How building an industry DBMS differs from building a research one

Markus Dreseler @ CIDR
Background

- 2015 - 2021: PhD student at the chair of Prof. Plattner @ HPI Potsdam
  - Built **Hyrise**, an open-source in-memory research DBMS*
- Since April 2021: Senior Software Engineer at **Snowflake** in our Berlin office

- This talk is based on my personal experience
- It is neither representative nor comprehensive

*) Dreseler et al., Hyrise Re-engineered: An Extensible Database System for Research in Relational In-Memory Data Management (EDBT 2019) - https://github.com/hyrise/hyrise
Snowflake builds a DBMS, right?

- When I was looking for jobs, I thought that Snowflake built a cloud DBMS.
But at least I know SQL…

**MATCH_RECOGNIZE**
Recognizes matches of a pattern in a set of input, and returns all matches.
The pattern is defined in a set of regular expressions.

**ARRAY_UNION_AGG**
Returns an ARRAY that contains the union of the distinct values from the input ARRAY's in a column.

**CONNECT BY**
Joins a table to itself to process hierarchical data in the table.
Similarities
Databases really work like they teach in university

Everything is more complex in an industry database, but the SQL core is similar:

*) Logical / Physical Query Plan
Still doing R&D and still writing papers

Instant feedback and discussions
Workloads and Optimizations
Relevance of optimizations

- In research, incremental improvements rarely cause much excitement.
- In industry, if we make every query 1ms faster, we save a year worth of compute a day.
  - Optimizations do not have to be fancy.
- **Finding the right thing** to work on:
  - Customer requests
  - 15+ PB of metadata in Snowhouse
  - Background tooling across the entire fleet.
- But: Having 5% of queries regress is only fine in research.
Reliability
Testing and Validation

Local Testing
- Unit Tests
- End-to-End Regression Tests
- Sanitizer Builds
- Static Code Analysis
- Query Permutation Testing
- Re-executing production queries*
- Up- and downgrade testing

Merge Gates

Main Builds (every 4h)

Release Testing

We had this in Hyrise

This was new for me

*) Yan et al., Snowtrail: Testing with Production Queries on a Cloud Database, DBTest ’18
Parameter protection (aka. feature flags)

- Used to **guard new code paths**:

```c
603  m_missSel->fillWithSubtractedVectors(*probeSel, *selection);
609  if (InitParams::getParamBool(PR_ENABLE_FIX_360233)) {
610    m_missSel->fillWithSubtractedSubsequenceVectors(*probeSel, *selection);
611    } else {
612    m_missSel->fillWithSubtractedVectors(*probeSel, *selection);
613    }
```

- Multi-level - can be enabled for individual queries
- This allows us to have a **single binary** while still supporting
  - Running code in test environments first
  - Private and public previews
  - Behavior changes
Edge Cases and Resilience

- At **2,400,000,000 queries a day**, everything you can think of happens
  - Race conditions - if it is in the code, it will happen
  - ECC RAM failing / Bit Rot
  - Cloud providers being out of instances
  - Cloud providers giving us bad instances

A researcher could just restart the experiment
Wrapping up
What do I miss?

- Being able to know each part of the code
- Hacking up a performance improvement between a lecture and lunch
- Not worrying about regressions

What is great?

- Code is used billion times a day
- Data-driven development (Snowhouse)
- Support Rotation is stressful, but allows you to have an immediate impact on blocked customers
- Working alongside hundreds of engineers with different expertise
How can you be part of this?

Try Snowflake

You can try Snowflake for 30 days and spend $400 on us:

https://signup.snowflake.com/

Chat with us at CIDR

Marcin Zukowski

Berni Schiefer

Florian Funke

Jan Kossmann

Markus Dreseler

Join us

● Engineering offices in Berlin, San Mateo, Warsaw, Toronto, Bellevue