TencentDB for TcaplusDB FAQs

Contents

FAQs
Database Features

Last updated：2024-10-15 17:52:59

Does TcaplusDB support data removal?

TcaplusDB supports table-level data elimination based on the last write time of the record.

What are the data structures of TcaplusDB?

The data structures supported by TcaplusDB include list arrays, querying by partial key (index), key-value, and key-object (where the value of a single key can be any data structure, for example, the game server can serialize a Lua table into the value field).

What is the single-instance memory capacity and CPU utilization of the TcaplusDB API?

The memory consumption for a single instance of the API is up to 73MB, and CPU utilization is up to 30%.

How many tables are in a table group in TcaplusDB?

The maximum number of tables in a single table group in TcaplusDB is 256. If the number of tables in your group exceeds 256, you can add new table groups or merge tables. If you need technical support, please submit a work order and select "Other Tencent Cloud Products".

What are the restrictions on the key and value fields in TcaplusDB?

The number of key fields of the generic table is 4, the number of key fields of the list table is 3, and the size of a single key field is 1024 B. The number of value fields of the generic table is 128, the number of value fields of the list table is 127, the size of a single value field is 256 KB, and the maximum value of the overall record is 1 MB.

How long is the TcaplusDB backup file retained?

The engine files backed up by TcaplusDB are saved for 7 days, and the Ulog process is saved for 7 days. The storage periods in different TcaplusDB environments vary. For specific TcaplusDB environment details, please contact customer service.

Is the game server connected to all tcaproxy (access layer)?

To save costs on maintaining TCP connections for gameserver and tcaproxy (access layer), gameserver can choose to establish connections with selected tcaproxy (access layer).

How do I perform data analysis of TcaplusDB data?

TcaplusDB data can be exported in any format, including json, pb and other formats, and can be imported into data analysis systems such as tdw. TcaplusDB supports the import of real-time data into MySQL database and so on.

What is the maximum size of a single table in TcaplusDB? What is the limit on the number of records?

A single table can be subdivided into 10,000 data shard, each data shard is 256 GB, that is, the total size of a single table is 10,000 * 256 GB. A single table has no limit on the number of records, and the number of records in a single table is related to the size of a single record.

Does TcaplusDB support multi-table transaction operations and batch write operations?

TcaplusDB currently does not support multi-transaction operations, meaning it cannot simultaneously commit multiple operations and roll back all actions if subsequent operations fail. In such cases, solutions usually need to be found from the business side. A "subtract first, then add" strategy can be employed, or recording reconciliation logs of critical operations, and batch write operations are also not supported temporarily.

Is the TcaplusDB API upgrade backward compatible?

TcaplusDB API upgrade is forward compatible, and existing APIs, command words, and features will not be modified.

What is TcaplusDB?

Game Database (Tcaplus Database, TcaplusDB) is a distributed NoSQL database designed by Tencent specifically for games. It provides high performance, millisecond-level read/write latency, and secure data storage services for managing explosive growth and massive game data without business perception. It is widely used in hundreds of Tencent games, including Peacekeeper Elite, Honor of Kings, and CrossFire, supporting data requests from over 40 million concurrent users daily.

What scenarios is TcaplusDB suitable for?

Mobile games, PC games, web games, support for both zone and server segmentation and global server coverage deployment methods, and also non-game scenarios requiring high concurrency and low latency access requirements.

What advantages does TcaplusDB have over Redis+MySQL as a game database?

TcaplusDB uses a hybrid deployment method combining memory and efficient SSD Cloud Block Storage (CBS), meeting the needs for rapid data read/write and persistent storage simultaneously

For relational database users, TcaplusDB offers a SQL-like TCCLI (Tcaplus Client), allowing users to seamlessly switch and perform CRUD operations

Users do not need to worry about large table files causing slow query loading. TcaplusDB supports single table sizes up to 2.56PB without reducing query efficiency

Redis+MySQL scaling requires manual intervention and cannot cater to sudden spikes in business demand, which also incurs high costs. TcaplusDB users only need to focus on business development while all database scaling and performance optimization are handled backend by professional DBAs.

How does TcaplusDB ensure data security?

Data Disaster Recovery: The primary/secondary node clusters mainly adopt intra-city cross-IDC deployment (or cross-rack, cross-floor deployment methods, with primary/secondary node synchronization delay up to 10ms), with binlog backups every 15 minutes and full cold backups daily. Every module of the entire cluster supports lossless upgrades and scaling, ensuring business continuity without user perception

Data Consistency: Real-time synchronization of master-slave data with a req/ack mechanism ensures complete consistency. The master-slave switch is confirmed only after ensuring data consistency between master and slave. In case of failure, data recovery is performed using cold backups and binlog streams, ensuring data integrity. Periodic full checks for consistency and an exception repair mechanism are in place for master-slave data. At the user request level, the response code from real-time feedback determines the success of data operations

Data Content Security: Data landing uses CRC checks to prevent tampering. Data in CFS employs Google Snappy Compression, and data read/write operations involve serialization and deserialization. Even if data files are hijacked, the data content cannot be parsed
Access Security: Integrated with Tencent Cloud User Permission Management System (CAM). TcaplusDB backend supports access based on IP allowlists, allowing specified IP clients to read/write TcaplusDB data. Data interactions within a table group require password verification, and every user operation is logged in audit logs
External Compliance: Data files use AES-128-CBC encryption, complying with EU GDPR standards.

