(/learn)

Summary: BigTable

Here is a quick summary of BigTable for you!

We'll cover the following

Summary#

- BigTable is Google's **distributed storage system** designed to manage large amounts of structured data with high availability, low latency, scalability, and fault-tolerance goals.
- BigTable is a **sparse**, **distributed**, persistent, **multidimensional sorted map**.
- The map is indexed by a unique key made up of a row key, a column key, and a timestamp (a 64-bit integer, "real time" in millisecond).
- Each row key is an arbitrary string of up to 64 kilobytes in size, although most keys are significantly smaller.
- Unlike a traditional relational database table, BigTable is a **wide-column datastore** with an unbounded number of columns.
- Columns are grouped into **column families**. Each column family stores similar types of data under a 'family:qualifier' column key.
- The row key and the column (family:qualifier) key uniquely identify a data cell. Within each cell, the data contents are further indexed by timestamps providing multiple versions of the data in time.
- Every read or write of data under a single row is atomic. Atomicity

across rows is not guaranteed.

- BigTable provides APIs for metadata operations like creating and deleting tables and column families. BigTable clients can use data operation APIs for writing or deleting values, lookup values from individual rows, or iterate over a subset of the data in a table.
- A table is split into smaller ranges of rows called **Tablets**. A Tablet holds a contiguous range of rows.
- Tablets are the unit of distribution and load balancing.
- BigTable architecture consists of **one master server** and **multiple Tablet servers**.
- Master is responsible for assigning Tablets to Tablet servers, as well as monitoring and balancing Tablet servers' load.
- Each Tablet server serves read and write requests of the data to the Tablets it is assigned.
- BigTable clients communicate directly with the Tablet servers to read/write data.
- Each Tablet server stores the data in immutable **SSTable** files which are stored in **Google File System (GFS)**.
- New committed updates are first stored in a memory-based **MemTable**.
- BigTable performs all read operations against a combined view of SSTables and MemTable.
- Periodically, the MemTable is flushed into an SSTable, allowing for efficient memory utilization.
- To enhance read performance, BigTable makes use of caching and **Bloom filters**.
- BigTable relies heavily on distributed locking service **Chubby** for master server selection and monitoring.

References & further

reading#

- BigTable (https://research.google/pubs/pub27898/)
- SSTable (https://medium.com/databasss/on-disk-io-part-3-lsm-trees-8b2da218496f)
- Dynamo (https://www.allthingsdistributed.com/2007/10/amazons_dynamo.html)
- Cassandra (https://cassandra.apache.org/)
- HBase (https://hbase.apache.org/)

← Back

BigTable Characteristics

Next →

Quiz: BigTable

☑ Mark as Completed

Report an Issue