DESIGN OF A HYBRID TRANSACTIONAL AND ANALYTICAL PROCESSING (HTAP) DATABASE SYSTEM

By Subashree Venkatesan Sundharesan

STORAGE

To meet the needs of both Online Transaction Processing (OLTP) and Online Analytical Processing (OLAP), a hybrid approach is used in the suggested storage model. A row-based format is used for transactional data, with slotted page structures enabling quick access to a single record. Columnar storage is used in analytical applications to facilitate effective sequential scans and aggregation operations on large datasets.
We introduce in-memory delta storage to manage erratic update patterns. During off-peak hours, this delta store compiles the most recent updates and regularly merges them with the primary columnar repository. This technique maintains optimal query performance while guaranteeing high write throughput during periods of heavy transaction volume.
The unique benefits of each type of storage provide the justification for this design. For quick, frequent transactional queries, row-based systems work well, although columnar storage greatly

UPDATE MANAGEMENT

The update management procedure is governed by a tiered approach:

1. **Instantaneous High:** Volume updates are handled by the in-memory delta store during peak hours (7:00–10:00 PM and 10:00–12:00 AM).

2. **Batch Processing:** This technique lowers the performance load on transactional and analytical queries by running during off-peak hours.

3. **Background Merging:** Combines core columnar storage and delta updates to provide reliable and efficient query performance.

This multi-level approach makes it possible to handle updates with ease without sacrificing system stability or query performance.

QUERY PLANNING AND OPTIMIZATION

Combining algorithms with information gleaned from statistics allows for efficient query handling:

1. **Cost-Based Optimization:** This method maximizes query efficiency by choosing the best join order.

2. **Dynamic programming:** Works best for queries that don't require many table joins.

3. **Greedy Join Ordering:** A strategy that reduces optimization time while handling queries involving several tables.

Crucial statistics including row counts, join-column correlation metrics, data distribution histograms, and column cardinality are used by the query optimizer. With an emphasis on the columns that have the biggest influence, these data are updated adaptively based on workload patterns.

QUERY PROCESSING AND PARALLELISM

The system uses both intra-query and inter-query parallelism to get low latency and high throughput:

- Concurrent execution of separate queries is known as inter-query parallelism.
- Intra-query parallelism divides up operations such as joins and scans among several processing threads or nodes.

Utilizing contemporary CPU architecture, the query engine uses vectorized execution for analytical queries. Furthermore, dynamic load balancing during query execution is guaranteed by a work-stealing mechanism.

ADDITIONAL OPTIMIZATIONS

1. **Materialized Views:** These views optimize response times for recurring analytical queries by automatically creating and updating views for frequently executed queries.

2. **Query Result Caching:** Consistency is ensured via an update-triggered invalidation mechanism that stores results for read-only queries.

3. **Adaptive Buffer Pool Management:** To balance OLTP and OLAP workloads, this technique uses an LRU-K replacement policy supplemented with query-specific hints.

4. **Data Placement Strategies:** These strategies optimize the allocation of data between columnar and row-based stores according to update frequency and access patterns.

CONCLUSION

By combining a hybrid storage type, sophisticated update mechanisms, and state-of-the-art query optimization, this HTAP database design combines transactional and analytical capabilities. The system can handle high-frequency updates with ease and guarantees reliable performance for intricate analytical jobs.

REFERENCES

1. Tan, Z., et al. (2024). HTAP Databases: A Survey. *arXiv:2404.15670v1*.
2. Pavlo, A., et al. "Database Storage Techniques." *CMU Database Systems Course Materials*.
3. GeeksforGeeks. (2024). "Buffer Management in DBMS."
4. Tan, Z., et al. (2022). HTAP Databases: New Horizons. *Proceedings of SIGMOD*