The Case for In-Process Analytics

Hannes Mühleisen & Mark Raasveldt
Architecture Timeline

Dumb Terminal
Mainframe

/ 1974

1984
\ Client/Server
Sybase

1994

In-Process
BerkeleyDB

/ 2000
\ SQLite
Reflections on Client/Server for OLAP
ML is not going to move into DB. Even if we wish it very hard.
ML is not going to move into DB. Even if we wish it very hard.
Conclusion: Integrate not try to absorb
Client Server ruins DB/ML marriage
Can’t transfer serious data amounts

Perception
Client Server ruins DB/ML marriage
Can’t transfer serious data amounts

Reality
Conclusion: Client/Server very problematic

If DB is bottleneck in ML pipeline, it is removed
In-Process Integration

- Zero-Copy
- No Server Management
- Easy Installation
- Script Portability
- Function Pointer UDFs
In-Process can be tricky

- Random Hardware Quality
- Self-Checking required
- Can’t crash, would take host down
- Can’t use mmap, signal handlers, locale, errno etc.
- Strings…

That's no moon!
DuckDB

- In-process OLAP DBMS, written in C++11
- Full SQL support
- No external dependencies
- APIs for C, C++, CLI, Python, R, Java, Node.JS, …
- Extensively tested
- MIT License

www.duckdb.org

100,000 Downloads/week
Yes I am mad

Demo?!
2014: We demo Zero-Copy DB <> R Integration (using “memory rewiring” avant la lettre)

2016: Gabe Becker proposes R ALTREP (Lazy vectors) at DSC

2018: ALTREP released in R 3.5.0

2021: DuckDB releases ALTREP for Strings
Conclusion

• OLAP systems are better in-process

• New challenges!
  • In-Process cooperation
  • Hardware second-guessing
  • Bulk Transactions
  • Gracious Out-of-core

@hfmuehleisen
duckdb.org