

TencentDB for TcaplusDB

TcaplusDB RESTful APIs

Product Documentation



Copyright Notice

©2013–2026 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice



All trademarks associated with Tencent Cloud and its services are owned by the Tencent corporate group, including its parent, subsidiaries and affiliated companies, as the case may be. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Contents

TcaplusDB RESTful APIs

Description

Go

Java

PHP

Python

Downloading RESTful API Samples in Multiple Languages

TcaplusDB RESTful APIs

Description

Last updated: 2024-12-04 10:16:20

This document is the TcaplusDB RESTful API v1.0 User Manual.

Overview

TcaplusDB RESTful API enables you to remotely interact with TcaplusDB through HTTP requests. After sending an HTTP request where data is transferred in JSON format by calling the RESTful API, you will receive a response package in JSON format. You can send RESTful API requests in any programming language or tool to perform CRUD operations on data.

Preparations

Make sure that you have created a cluster in the [TcaplusDB Console](#) and obtained the corresponding cluster information, including `AppId` (cluster access ID), `ZoneId` (table group ID), and `AppKey` (cluster access password). Currently, TcaplusDB RESTful API only supports tables defined through protobuf.

Current Version

By default, the version of TcaplusDB RESTful API is v1.0.

Request

All API call requests are sent over HTTP, and all data is transferred in JSON format. Below is a typical format for access request URI:

```
http://{Tcaplus_REST_URL}/{Version}/apps/{AppId}/zones/{ZoneId}/tables/{
TableName}/records
```

- `Tcaplus_REST_URL`: TcaplusDB RESTful URL access point
- `Version`: TcaplusDB RESTful API version number, which is "ver1.0" by default
- `AppId`: cluster **access ID**
- `ZoneId`: table group **ID**
- `TableName`: table name

Sample code:

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_rest_test/records
```

HTTP header

Setting HTTP headers allows you to transfer additional information through requests and responses at the HTTP client.

Authentication

`x-tcaplus-pwd-md5`

Fill in this field with the calculated MD5 of your `app key`, which is used to authenticate the client's access to the data.

Calculate the MD5 of the string by running the `bash` command:

```
# echo -n "c3eda5f013f92c81dda7afc273cf82" | md5sum
879423b88d153cace7b31773a7f46039 -
```

Operation check

- `x-tcaplus-target` specifies TcaplusDB RESTful API request operations, including:
 - `Tcaplus.GetRecord` Gets record
 - `Tcaplus.AddRecord` Inserts record
 - `Tcaplus.SetRecord` Sets record
 - `Tcaplus.DeleteRecord` Deletes record
 - `Tcaplus.FieldGetRecord` Reads specified fields
 - `Tcaplus.FieldSetRecord` Sets specified fields
 - `Tcaplus.FieldIncRecord` Auto-increments specified fields
 - `Tcaplus.PartkeyGetRecord` Batch reads indexes
- `x-tcaplus-idl-type` specifies the TcaplusDB table type. Currently, only the protobuf (pb) type is supported.

Setting tag

- `x-tcaplus-data-version-check` Specifies TcaplusDB data version check policy use with `x-tcaplus-data-version`. It can be set as:
 - `x-tcaplus-data-version-check` specifies the TcaplusDB data version check policy to implement the optimistic locking feature, which is to be used with `x-tcaplus-data-version` tagging. Valid values include:
 - `1`: write operations can be performed only if the data version number at the client is identical to that at the storage layer. The version number will be increased by 1 each time this operation is performed.
 - `2`: the relationship between the client version number and the server version number is ignored. The client version number will be forcibly set at the storage layer.

- `3` : the relationship between the client version number and the server version number is ignored. The version number at the storage layer will be increased by 1 each time the write operation is performed.

These tags are only applicable to `Tcaplus.AddRecord` and `Tcaplus.SetRecord` .

- `x-tpcaplus-data-version` is used with `x-tpcaplus-data-version-check` to set the data version number of the client. Valid values include:
 - `version <= 0` means to ignore the version check policy.
 - `version > 0` means to specify the version number of the data record at the client.
- `x-tpcaplus-result-flag` sets whether the response contains the complete data policy. Valid values include:
 - `0` : the response only contains whether the request succeeded or failed.
 - `1` : the response contains the same values as the request does.
 - `2` : the response contains the latest values of all fields of the data that has been modified.
 - `3` : the response contains the value before the record is modified.

JSON request data

Request Data:

