KV storage engine with LSM-tree structure

1. Technical topic Sharing

LSM-tree writes data sequentially in append-only manner, which consists of SkipLists in memory and SSTables on disk. Different from the B+ tree, data is not globally ordered but internally ordered at each level. Therefore, an additional metadata table is maintained that records the metadata, minimum key, and maximum key of each SSTable for data query.

Because of the append-only way of writing data, instead of the in-place way of the B+ tree, a background thread is needed to perform a compaction to merge the written memtable and write it to the SSTable.

In a multilevel SSTable with LSM-tree structure, the files at each level are ordered, but the whole is unordered, which leads to read amplification. Furthermore, the order of each level requires to be established by compaction, which leads to a serious problem of write amplification of the data.

2. Extra: Analysis of application examples

2.1. Ethereum Requirements Analysis

1.During block processing, EVM linearly waits for state data to be read, so the throughput of Ethereum is closely related to the reading performance of the storage engine.

2.Ethereum sets the amount of gas consumed for various operations of smart contracts, which is used to control the consumption of computing resources. needs to be storage engine read and write performance to remain stable.

2.2. Storage Engine Analysis

In the case of Geth, Ethernet's current storage engines are LevelDB as well as PebblesDB.

For Requirement 1: Both are storage engines based on the LSM-tree structure, providing better write performance and worse read performance.

For Requirement 2: The read and write amplification of the LSM-tree structure makes the actual computational resources consumed for EVM operations with the same Gas price inconsistent.

Take the SLOAD operation as an example, the difference in computational resources consumed can be several times if the data are at level 1 and level n, respectively.