Field Size Limitation for TcaplusDB API Versions

For APIs earlier than Tcaplus 3.32.0, a single value field is 128KB, and a single record is 256KB.
For APIs from Tcaplus 3.32.0 to Tcaplus 3.40.0, a single value field is 256KB, and a single record is 1MB.
For APIs from Tcaplus 3.40.0 and onwards, a single value field and a single record can be up to 10MB.

What are the current sizes of TcaplusDB's send and receive buffers?

Currently, the send and receive buffers for gameserver and a single tcaproxy are 10MB each.

How to use the error codes for the batch interface?

The error codes for the batch interface are divided into two types: Total Error Code and Sub-operation Error Code. When at least one sub-operation is successful, the Total Error Code is 'Success'; when all sub-operations fail, the Total Error Code is 'Failure'. The Total Error Code can be obtained through the response return value, while the Sub-operation Error Code is obtained by fetching the return value of an individual record.

Is the maximum number of primary fields for a Tcaplus table 256?

Currently, a table supports up to 256 primary fields

For a game, how many connections does Tcaplus support at most?

By default, a proxy supports up to 3000 connections, but the connection count can be increased by modifying the configuration. Additionally, an API instance currently supports connecting up to 200 proxies

Is Tcaplus's access to the database parallel?

It supports parallel access. Requests for each connection (session) on the API side are serial, but different connections are parallel. On the API side, connections are not thread-safe, so avoid using the same connection with multiple threads.

How much data can a single shard store?

A single shard can store up to 256GB of data. It is generally recommended to maintain the data volume under 10GB. Note: A shard has initial space, but it is not fixed and depends on the specific table engine settings, such as the number of Hash Buckets. A table supports up to 10,000 shards, each with 256GB. TcaplusDB is a distributed database with no row limit.

What is the compatibility of TcaplusDB PB versions across different versions?

PB is forward and backward compatible

Compared to int, what should be noted when using string as Tcaplus's primary key or sharding key?

The performance gap compared to int is not significant. The general principle is to ensure the key is not too long.

If a table has a large data volume, for example, 1 billion records, and can't fit on one physical machine, will Tcaplus distribute this data across multiple physical machines?

During operation, machines can be added to distribute the data to new storage, and each storage group follows a master-slave mode.

When master-slave data is synchronized in real-time, if the master machine crashes due to physical reasons, will there theoretically be some short-term data loss?

Theoretically, there is a possibility of short-term data loss. It is an eventual consistency synchronization due to the system flushing from memory to disk. If the master machine crashes, it will immediately switch to master-slave mode.

If the remote meta version of Tcaplus is higher and the local svr's Tcaplus meta version is lower, will the differing data be stored in tcaplus_db (for instance, if a new field is added)? Or will it be directly discarded?

Written with default values

Regarding the Table Partition Factor in the table Definition, is there a default Table Partition Factor if not defined? Can existing table

splittablekey add new fields?

TDR Table: Without a defined Table Partition Factor field, the system defaults the primary key field as the Table Partition Factor field.
PB Table: Without a defined Table Partition Factor field, the intersection field of multiple local indexes is the Table Partition Factor field. If no local index intersection field is defined, the primary key is the default Table Partition Factor.
Once the table splittablekey is defined, it cannot be modified, added, or deleted.

Regarding changes to the List Table, note the first addition of customattr2 in Definition listnum:

The number of elements in the List Table can be increased, but during the first addition of customattr2 for table modification, listnum in customattr2 cannot be increased immediately. It should be modified after the change is successful. Otherwise, the change will fail.

Is Tcaplus in Primary/Backup Mode? During a write request, does the host send a response package as soon as the write is successful, or does it wait for standby machine synchronization before responding?

Yes, it is a Primary/Backup Mode with one primary and one secondary. The host sends a response package as soon as the write is successful.

Database Use

Last updated：2024-10-15 17:53:24

How should I get the definition of the error codes in a response packet?

We recommend that you call functions `TcapErrCode::TcapErrCodeInit` and `TcapErrCode::GetErrStr` in the game server to get error codes, or you can search locally for the TcaplusDB API header files.

How does TcaplusDB's optimistic locking work and how to use it?

Let's use the purchase of train tickets as an example:

1. 100 people want to snap up the same train ticket. The recorded version number of the train ticket is 10, and the 100 people use the same recorded version number to snap up the train ticket.
2. 100 people conduct write operations to this ticket. After the write operation is completed, the recorded version number will increase. Therefore, for the operation of a single key, the tcapsvr worker threads are queued. When the first person successfully purchased the ticket, the recorded version number of the train ticket changed to 11, and the recorded version number of the remaining 99 people is still 10.
3. When tcapsvr handles the write requests of the 99 people, an error will occur because the recorded version number at tcapsvr end is inconsistent with the version number in the request. The concurrent principles are as follows:
    - N requests. If only the first request succeeds and the remaining N-1 requests fail, the version protection rules are used.
    - N requests. If all N requests are required to be executed, no version protection rules are required. TcaplusDB operates on the same key.

