATE BUILD SETUP

- Second reason: Even compilers are not infallible

llvm / llvm-project Public (contains the clang compiler code)

[C++20] [Modules] Compilation failure std module without magic comment clang:modules

#62112 by mordante was closed 4 days ago

`clang-format` oscillates formatting of `// clang-format off` clang-format

#62107 by Svetlitski-FB was closed 3 days ago

CXXUnresolvedConstructExpr has invalid source locations and causes Clang coverage crash when visiting it. clang:frontend confirmed coverage crash-on-valid

#62105 by ZequanWu was closed 3 days ago

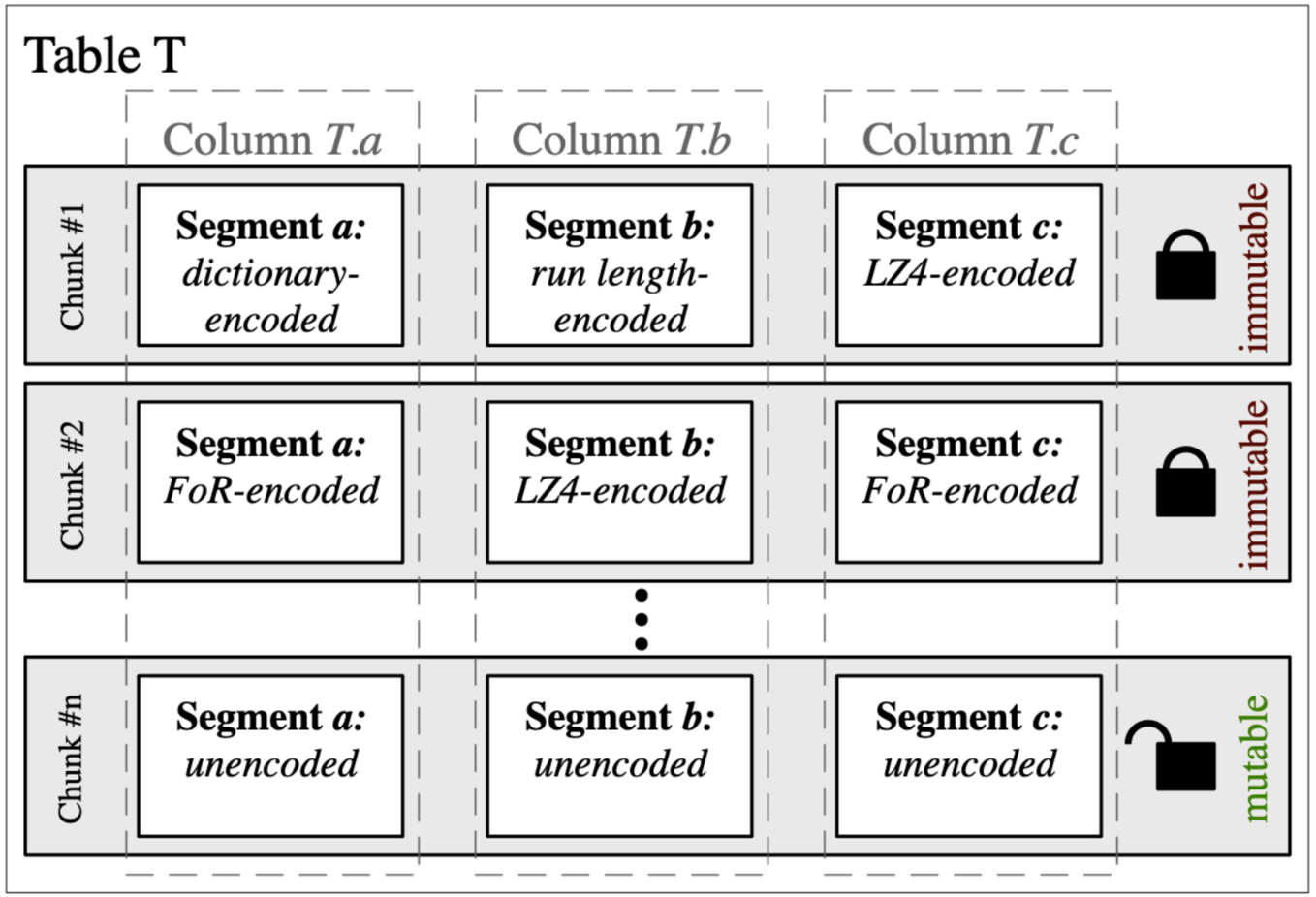
[clang-tidy] Crash in clang::ASTContext::getTypeInfo after initializing an array with a value of unknown type clang-tidy crash

#62097 by alexhenrie was closed 4 days ago

Missed optimization for `*(ptr << 2)` backend:X86 llvm:optimizations missed-optimization

#62093 by WaffleLapkin was closed 2 hours ago

THE OPOSSUM TABLE MODEL



Martin Boissier: *Robust and Budget-Constrained Encoding Configurations for In-Memory Database Systems*. PVLDB 15(4) (2021).

DOCUMENT WALKTHROUGH

Develop your own Database
The Opossum Blueprint

SoSe 2023 :: Sprint 1

Overview

In this first sprint, you will implement Opossum's basic storage classes, i.e., segments, chunks, and tables. We provide some code that will help you with this and test cases that you can use to verify your implementation.

