Get Demo Try Free

Real-World HTAP: A Look at TiDB and SingleStore and Their Architectures

2022-07-26 Thought Leadership

Ghimsim Chua

Author: Ghimsim Chua (Product Manager of TiDB Cloud)
Editors: Fendy Feng, Rick Golba

The Rise of HTAP

We have seen an increasing number of Hybrid Transactional and Analytical Processing (HTAP) capable databases in recent years. Well-established players like Oracle, Microsoft, SAP Hana and MariaDB have added HTAP capabilities by adding column stores to their traditional row store databases. Google's AlloyDB and Snowflake's recent announcement of UniStore are now part of the ever growing list of HTAP capable databases that includes SingleStore, TiDB, HyPer, Procella, Wildfire, and Janus.

Why is this? One reason is the explosion of real-time data in a lot of industries, old and new. As industries modernize, increasing amounts of data are collected. Data sizes are growing exponentially larger, more quickly than ever before. A lot of services have streaming data where they need to ingest thousands or millions of events or transactions per second. On top of that, users want to monitor and analyze the data coming in on a real-time basis to be able to make critical business decisions. Reports and analytical queries that take hours or days to complete are no longer acceptable to businesses that are hyper competitive. Furthermore, complex back-end architectures with multiple components and processes are prone to faults or breakdowns. The need to simplify complex architectures and reduce the total cost of ownership is growing.

HTAP to the rescue

A good example of how HTAP can solve these problems is found in the Fintech industry.

Online, mobile, and card-free payment apps have exploded onto the scene over the last several years. Consumers expect to be able to order and pay for items on multiple devices quickly and easily. Merchants accepting these payments expect immediate access to the transactions for analysis of buying trends. As the number of customers grow, Fintech companies need ever faster backend systems to handle terabytes of data while processing thousands or millions of new transactions every second. On top of that, these systems have to process real-time analytical queries for fraud detection and monitoring to compete in the cut-throat marketplace. This places extreme stress on any database back end. Similar explosions of data can be seen in industries as varied as AdTech, IoT, eCommerce, Logistics and so on.

An HTAP database offers Fintech developers the performance and scalability to handle the volume performance, scale, and concurrency their business demands today.

Simplified architecture of HTAP databases

A brief history of SingleStore

With the rise of HTAP databases, there is also a proliferation of different HTAP implementations. SingleStore, with its unique "single storage" architecture, is an interesting case study to compare with TiDB. This analysis helps us better understand the pros and cons of each design.

SingleStore was launched to the public on April 23, 2013 as MemSQL. Early versions of MemSQL only supported transactional queries, and were highly optimized for cases where all data could fit within a computer's main memory. MemSQL was also one of the earliest databases to add general support for an on-disk column-based storage format to work alongside the in-memory row store. More recently, it changed its architecture significantly to move away from exclusive in-memory workloads and also introduced a new way to store data, which they called SingleStore and is also the new name of their company. The new name highlighted the goal of achieving a universal storage format capable of supporting both transactional and analytical use cases.

A comparison of TiDB and SingleStore

TiDB and SingleStore are both MySQL-compatible, distributed, and HTAP-capable database engines designed to have the versatility to run both transactional and analytical workloads with good performance. TiDB is a fully open source database system, whose three kernel components, TiDB, TiKV, and TiFlash, are all open source projects. Additionally, TiKV is a CNCF graduate project. SingleStore, however, is not open source, and there are no known plans to open source it.

Both TiDB and SingleStore have separate compute and storage nodes. In TiDB, the TiDB nodes receive and process results to be returned, while TiKV and TiFlash are nodes that store data. Similarly, in SingleStore, there are aggregator nodes which are the compute nodes, while leaf nodes are the storage nodes.

TiDB vs SingleStore in architecture design

Data store architecture

The data store architectures of TiDB and SingleStore are where the main differences arise.