    Queue execution calls the function `SetCheckDataVersionPolicy`, which includes values:
    - `CHECKDATAVERSION_AUTOINCREASE`: Checks the record version number. The version number will auto-increase only when it matches the server's version number.
    - `NOCHECKDATAVERSION_OVERWRITE`: Does not check the record version number. It forces the client’s record version number to be written to the server.
    - `NOCHECKDATAVERSION_AUTOINCREASE`: Does not check the record version number. It auto-increases the server's version number.
    - The default type is `CHECKDATAVERSION_AUTOINCREASE`, which is recommended for use.

What are the use cases and precautions for the LIST table?

Where there is a 1:N usage scenario, when N < 1024, the list table is prioritized, such as storing the player's most recent 100 emails, the most recent 100 battle records, and so on. The list table supports insertion at the head of the queue and removal at the end of the queue, insertion at the end of the queue and removal at the head of the queue, as well as the Top N operations sorted by insertion time. The number of units under a single key can be increased by modifying the table, because the old data needs to be compatible and cannot be modified to become smaller. You can obtain the total number of records under a single key by using listgetall. It is recommended to obtain data according to the offset and limit of a certain threshold that you set. For listreplace, listdelete, and listdeletebatch, you need to specify the correct index. For listaddafter, you need to specify the elimination rule when the number of element units under a single key reaches the upper limit. Since the SetListShiftFlag function settings are called, the list table has two increasing and two obtaining directions. There are 4 possibilities:

1. Query A, B, C, D, E as offset = positive number and limit = 2, and the result is A, B; C, D; E.
2. Query A, B, C, D, E as offset = negative number and limit = 2, and the result is D, E; B, C; A.
3. Query E, D, C, B, A as offset = positive number and limit = 2, and the result is E, D; C, B; A.
4. Query E, D, C, B, A as offset = negative number and limit = 2, and the result is B, A; D, C; E.

Additionally, calling `GetRecordMatchCount` can retrieve the total number of records.

What is the difference between INSERT, UPDATE and REPLACE?

For INSERT operation, when the key does not exist, an INSERT operation is performed; when the key exists, an error code is returned: `TcapErrCode::SVR_ERR_FAIL_RECORD_EXIST` .
For REPLACE operation, when the key does not exist, an INSERT operation is performed; when the key exists, if the optimistic lock is used, different operations are performed based on the result of the optimistic lock. If the operation is successful, the REPLACE operation is performed, and if it fails, an error code is returned: `TcapErrCode::SVR_ERR_FAIL_INVALID_VERSION;` If optimistic locking is not used, the REPLACE operation is performed.
For UPDATE operation, when the key exists, if the optimistic lock is used, different operations are performed based on the result of the optimistic lock. If the operation is successful, the UPDATE operation is performed, and if it fails, an error code is returned: `TcapErrCode::SVR_ERR_FAIL_INVALID_VERSION` , If optimistic locking is not used, the UPDATE operation is performed. When the key does not exist, an error code is returned: `TcapErrCode::TXHDB_ERR_RECORD_NOT_EXIST` .

How do I get the number of records in a table?

There is a count command word in the TcaplusDB API. If tcaplus_client is used, you can use the count table name command to get the number of records in the table.

Does TcaplusDB support traversal operations?

TcaplusDB supports traversal operations, including traversal operations for GENERIC and LIST tables. You can use the `SetOnlyReadFromSlave(bool flag)` API to traverse the data from the secondary tcapsvr, which will not affect the services provided by the primary tcapsvr.

Does TcaplusDB support updating and obtaining partial fields?

TcaplusDB supports partial field updates. When updating and obtaining records, it is recommended to explicitly call API `SetFieldNames(IN const char* field_name[], IN const unsigned field_count)` to determine the fields of this read and write operation, and reduce the network traffic overhead caused by invalid fields.

Is TcaplusDB order-preserving for the continuous operations of a single primary key?

For the same game server, the operations of the same primary key are order-preserving, while the operations of different primary keys are not order-preserving. For different game servers, the order is not preserved.

Does TcaplusDB support table definition changes?

TcaplusDB supports table definition changes. If you simply add common fields and modify macros, use table change operations; for the rest of the scenarios, you need to dynamically modify the table structure. That is, use the data migration + log flow process to achieve table definition changes. Please submit a ticket and select "Other Tencent Cloud Products" to apply.

How do I know whether the packeting of response packet has ended?

For traversal, judge whether the traversal ends according to state, that is, API GetState. For the rest of the packeting scenarios, judge whether the packeting ends according to the function `HaveMoreResPkgs`.

What is the difference between GetRecordCount and GetRecordMatchCount?

A request may have N response packets. If there are multiple packets, GetRecordCount refers to the number of records in the response packet, and GetRecordMatchCount refers to the data records stored on the tcapsvr (storage layer) end (total number of records for a single key).

Does TcaplusDB have a pass-through field?

The CS protocol of TcaplusDB is divided into two parts: Head and Body. UserBuff (maximum size is 1 KB), AsyncID and Sequence in Head are all pass through fields. You can use them accordingly.

What is the role of SetResultFlag?

When performing write operations, the response packet supports returning records. When performing read operations, calling this function is invalid. The description of the specific value of result_flag is as follows:
0 refers to: Simply return whether the operation is successful or not without returning the value field.
1 refers to: Return data consistent with the request field.
2 refers to: The latest data for all fields of the change record must be returned.
3 refers to: The old data for all fields of the change record must be returned.
The SetResultFlagForSuccess API can set the data returned if the operation succeeded; the SetResultFlagForFail API can set the data returned if the operation failed.

