Hochschule Aalen

Hybrid Transactional and Analytical Processing Databases: A Systematic Literature Review

Daniel Hieber, Gregor Grambow
Dept. of Computer Science - Aalen University
Presented by: Daniel Hieber
daniel.hieber@hs-aalen.de

Daniel Hieber

- Aalen University
  - since 04.2020 research assistant
  - since 03.2020 master student - machine learning and data analytics
  - 09.2016 - 02.2020 bachelor student - software engineering
- Work Experience
  - 09.2018 - 02.2020 inovex GmbH bachelor student
  - 03.2019 - 09.2020 inovex GmbH working student - application development
  - 09.2018 - 02.2019 inovex GmbH intern - application development
- Research Interests
  - communication in distributed systems
  - databases, big data, data analysis
  - digitalization of emergency services
  - medical informatics

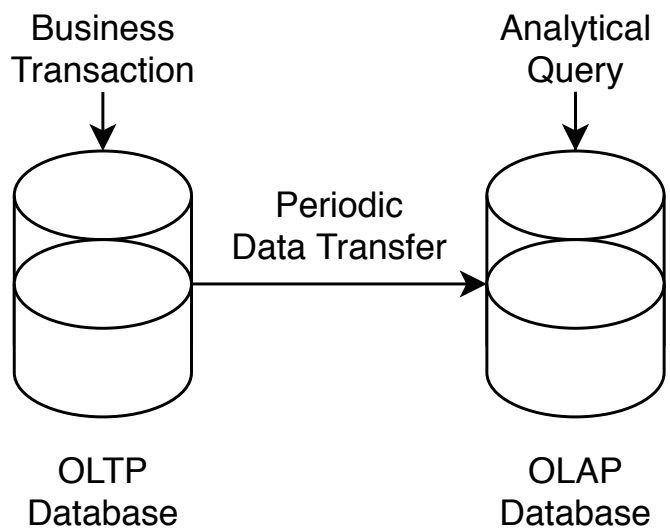
Table of Contents

Literature Review Methodology

- following Kitchenham's systematic review procedure
- utilizing Google Scholar
- search strings:
  - htap "data warehouse" OR OLTP OLAP
  - HTAP OR OLAP OLTP
  - hybrid transactional analytical processing
- search returned 583 papers to refine further
- summarized 41 papers

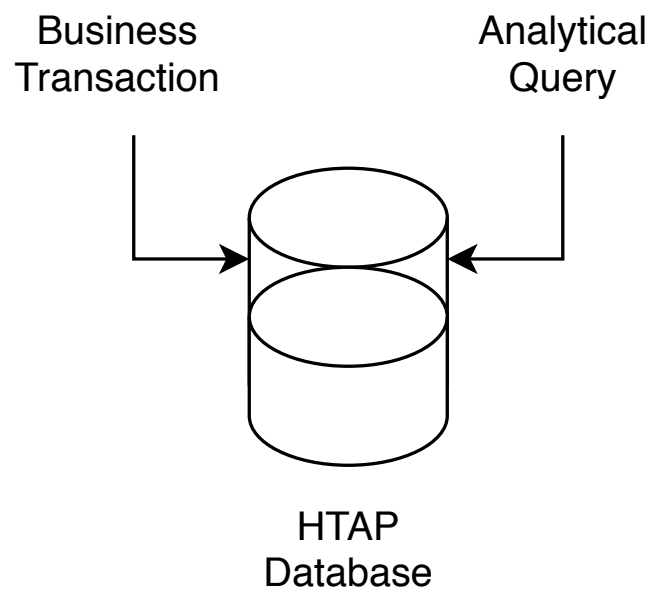
Problem and Current/Old State

- 2 databases technologies for efficient work in big companies
  - OLTP databases for transactional processing
  - OLAP databases for analytical processing
- increased total cost of ownership (TCO)
- increased maintenance
- analytical queries run on old data



The Solution: HTAP Databases

- Hybrid Transactional Analytical Processing
- OTLP and OLAP in one database
- decreasing TCO and maintenance
- enabling analytical queries on fresh data
- research started around 2010
- already many systems in use
- e.g. HyPer, Hyrise, SAP HANA, Wildfire, SnappyData, MemSQL

