In-Memory Analytic DBMSs: Design and Lessons Learned

Pedro Eugenio Rocha Pedreira

Facebook Inc.
1 Hacker Way
Menlo Park, CA, USA
pedroerp@fb.com

Abstract

In-memory DBMSs are systems that primarily rely on main memory for data storage. Many open source and commercial in-memory OLTP DBMSs were developed and studied in the last decade, such as VoltDB, Oracle TimesTen and Hekaton, as the amount of main memory available outpaced the size of most transactional working sets. As memory density kept increasing, a more recent trend is to leverage in-memory systems to speedup critical analytical workloads. In this scenario, in-memory analytic DBMSs have become a viable option to speedup analytic queries to important datasets.

Today, there are three main configurations in which in-memory analytic DBMSs can be used. First, these systems can be seen as *accelerators* for critical data marts which are part of much larger – and slower – data warehouses. These use cases focus on fast reporting and allowing users to interactively slice and dice portions of the data warehouse data. Second, these systems can be used to incrementally consume logs from messaging systems such as Kafka, providing analytics over realtime data and minimizing data latency. Third, some systems like MemSQL, SAP Hana and Hyper take a different set of trade-offs and are able to handle both OLTP and OLAP workloads in a hybrid configuration, also known as HTAP (Hybrid Transactional/Analytic Processing).

Besides the differences from traditional disk based analytic systems, in-memory analytic DBMSs also differ from in-memory OLTP DBMSs in many aspects. Since analytic queries are mostly composed of large memory scans, the lack of data locality, instruction and data cache misses, virtual and non-inlined function calls, remote NUMA accesses and branch mispredictions all have a non-trivial impact on query performance. JIT compilation at query-time, therefore, becomes a first class citizen to tighten scan loops, as opposed to the traditional approach based on stored procedures. In addition, the traditional *volcano* iterator chaining model, which is usually implemented using virtual calls, becomes less suited for these types of system and more data-centric query processing models are developed.

Furthermore, since analytic systems are designed to store large volumes of data, in-memory analytic DBMSs must carefully choose which parts of the dataset to keep in memory in order to minimize I/O reads at query time. The data structures used to store and index the in-memory data can also be substantially different from the ones traditionally used in clustered and secondary indexes in order to provide fast and cache-friendly scans as well as some form of data pruning. Lastly, transactions on these systems might also have different requirements due to the low number of transactions per second and high ingestion rates, creating opportunity for the development of novel and more lightweight concurrency control protocols.

This tutorial will discuss how in-memory analytic DBMSs are designed and built and outline the architecture of some state-of-art in-memory database systems, stressing the characteristics that differentiate them from the traditional DBMS design literature. In addition, the author will discuss some of the lessons learned while building and providing Cubrick as a service at Facebook, and highlight some of the many research opportunity avenues.

Author

Pedro Eugenio Rocha Pedreira is a Software Engineer at Facebook focused on database research. For the last 5 years he has led the team that develops Cubrick, a new in-memory analytic DBMS that targets interactive queries over highly dynamic datasets. Cubrick powers many internal analytic products and workloads and leverages a new indexing technique called Granular Partitioning, which he has proposed in his Ph.D thesis. Prior to joining Facebook, Pedro spent about 4 years working for the Brazilian Government, supporting the databases and infrastructure leveraged by the Brazilian Electoral Systems. He received his B.Sc., M.Sc and Ph.D from the Federal University of Parana (UFPR) in Curitiba/PR, Brazil.

References

Chen, J., Jindel, S., Walzer, R., Sen, R., Jimsheleishvilli, N., and Andrews, M. (2016). The MemSQL Query Optimizer: A modern optimizer for real-time analytics in a distributed database. *Proc. VLDB Endow.*, 9(13):1401–1412.

Diaconu, C., Freedman, C., Ismert, E., Larson, P.-A., Mittal, P., Stonecipher, R., Verma, N., and Zwilling, M. (2013). Hekaton: Sql server's memory-optimized oltp engine. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD'13, pages 1243–1254, New York, NY, USA. ACM.

Färber, F., Cha, S. K., Primsch, J., Bornhövd, C., Sigg, S., and Lehner, W. (2012). SAP HANA Database: Data Management for Modern Business Applications. *SIGMOD Rec.*, 40(4):45–51.

Kemper, A. and Neumann, T. (2011). HyPer: A Hybrid OLTP&OLAP Main Memory Database System Based on Virtual Memory Snapshots. In *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*, ICDE '11, pages 195–206, Washington, DC, USA. IEEE Computer Society.

Kohn, A., Leis, V., and Neumann, T. (2018). Adaptive execution of compiled queries. In *Proceedings of the 2018 IEEE 34th International Conference on Data Engineering*, ICDE'18, Paris, France. IEEE Computer Society.

Mukherjee, N., Chavan, S., Colgan, M., Das, D., Gleeson, M., Hase, S., Holloway, A., Jin, H., Kamp, J., Kulkarni, K., Lahiri, T., Loaiza, J., Macnaughton, N., Marwah, V., Mullick, A., Witkowski, A., Yan, J., and Zait, M. (2015). Distributed architecture of oracle database in-memory. *Proc. VLDB Endow.*, 8(12):1630–1641.

Neumann, T. (2011). Efficiently compiling efficient query plans for modern hardware. *Proc. VLDB Endow.*, 4(9):539–550.

Pedreira, P., Croswhite, C., and Bona, L. (2016). Cubrick: Indexing millions of records per second for interactive analytics. *Proc. VLDB Endowment*, 9(13):1305–1316.

Pedreira, P., Lu, Y., Pershin, S., Dutta, A., and Croswhite, C. (2018). Rethinking concurrency control for in-memory olap dbmss. In *Proceedings of the 2018 IEEE 34th International Conference on Data Engineering*, ICDE'18, Paris, France. IEEE Computer Society.