Can I perform increase operations on multiple non-primary key fields at a time? What if the primary key does not exist?

The increase operation can increase multiple common fields at a time, which requires the request passed by gameserver to assign values to multiple fields. If one of the keys does not exist when the increase operation is performed, it can be set by the SetAddableIncreaseFlag function. If the key does not exist, it will insert the key and perform the increase operation. The non-increase field of the key will not be actually stored, and the default value of the non-increase field will be used when the record is read; if the key exists, the increase operation will be performed directly.

How do I reduce traffic costs when TcaplusDB reads records?

When TcaplusDB reads a record, it can be set so it does not return the value field if the record has no change in a fixed period of time, neither does it return the value field if the record version number does not change. For more information, see the SetFlags function.

Does TcaplusDB support rollbacks? What is the supported rollback granularity?

TcaplusDB supports rollbacks, including all-server/all-region rollback, single-table rollback, and can roll back N records from 100 billion records. It also supports cold standby time rollback (most recent 01:05:00 am), precise time rollback (to the second), and fuzzy rollback (you can specify the rollback rules). For speed reference, precise time rollback takes about 2 hours for a rollback of 300 GB data and 200 GB Ulog. The principle of the cold standby time rollback is to replace the engine file. The principle of precise time rollback is to roll back the cold standby engine file + Ulog to the needed time point. A key-based rollback requires you to block these keys first and then unblock them after TcaplusDB has finished the rollback.

How efficient are queries by partial keys (indexes) with TcaplusDB?

It is recommended to use queries by partial keys (indexes) in 1:N (N > 1024) app scenarios. The number of primary keys under a single index key is equal to 10 GB/the size of primary key of a single record. A single read-write index operation takes about 100 ms (there are more than 100,000 pieces of data records under a single index key).

What is the timeout mechanism of the TcaplusDB API? What does the 'it is timeout' error mean?

The TcaplusDB API assigns an ID to each request. After successful transmission, the ID is added to the timeout data structure. If the response packet for this request comes back, the ID is then deleted from the data structure. If the response packet hasn’t been processed by the application layer within 3 seconds, an error log will be printed with keywords including 'it is timeout'. At this point, it is necessary to check if the gameserver is blocked. It might be due to packet loss while the tcaproxy (access layer) is returning packets to the gameserver side, or the response packet has arrived at the gameserver side, but the gameserver hasn't processed it in time
TcaplusDB recommends implementing your own timeout mechanism, allowing for the retrying of timed-out requests and other processes.

What are the roles of SendRequest, OnUpdate, and ReceiveResponse functions of TcaplusDB API?

The role of SendRequest is to send requests. This request may have been sent to the network, or it may be blocked in the sending channel of the gameserver and a tcaproxy (access layer). The role of ReceiveResponse is to get the response packet from the local receive queue. OnUpdate is responsible for sending the requests of the send queue to the network and receiving response packets from the network to the receive queue. In message-driven programming mode, the recommended call frequency for OnUpdate calls is 1 ms.

How does TcaplusDB achieve global auto increase of fields?

You need a single Definition table, set one value field type to int64_t (multiple value fields can enable counter arrays), multiple gameservers can concurrently perform an increase operation on a single value (no need to set version protection rules), and obtain the increase result of this return, then the result of this increase is globally incremental.

What rules are defined for queries by partial keys (indexes)?

The index key must be a part of the primary key, the index key must contain the shardkey, and the index key cannot be the primary key.

How do I achieve two-way query of the ID and name of a single table?

The third key field x is used as the shardkey, and the index is built on ID and x, name and x to achieve this feature. For example, when storing player information, the player's district can be added, that is, the primary key is uin, name, district, and the index key is uin and district, and name and district.

Does tcaplus_client support the display of fields at the second nesting level or above?

Yes. You may execute help select to see how select * into a.xml is used.

What should be noted when changing the table in TcaplusDB?

1. You can only add new fields. You cannot modify the type and name of an existing field, or delete an existing field. To modify these, you need to dynamically modify the table structure.
2. The length of an array or string in a non-primary key field can be increased but not reduced. To modify these, you need to dynamically modify the table structure.
3. For a new field, its version should be incremented by 1 from the existing version number. Accordingly, the entire xml file’s version number (at the top of the file, the Definition's) should also be incremented by 1 (consistent with the new field’s version number). The TcaplusDB table must be updated before making changes to the gameserver side table. If you need to dynamically modify the table structure, TcaplusDB supports it, please submit a ticket and select "Other Tencent Cloud Products" to apply.

What should be noted when defining the table in TcaplusDB?

1. The refer field needs to be added to the count field.
2. The sub-table factors requires high dispersion.
3. The index key cannot be the same as the primary key.

How does the backend perform version control when the TcaplusDB table structure changes?

TcaplusDB solves multi-version compatibility issues based on the table's version number (Version No.), meaning the TcaplusDB storage layer will store data of the same table with different version numbers; therefore, when the gameserver or client performs data read and write operations, you need to specify the table's version number.
When different clients alternately write to the same table and use different versions of the table Definition, PB tables and TDR tables handle differential fields differently.
Assuming gamesvr1: the table has fields abc
Assuming gamesvr2: the table has fields ab
Assuming TcaplusDB: the table has fields abc

For PB tables:
1. Assuming abc are PB table fields, and two servers alternately write, the c field will be cleared.
For TDR tables:
1. Assuming abc are primary TDR table fields, and two servers alternately write, the c field will not be cleared.
2. Assuming abc are secondary TDR table fields, and two servers alternately write, the c field will be cleared.