TiDB data store

TiDB implements its HTAP capabilities by storing data in two different data stores. It has two different kinds of storage nodes: TiKV nodes are row-based stores, while TiFlash nodes are column-based stores. Data in TiKV nodes is durably persisted to block storage like Amazon Elastic Block Store (EBS). To ensure scalability, a TiDB deployment typically contains multiple TiKV nodes, which can be horizontally scaled out to increase data storage capacity.

For each write transaction to be considered committed, the data has to be replicated three times using a Raft consensus algorithm and stored in three different TiKV nodes in three different Availability Zones (AZs) to ensure fault tolerance and high availability. For TiFlash, the data in its column store is kept consistent with the data in TiKV nodes through an asynchronous Raft replication process, which was specifically designed for TiFlash. The third type of node, the TiDB

SingleStore data store[1]

SingleStore also deploys multiple distributed aggregator and leaf nodes for horizontal scalability. Data is replicated twice in different leaf nodes for High Availability. However, each leaf node stores data in multiple different structures, an in-memory row store and column store, and log files and data files which are saved on disk. When it is run in a cloud environment, it can also store to a blob store like AWS S3.

For fast reads and writes, data is read and written to the in-memory row store. Write operations also write affected rows to a log persisted to local disk and replicated to other leaf nodes for durability. When enough rows are amassed in the in-memory row store, a background flusher process deletes the rows from the row store and converts those rows into an in-memory column store segment. A data file is created to store the data in the column store segment and then uploaded asynchronously to blob storage (if running in the cloud) after being committed. Hot data files are kept in a cache locally on disk for use by queries, and cold data files are removed from local disk once uploaded to the blob store. When a node encounters a segment of data that is not in any of its in-memory structures or log files, it pulls the data from the blob store.

Transactions are committed to the tail of the log and replicated to other nodes if run without blob storage. Snapshots of row store data are taken on partitions and written directly to local disk or blob stores. To add more compute to the cluster, new replica databases get the snapshots and logs they need from blob storage and replicate the tail of the log from the master databases. Column store data files are pulled from the blob store on demand and stored in the data file cache as needed.

Implications of each storage architecture

Extraneous storage

Because TiDB stores its data in both row (TiKV) and column (TiFlash) stores, critics argue that it is storing an extra copy of the data beyond what is needed for high availability or durability, wasting storage space. Furthermore, as data is asynchronously replicated from TiKV to TiFlash, there is a replication lag between the two stores.

Get Demo Try Free

Raft replication

The Raft consensus algorithm is built into TiDB's synchronous replication between TiKV nodes. Furthermore, it keeps three copies of data at all times to maintain high availability and durability. For SingleStore, the complicated data handling processes do not use Raft or Paxos algorithms to maintain consistency between primary and secondary replicas. Without Raft to maintain data consistency, a network failure may introduce data inconsistencies in its data partitions, resulting in a split-brain database.

Furthermore, in SingleStore, keeping a persistent state on a local disk or memory requires a local high-performance replication and recovery protocol. It is also less elastic as adding or removing hosts requires carefully moving the local data not yet in blob storage to maintain durability and availability guarantees. In the event of multiple concurrent failures, like a loss of all nodes which have a copy of a single partition, committed data could be lost.

As a result, TiDB is more applicable to mission-critical applications where data inconsistencies cannot be tolerated.

Hot and cold data separation

In TiDB, although there is no separation of hot or cold data; all data is immediately available and used optimally to serve OLTP and OLAP queries. The downside to this is that if the data set is large, more expensive compute and storage resources may need to be used to maintain the entire set of data, even if only a small set of data is frequently used and the rest of the data is never touched.

SingleStore's architecture, on the other hand, uploads data to long term blob storage which is significantly cheaper than block storage. Over time, only hot data is retained in the leaf nodes and no compute resources are wasted to maintain the cold data in the blob storage. As a result, it can store significantly bigger datasets than local disk or block storage, and the costs are significantly less. Additionally, it is able to provision and scale quickly, as new leaf nodes can simply load snapshots and replay logs independently.