Preliminary Information

This first project serves two purposes: First, it allows you to get a better idea of what this seminar will be about. Second, it should give you an idea of the level of C++ programming that we will be expecting in this class. The discussed concepts will be challenging for some students who have not worked with C++ for a while. If you manage to get through this sprint, you should be able to learn more advanced C++. Once we have built the foundation for our database, we will focus more and more on database architectures and concepts.

We would like you to work on the projects in groups. Remember that this project is a part of the *Leistungserfassungsprozess*. Discussing abstract concepts with other students is ok, sharing (parts of) an implementation is not. Please use a GitHub repository for your development.

ABOUT CORRECTNESS

- For the sprints, we are using a stripped Hyrise code base
- Some things look slightly different in the main repository, 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

AGENDA

- Course Overview
- First Work Package
- **Administration**

ENROLMENT, DEADLINES, DELIVERABLES, MODULES

General information

- Lecturers: Daniel Lindner, Marcel Weisgut, Martin Boissier, Stefan Halfpap, Thomas Bodner, Prof. Felix Naumann, Prof. Tilmann Rabl
- Weekly hours: 4 (6 ECTS)
- Weekly slot: Thursday, 9.15–10.45, L-1.06
- Enrolment deadline: **please enrol and form groups until April 26, 2023**
- Programs: IT Systems Engineering MA, Data Engineering MA
- Project Seminar, Compulsory Elective Module
- Moodle: https://moodle.hpi.de/course/view.php?id=429 (enrolment key: `namespace opossum`)

Modules

- ITSE MA: BPET, OSIS
- DE MA: DASY

Deliverables

- Programming tasks (80 %)
  - Sprint implementations
  - Code review of other group's sprint implementations
  - Project Implementation
  - Code review of other group's project implementation
- Further components (20 %)
  - Project presentation
  - Active participation in seminar meetings

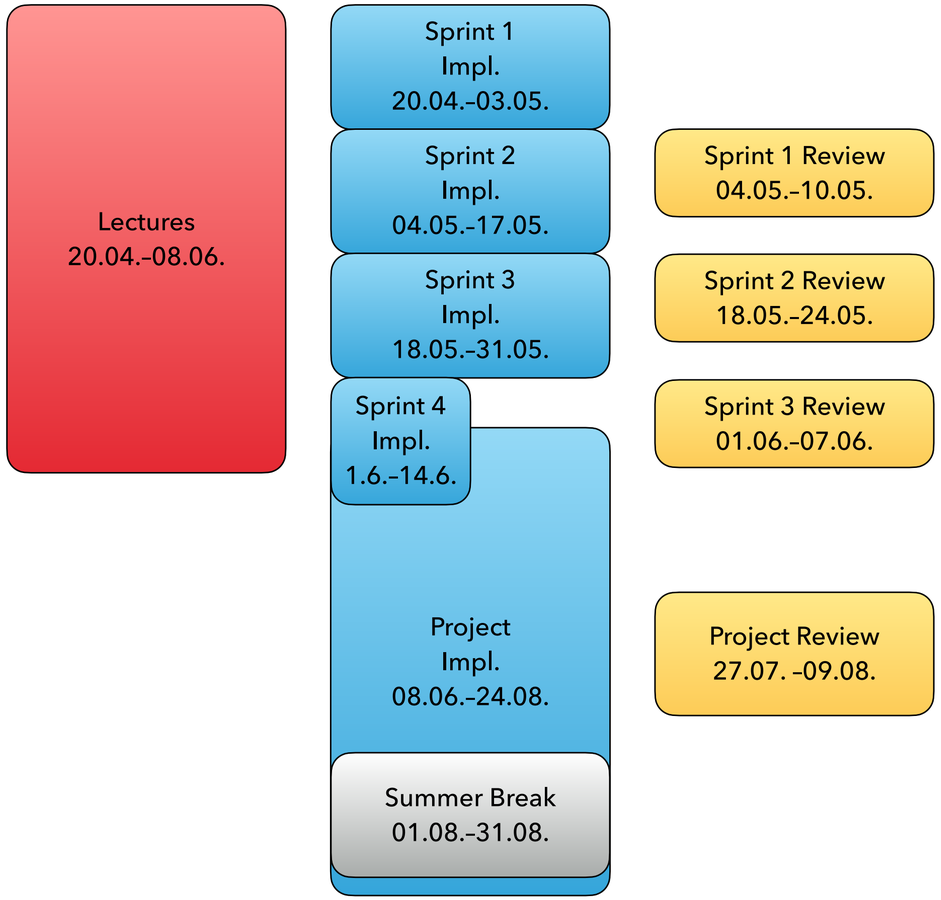
- Criteria for programming tasks are, besides functionality:
  - Code quality, performance, test coverage

MOODLE

- We use Moodle to organize the class and communicate: https://moodle.hpi.de/course/view.php?id=429
- Assignments, e.g., sprints, are to be submitted via Moodle
- Questions and answers may be discussed here, please use common sense in how much of your implementation you should share

DELIVERABLES & PHASES

- 03 May — Sprint 1 Code
- 10 May — Sprint 1 Review
- 17 May — Sprint 2 Code
- 24 May — Sprint 2 Review
- 31 May — Sprint 3 Code
- 07 June — Sprint 3 Review
- 14 June — Sprint 4 Code
- 26 July — First Project Code (Beta)
- 03 August — Project Presentation (does this fit you?)
- 09 August — Project Review
- 24 August — Final Project Code



NEXT WEEK

- Deep dive into first C++ concepts
  - Templates
  - Smart Pointers
  - Resource Acquisition is Initialization (RAII)