How does TcaplusDB handle request latency?

If a request does not respond within 3 seconds, TcaplusDB will log an error containing the keyword timeout; it is recommended that the client request code also includes timeout logic or retry count.

How does TcaplusDB ensure read-write request stability?

TcaplusDB defaults to a Master-Slave model where the Master handles user read and write requests and the Slave acts as a hot standby node. When there is a sudden surge in QPS, read-write shunting will automatically activate, with the Master handling write operations and the Slave handling read operations until background expansion is complete.

Does TcaplusDB support a retention policy?

Yes, TcaplusDB will retain all data by default and allows users to set a retention period. Expired data will be automatically eliminated. This is currently supported by ticket.

Does TcaplusDB support cross-regional migration?

Yes, TcaplusDB supports both live migration (using cross-zone hot standby) and cold migration. It allows data export as Json files and also supports MySQL data import into TcaplusDB, currently supported by ticket.

How does TcaplusDB ensure hot data?

TcaplusDB uses the LRU cache algorithm to ensure data hotness.

How does TcaplusDB store gamer data?

A single TcaplusDB record represents a single player's data, operated based on primary key fields or indexes.

Can an index be added or a primary key modified after TcaplusDB table creation?

No, if there is a need for modifications, the table must be recreated.

Does TcaplusDB support transactions?

Not at the moment. It is recommended to operate the relevant field definitions in the same table for a transaction-like experience.

Will asynchronous request responses in a single thread be processed sequentially? Does it require using asyncid for synchronization? (primarily for features, not using sync API)

In a single thread, read and write operations for the same key are processed sequentially, ensuring order. The sequence of operations for different keys is not guaranteed. If the project team uses asyncid to map request and response, it works as a passthrough field including asyncid, sequence, and userbuff.

What is the delete performance? Is it necessary to use the update command instead of delete?

Delete is a regular write operation with no performance issues, and the project team can perform delete operations at any time.

Threads and Concurrency

The TcaplusDB API is single-threaded and does not support multithreading. One tcaplus API object corresponds to one request and one response. Since the TcaplusDB API is non-thread-safe, multiple threads cannot use the same TcaplusDB API object. For concurrent scenarios, it is recommended that each thread use a separate TcaplusDB API object. From an operational perspective, having 1-2 TcaplusDB API instances on a single machine is sufficient, and the QPS of a single TcaplusDB API instance can reach 70,000-100,000.

What is the SDK memory usage?

A TcaplusDB API instance will occupy around 200M of physical memory. The instance uses static allocation, and memory consumption mainly lies in the protocol, with separate structures for receiving and sending packets. Therefore, some memory needs to be pre-allocated. Note: Different versions of the TcaplusDB API may pre-allocate different amounts of memory due to varying supported record sizes.

Can the Tcaplus primary key be a structure or only a basic type?

Using structures is not recommended. Although theoretically possible, using structures can complicate later stages for various tools, problem location, rollback, and providing the key.

Error 525 when requesting to read data

This generally indicates a backend timeout. If there is high concurrency, it could be an issue with communication between the gameserver and tcaproxy. Check the following:
Is tcaproxy functioning normally?
Is the TcaplusDB API version greater than the access layer version?
Has the TCP connection count of tcaproxy exceeded the limit?

Does distributed index support the delete method with where conditions?

You can use distributed indexing to read the data and then delete each entry one by one.

If a newly added table group with a new global index rolls back data, will the index remain or be rebuilt?

Distributed indexing is automatically maintained during data insertion. When using data import, data will be synchronized to the global index.

If a table has multiple primary keys, can you delete the data by specifying one of the primary keys?

Primary key, delete according to the delete command. Local index, delete according to the deletebypartkey command.

If the telnet ip port can access the tcapdir address but fails to connect to the database cluster, how to investigate?

Telnet to port 9999
Confirm whether it's a Tencent Cloud environment or an ordinary cluster environment.
Tencent Cloud requires connecting to the customer's VIP+9999 port, while ordinary clusters connect to the real dirip+9999 port.
Since a cluster has multiple dirs, it's best to get the API connection log to see which dir failed to connect or if connected and failed due to proxy version higher than the API version.

Does Tcaplus support different zones within a single app using different protocols?

Tcaplus supports different zones within a single app using different protocols, e.g., zone1 using PB protocol and zone2 using TDR protocol.

Will table reconstruction or deletion and recreation affect the cache configuration?

Reconstruction can retain the configuration, but deleting and recreating will remove cache backup since cache is deleted with the table.

Does adding columns require table reconstruction?

Adding columns only requires performing online modification.

Does expanding the count in the table structure require table reconstruction?

No need, you can directly modify it, and there's no need to add a version number. The count can only be increased, not decreased.

Do you need to change the version number of tables.xml when deleting a table or rows from a table?

For an existing table, you cannot delete fields. You can only add new fields. If you need to delete fields, you must delete the entire table and then rebuild it.

Can the length of the blob on the business side be dynamically adjusted without affecting the business side?