At the point of this writing, TiDB is also planning to have the ability to store cold data in blob storage in the near future.

Different partitioning schemes

SingleStore uses hash partitioning, whereas TiDB uses range partitioning. Range-partitioning allows for the use of clustered indexes for fast data access of consecutive rows. Hash partitioning, however, requires extra indexes to be built for faster data access. Additional scans of indexes and data may be required for handling queries that need to retrieve consecutive rows of data.

Scaling

Because TiDB does not store data outside of TiKV and TiFlash nodes, expanding its storage capacity implies increasing the number of TiKV and TiFlash nodes. The cost of these nodes can add up quickly. This can also potentially be an expensive operation as partitions need to be rebalanced when new nodes are added.

For SingleStore, the data storage size is not bound by the number of aggregator or leaf nodes when it is run in the cloud with blob storage. A SingleStore cluster of any size can query data from blob storage written by a much larger cluster. Adding new leaf nodes is also quick, as they only need to load the snapshot from blob storage and replay the logs to bring it up to date.

Get Demo Try Free

Open source	Yes	No
Data architecture	Row/column	Row/column
Storage format	Row/column	Column
Durable storage	Local, EBS	Local, EBS, blob
Hot / cold data	No	Yes
High availability	Yes / raft, 3 copies in 3 AZs	Yes, 2 copies in different AZs
Partitioning method	Range	Hash
Mission critical applications	Yes	No

Wrapping up

Besides the obvious architectural differences, what can we infer from the two different design philosophies? SingleStore's storage architecture seems to *prioritize performance over data durability* while TiDB's storage architecture prioritizes *data durability over performance.* Data durability is the measure of the likelihood of data loss or corruption. All transactions in TiDB are recorded in long-term, durable storage the moment it is committed, and all transactions are recoverable even in the event an entire region is taken down. Furthermore, the use of Raft consensus protocol to replicate three copies of data ensures *data consistency* in the event of a network partition failure. SingleStore, however, does not use any quorum-based algorithms such as Raft or Paxos to replicate data across its stores, so it may be susceptible to split-brain issues.

This fundamental difference does not mean that one design is superior to another, however. **TiDB's data durability is required in many mission-critical or System of Record use cases where data loss is not acceptable,** such as bank transactions, online payment, eCommerce transactions, logistics, and healthcare records. SingleStore's high data ingestion performance is useful in cases where the loss of some data does not have a significant impact on the overall functioning of the system. Losing a few hundred IoT data events, Ad clicks, or media events out of billions of events isn't going to have a material impact to the operational analytics report in a dashboard.

Both TiDB and SingleStore provide great options for developers looking beyond what their traditional OLTP or OLAP databases can do as each has their own advantages in their own application scenarios.

[1] **Reference**: Cloud-Native Transactions and Analytics in SingleStoreDB

Read more about HTAP:
The Beauty of HTAP: TiDB and AlloyDB as Examples
The Long Expedition toward Making a Real-Time HTAP Database

HTAP Real-time analytics

Have a question or comment about the article? Visit the TiDB Forum >

The Beauty of HTAP: TiDB and AlloyDB as Examples

Sign up for the Virtual HTAP Summit 2022, and get inspired from industry leaders around the globe. Register Now

Get Demo Try Free

SHARE:

Related Resources

Product

AI Takes Over GitHub Analysis: The Future of Data Science is Here

Thought Leadership

The Next Revol

View All >

Subscribe to Stay Informed!

Name

Email Address

>

TiDB Cloud

Get the massive scale and resiliency of TiDB databases in a fully managed cloud service

Try Free Learn More >

TiDB

Sign up for the Virtual HTAP Summit 2022, and get inspired from industry leaders around the globe. Register Now

Get Demo Try Free

Sign up for the Virtual HTAP Summit 2022, and get inspired from industry leaders around the globe. Register Now