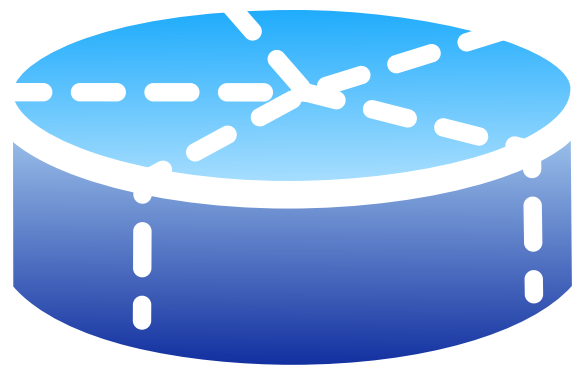


Fundamental Architecture

- Scaling
  - scale-up on single server
    - all researched systems of this category rely on main memory
    - Non-Uniform Memory Data Access - NUMA
    - e.g. HyPer, Hyrise
  - scale-out to multi server clusters
    - OLTP on one or multiple servers
    - e.g. SnappyData, SAP HANA
- Data and Table Structure
  - most systems use column stores (e.g. Wildfire, Casper)
  - some also support row but prefer column (HyPer, Hyrise)
  - some systems are hybrid (e.g. Janus, PostgreSQL)

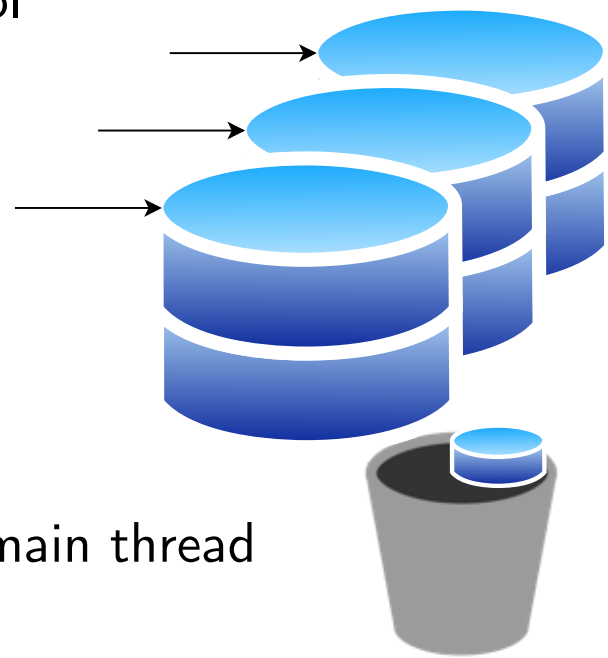
Fundamental Architecture

- Saving and Partitioning Data
  - scale-up systems need to compress data or move it to other storage
  - HyPer and Hyrise implement own data structures to handle data efficiently
  - SmartLTM writes data to files on hard drive
  - right partitioning technique can enable databases to skip up to 90% of data in searches



Concurrency and Garbage Collection

- Concurrency
  - Multi Version Concurrency Control
  - Snapshotting
    - fork system call
    - virtual memory snapshot
    - copy on write
- Garbage Collection
  - passive - e.g. periodically in the background
  - active - e.g. every transaction in main thread
  - eager pruning or save pruning



Query Handling

- compiling/executing queries as LLVM assembler code
- executing queries on one or multiple servers in parallel
- dynamically schedule memory to OLTP/OLAP workloads as required
- Query Language
  - default SQL supported by most
  - often extended scripting languages with SQL basis
  - some other specific implementations

Big Data on HTAP Databases

- Wildfire
  - Big Data as main use-case
  - heavily distributed
- SnappyData
  - streaming of big workloads
  - heavily distributed
- SAP HANA data platform
  - big data platform on top of SAP HANA
- HyPerInsight
  - data exploration
  - reduces time required to get used to new data sets
- ...

Recovery and Logging

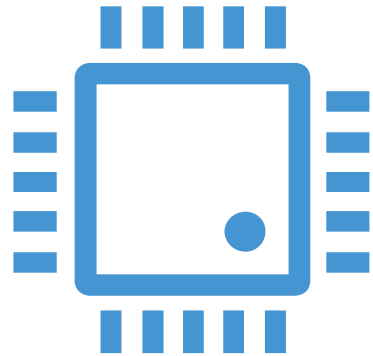
- logging data from volatile memory is very important
- recovery requires persistent storage
- different solutions:
  - save (transaction) logs on persistent storage
  - save data periodically on persistent storage
  - on failure load data from storage and replicate from logs
  - replicate data to multiple servers
  - load data from replication server on failure

Stream Processing

- streaming data as special OLTP use-case
- extensions with other software is possible (SAP HANA)
- allow more specific queries than stream processors
- main memory databases are still generally inferior to stream processors
- increasing research
- e.g. SnappyData, AIM

Future

- HHTAP - Heterogenous HTAP
  - utilizing GPU and CPU
  - shared memory for GPU und CPU
- streaming workloads
- optimization
  - less new database development
  - focus on improving existing systems



Conclusion

- HTAP databases combine OTLP and OLAP in one database
- data can be analysed in realtime
- scale-out and scale-up systems
- many different approaches
  - main memory / shared memory
  - concurrency handling
  - query handling
  - data storage and table structures
  - indizes
- the future research and development focuses on
  - HHTAP
  - streaming workloads
  - optimization
- free and open source solutions are available