Yes, the count can be increased since it was defined in the count, and there's no need to add a version number.

Does changing a primary field in the table require updating the write cache?

If these fields do not need to be configured for write cache, then no changes are necessary. If they need to be configured, then the write cache configuration must be updated.

Redis Replacement Scenarios

1. Can TcaplusDB replace Redis's pub/sub feature?

   Currently, TcaplusDB does not have a pub/sub feature. You can simulate it using a list table, such as Process A writing data to the list table and Process B reading data from the list table. This would increase code modification costs.
2. Does it support Redis's expired (auto expiration feature)?

   Currently, TcaplusDB supports data elimination. For example, if the account table is set for 30-day elimination, it will physically delete data with no write operations within 30 days. This feature is supported.
3. The TcaplusDB Client has setttl. Does it support the API?

   Setting setttl for a single record is supported by the TcaplusDB API. However, in the project group's usage scenario, you can set full table data elimination. For example, a table with hundreds of millions of records can be set to eliminate data with no write operations within 30 days.

4. Can TcaplusDB's list table be used to replace Redis's stored set feature?
   If it's TopN, it can be achieved using a list table, meaning this feature is supported.
5. Is there something like Redis's pipeline, where multiple queries can be processed with a single request? For example, if there are 10 queries, 1 request can handle all 10 queries.
   TcaplusDB provides batch operations to meet similar needs. For details, please refer to the SDK: https://github.com/tencentyun/tcaplusdb-go-sdk/blob/v0.0.10/tdr/request/batchget_req.go.

Rules for the growth of List table indexes

1. idx starts from 0
2. idx increments by 1 starting from 0 until it reaches the maximum value of the int type. Then, an available idx will be used, meaning under any circumstances, the currently used idx will not repeat.
3. If a key is deleted, idx starts from 0 but can also continue from the last idx. Default is from 0, but can be set to continue from the last idx using SetFlags.

/*TCAPLUS_FLAG_LIST_RESERVE_INDEX_HAVING_NO_ELEMENTS
*   After setting this flag, when deleting the last element of a List table, the index and version need to be preserved.
*   For operations like ListDelete, ListDeleteBatch, and ListDeleteAll, when deleting the last element of a list table,
*   Setting this flag ensures that when a new List record is written, the version number increases incrementally and is not reset to 1.
*
*   Use cases :
*   Businesses need to decide whether to preserve the index and version when deleting the last element of a table
*   Mainly concerns the user experience with List tables
*
*/ int SetFlags(int32_t flag);
Indexes are discontinuous, they only increase, for example: 1 2 3 4 5, deleting 3 becomes 1 2 4 5. If AddRecord(2) is executed, it becomes 1 2 6 4 5.
If all elements of the list under a key are deleted, it is equivalent to the deletion of the key itself.
listaddafter supports deletion at the head and insertion at the tail; or insertion at the head and deletion at the tail.
listdelete, delete idx.
TCAPLUS_API_LIST_PRE_FIRST_INDEX: New element is inserted before the first element.
TCAPLUS_API_LIST_LAST_INDEX: New element is inserted after the last element

Regarding the upper limit issue of the increase-related interface, if the current value is 97 and increase5 with an upper limit of 100 is executed, will it succeed partially?

It won't succeed. It will return the error code SVR_ERR_FAIL_OUT_OF_USER_DEF_RANGE.

If tcaplus performs an insert operation on the table and the primary key of two consecutive insert operations is the same, will the second operation overwrite the first or fail?

The second operation will fail with error code -1293 SVR_ERR_FAIL_RECORD_EXIST indicating the insert record already exists

In the protoc file, can all defined types be used in tcaplus? For example, the any type

Yes, but conditional updates do not support any or union types.

How to set up traverse backup table in go SDK?

Set it through the SetOnlyReadFromSlave interface call

ListGetAllRequest returns an error SVR_ERR_FAIL_INVALID_SUBSCRIPT

The parameters for offset and limit are incorrect when pulling according to them. Please check the offset and limit parameters.

Creation method of local index and distributed index (global index)

Local indexes are defined in the table structure file and cannot be modified or deleted once defined. Distributed indexes are created on the corresponding console page (Tencent Cloud Console) and can be modified or recreated.

Does distributed index (global index) support fuzzy search with segmentation for strings?

Supports LIKE fuzzy query, but does not support word segmentation search.

Can fields in the table Definition be deleted?

Fields in an existing table cannot be deleted

In PB Table's add operation, if add fails because data already exists, is there a way to return the existing data?

Yes, in the int SetMessageOption(const ::google::protobuf::Message &msg, int32_t item, const std::string &option) function, the option can be set to return old data of all fields in the change record.

Does TcaplusDB Go SDK support connecting to the directory server (Tcapdir) via domain name?

Supports connecting to the directory server (Tcapdir) via domain name.

For TcaplusDB's list structure, is there an interface that only queries the number of elements in the list?

Currently, it is not possible to query the size of a list

What is the default elimination method when the Tcaplus list table is full?

There are three elimination methods, with TCAPLUS_LIST_SHIFT_HEAD removing the first element as the default.
TCAPLUS_LIST_SHIFT_NONE: Deleting elements is not allowed, and insertion fails if the LIST is full
TCAPLUS_LIST_SHIFT_HEAD: Removes the first element
TCAPLUS_LIST_SHIFT_TAIL: Removes the last element

If Tcaplus single row record is set to be eliminated after 300 seconds, is it eliminated 300 seconds after insertion? How is the lease renewed during this period?

Table level data elimination is based on the record's last modified time; an update or replace operation will automatically renew the term. The accuracy is roughly at the day level, suitable for scenarios such as retaining emails or battle records from the last 30 days. Record level elimination is set and renewed via setttl, suitable for scenarios where precise ttl time or different elimination times for different records are required. Accurate to milliseconds.

A TcaplusServer object can only access different zones within one set

1. Can multiple TcaplusServer objects achieve cross-set access -- Each object can only access one set, not across sets; for cross-set access, multiple APIs or multiple TcaplusServer objects can be used.
2. How is memory usage of multiple TcaplusServer objects -- Each API approximately uses around 40M of memory.

For the UpdateNetwork API, does this interface internally call functions like sleep or epoll_wait?

UpdateNetwork is a user thread and will not sleep, it sends and receives packets via the network thread. Only the network thread will use epoll_wait, which the user does not need to be concerned with.

What happens if SetResultFlag(2) and SetResultFlagForFail(0) are called simultaneously?

It's recommended not to use them together; SetResultFlagForFail and SetResultFlagForSuccess are recommended to be used in combination. If both SetResultFlag(2) and SetResultFlagForFail(0) are called simultaneously, the last setting will take effect.

Can Tcaplus directly use RepeatedPtrField as table fields for storage operations? Are there differences between using proto primitive types and structure types as basic Tcaplus table field storage methods?

Yes, there is no difference. Note that deeply nested structures may consume more CPU.

Does Tcaplus have an API for atomic operations?

Tcaplus has optimistic locking, supports single key transactions, TCAPLUS_CMD_INCREASE_REQ for auto-increment and auto-decrement operations, supports conditional updates under single key, and does not support other transactional operations.

In the FieldGet interface, if the dottedpaths field is empty, does it return all fields or only the key field with other common fields being empty?

FieldGet returns the required fields. If it is empty, only the key field is returned. To get all fields, you can use Get.

In the case of getbypartkey, if the business only wants to obtain the number of records that meet certain conditions, can it return only the number of records?

You can get one record by calling SetResultLimit(1), but to get the total number, you can use GetRecordMatchCount.

Can the ListNum of a list type table be modified?

The ListNum of a list type table can be increased but not decreased. ListNum is a required attribute and is generally set in the table Definition based on business size estimates

Does expanding the size in the table structure require modifying the field version?

No need, you can directly modify it. Size can only be increased, not decreased.

Is there a way to obtain the latest n records without knowing the index of the list table?

If the insertion is done in chronological order using addafter, you can directly retrieve from the tail using listgetall. SetResultLimit can specify the number of records. Alternatively, you can use a sortlist table, add a time field, and sort by time.

API access error: dir sign up fail, ret:-279, how to resolve?

Check if the business password is correct; check if the dir address is correct.

Does Tcaplus support fuzzy search and case insensitivity?

Supports fuzzy search, requires creating distributed index queries. Case sensitive.

Can there be multiple different TcaplusService::TcaplusServer in the same process?

Yes, hold multiple service API handles and initialize them separately.

Database Principles

Last updated：2024-10-15 17:53:55

How does the game server kick out an invalid tcaproxy (access layer) node?

TcaplusDB API supports disaster recovery in case of any tcaproxy exceptions. There are two main ways in which the API kicks out invalid tcaproxy processes:

1. The API physically considers that a tcaproxy is not available. The API sends heartbeat detection packets to all connected tcaproxy every second. If a game server does not receive the corresponding heartbeat return packets from tcaproxy within 10 seconds, the API will actively disconnect the TCP connection to the tcaproxy and actively connect the tcaproxy at the next onupdate.
2. API logically considers that a tcaproxy is not available. It calculates the request and response ratio of a tcaproxy every 10 seconds as a basis for judgment. The time out threshold for the API to a request packet is 3s. If it times out more than 3 times, the tcaproxy is considered unavailable and the request will not be sent to the tcaproxy. A getmetdata request is sent after 60s. If tcaproxy can correctly handle the getmetadata request , the API considers the tcaproxy available and the request is sent to the tcaproxy again.

If the game server finds that a tcaproxy is not available within 10s, it will not send data to the tcaproxy node.

How does the game server choose the tcaproxy (access layer) node?

The gameserver maintains a consistent Hash ring locally. Once a tcaproxy (access layer) node has been verified, it will be added to the Hash ring. If a tcaproxy (access layer) node reduces capacity or the TCP link between gameserver and tcaproxy (access layer) has been disconnected due to machine abnormalities, the gameserver will remove the tcaproxy (access layer) node from the Hash ring. The gameserver calculates hash values based on the primary key in the request (if it is a batchget request, it randomly selects a single tcaproxy (access layer) node), and then selects a single tcaproxy (access layer) node to send out on the consistent Hash ring.

Does TcaplusDB have the compression feature?

TcaplusDB has a compression feature using Google's Snappy compression algorithm. It includes protocol compression, i.e., compressing request/response packets between the game server and tcaproxy (access layer); and data compression, i.e., compressing data when storing it in tcapsvr (storage layer). If you want to save network traffic between the game server and tcaproxy, it is recommended to enable protocol compression using the SetCompressSwitch function in the TcaplusDB API. Enabling compression in the tcapsvr (storage layer) is also recommended to save disk space and improve disk IO performance. The CPU overhead for compression and decompression is manageable.

Is the TcaplusDB API thread-safe?

TcaplusDB API is non-thread-safe, mainly because components such as tlog and tdr are not thread-safe. It is recommended that a single thread uses a single API object and a single game region uses a single API object. If you need to interact across game regions, it is recommended to maintain multiple API objects with a single game server.

How does tcaproxy (access layer) achieve disaster recovery?

Tcaproxy (access layer) adopts a peer-to-peer design scheme, that is, all tcaproxy (access layer) nodes under a single game region contain routing information of all tables under a single game region. If a tcaproxy (access layer) fails, as long as the remaining tcaproxy (access layer) nodes are not overloaded, the game server will kick out the abnormal tcaproxy (access layer) node, which will not affect the use of game server. There is no single point of failure risk for the tcaproxy (access layer).

How does tcapsvr (storage layer) achieve disaster recovery?

The tcapsvr (storage layer) runs in a primary-secondary mode (primary tcapsvr and secondary tcapsvr). Primary and secondary tcapsvr synchronize data in real time, and are deployed at the different IDCs in the same city, ensuring that primary-secondary synchronization latency is less than 10 ms. If the secondary tcapsvr is abnormal, it will not affect the use of game server (if the read request offloading is not enabled, the requests of game server are processed by the primary tcapsvr. If the read request offloading is enabled, the secondary tcapsvr will assist in processing part of the read requests), and the DBA will rebuild the secondary tcapsvr; if the primary tcapsvr is abnormal, the secondary tcapsvr will perform failure recovery and the DBA will apply for a new machine to rebuild the secondary tcapsvr. There is no single point of failure risk for the tcapsvr (storage layer).

Does TcaplusDB have overload protection?

Both the access layer and the storage layer have process-level overload protection measures to ensure that services will not collapse during peak hours.

What is the principle of cold and hot data exchange for TcaplusDB?

TcaplusDB uses memory + SSD disk storage, a single engine file, and the first 1 GB is mapped in memory. The hot data is placed in the memory, the cold data is placed on the disk, and the LRU algorithm is used for hot and cold data exchange. The get operation of gameserver triggers the LRU swap-in operation. The LRU thread of tcapsvr (storage layer) is responsible for the LRU swap-out operation. Try to ensure that the hot data is stored in the memory, thus ensuring high cache hit rate and low single read and write latency.

How does tcaproxy (access layer) choose tcapsvr (storage layer)?

Each table defines a sub-table factor. If no sub-table is defined, the sub-table factor defaults to the primary key. Tcaproxy (access layer) selects the corresponding tcapsvr (storage layer) according to hash (sub-table factor) %1w, so the dispersion of the sub-table factor needs to be high.

What is the level of the lock in TcaplusDB?

The granularity of the lock in TcaplusDB is logging level.

If the number of shards is too large or too small, will it affect the performance of CRUD operations and engine tuning in TcaplusDB?

The impact is negligible. Online large tables are divided into at least hundreds of shards.

What factors are considered for the number of shards in a business?

Generally considered: Each shard has 1GB in memory, with about 500MB storing the Key and 300MB storing the Value. Having as many Keys in memory as possible can improve performance. Increasing the shard size to a maximum of 256GB per shard can affect recovery operations if the file is too large.

Does Tcaplus face penetration issues with high-frequency data reads?

Tcaplus has overload protection. When penetration causes service overload, some requests will be rejected at the access layer.

If a large number of requests flood in suddenly, will there be a queuing mechanism for tcaproxy and tcapsvr?

Yes, there will be a queuing mechanism. Exceeding the processing capacity will cause large latency and timeouts. For tcaproxy, it is related to the number of request connections and will distribute connections to the proxy for processing. For tcapsvr, it depends on whether the request data is dispersed. If concentrated on a few shards or keys, there will be queuing.

Traversal principle

Tcaplus supports full table traversal operations for GENERIC and LIST tables. It will initiate traversal tasks on each tcapsvr where the table is distributed based on sharding logic. Therefore, the overall traversal request performance is relatively low. It is recommended to set the traversal data to be read from the tcapsvr slave, which will not affect the services provided by the tcapsvr master, using the SetOnlyReadFromSlave(bool flag) interface.
If the data to be fetched each time is not much, consider first whether the local index can meet the requirements.
Once the traversal iterator starts, Tcaplus's tcapsvr will initiate an internal iterator to traverse and read the node data on the tree based on time slices. It finds the key first, then the value, and packages the traversed data into the buffer, returning it to the service API. The internal traversal logic of the service API will determine whether there is packet loss or duplicate packets based on whether the expected seq matches the received seq. Once it is confirmed that the received packets are as expected, the main thread can get a complete package when calling RecvResponse and then can parse a business record by calling FetchRecord.
Traversal can be interrupted and terminated at any time and can be conditionally recovered.
Overall, traversal within the service API adopts a ping-pong automatic trigger mechanism. After receiving the expected packet, it will automatically decide whether to initiate the next traversal request (e.g., if the backend storage server clearly returns a BUSY error, the traversal will be interrupted) until the traversal is clearly completed or the business stops the traversal.