```
{
  "ReturnValues": "...", // The data you configure to be retained, which
  will arrive at TcaplusDB with the request and be sent back as-is
  "Record": {
    ... // For the format of data records, please see "API Samples".
  }
}
```

Response

JSON response data

The `GetRecord/SetRecord/AddRecord/DeleteRecord/FieldGetRecord/FieldSetRecord/FieldIn`
`cRecord` operation returns a single data entry. The JSON response data is as below:

Response Data

```
{
  "ErrorCode": 0, // Return code
  "ErrorMsg": "Succeed", // Return message
  "RecordVersion": 1, // Data version number
}
```

```

"ReturnValues": "...", // The data you configure to be retained, which
will arrive at TcaplusDB with the request and be sent back as-is
"Record": { // For the format of data records, please see "API
Samples".
    ...
}
}

```

The `PartkeyGetRecord` operation may return multiple data entries. The JSON response data is as below:

```

Response Data
{
  "ErrorCode": 0, // Return code
  "ErrorMsg": "Succeed", // Return message
  "MultiRecords": [ // Array of multiple data entries
    {"RecordVersion": 1, // Version number of each data entry
      "Record": {...} // For the format of data records, please see "API
Samples".
    },
    ...
  ],
  "RemainNum": 0, // Number of data entries not accessed yet
  "TotalNum": 5 // Total number of eligible data entries
}

```

Return codes

HTTP Status Code	Return Code	Return Message
200	0	Successful
400	-2579	Data deserialization error
401	-279	Authentication error
400	-11539	Invalid request For the detailed reason, please see the return message field.
400	-2067	Operation type mismatch

400	-1811	Invalid parameter
400	-5395	No primary key is set
400	-2323	Data serialization error
400	-7949	Invalid data version number This error generally occurs when the data version check policy is used in a write operation
400	-1293	The record already exists
404	261	The record does not exist
404	-34565	The index does not exist
500	-	Internal system error Please see the return message field

API Samples

Tcaplus.GetRecord

```
GET /ver1.0/apps/{APP_ID}/zones/{ZONE_ID}/tables/{TABLE_NAME}/records?
keys={JSONKeysObj}&select={JSONSelectObj}
```

This API is used to query a record by specifying its `key` information in a TcaplusDB PB table. You will get the entire record by performing this operation. However, you can set the `select` variable to specify the fields to be returned in the response; otherwise, the information of all fields will be displayed. If no data records exist, an error will be returned.

The `keys` variable must be specified in the URI, which indicates the values of all primary keys. The optional `select` variable indicates the name of the field whose value you want to display. You can specify the fields in the nested structure by separating the path with a dot, such as `pay.total_money`.

Note:

The variables in the request must be URL-encoded. Please encode the space in the URL as `%20` instead of `+`.

Name	Type	Value
x-tcaplus-target	String	Tcaplus.GetRecord

x-tcaplus-version	String	Tcaplus3.32.0
x-tcaplus-pwd-md5	String	MD5 of AppKey(Password)
x-tcaplus-idl-type	String	protobuf

Sample

URL

- If URL is not URL-encoded:

```
http://10.123.9.70:31002/ver1.0/apps/2/zones/1/tables/tb_example/records?keys={'region': 101, 'name': 'calvinshao', 'uin': 100}
```

- If URL is URL-encoded:

```
http://10.123.9.70:31002/ver1.0/apps/2/zones/1/tables/tb_example/records?keys=%7B%22region%22%3A%20101%2C%20%22name%22%3A%20%22calvinshao%22%2C%20%22uin%22%3A%20100%7D
```

HTTP request header

```
[  
  "x-tcaplus-target:Tcaplus.GetRecord",  
  "x-tcaplus-version:Tcaplus3.32.0",  
  "x-tcaplus-pwd-md5:c3eda5f013f92c81dda7afcdc273cf82",  
  "x-tcaplus-idl-type:protobuf"  
]
```

Response data

```
{  
  "ErrorCode": 0,  
  "ErrorMsg": "Succeed",  
  "RecordVersion": 1,  
  "Record": {  
    "name": "calvinshao",  
    "lockid": [  

```

```
50,  
60,  
70  
],  
"pay": {  
  "pay_times": 2,  
  "total_money": 10000,  
  "pay_id": 5,  
  "auth": {  
    "pay_keys": "adqwacsasafasda",  
    "update_time": 1528018372  
  }  
},  
"region": 101,  
"uin": 100,  
"is_available": true,  
"gamesvrid": 4099,  
"logintime": 404  
}  
}
```

Tcaplus.SetRecord

```
PUT /ver1.0/apps/{APP_ID}/zones/{ZONE_ID}/tables/{TABLE_NAME}/records
```

This API is used to set a record by specifying its `key` information. If the record already exists, an overwrite operation will be performed; otherwise, an insert operation will be performed.

The `SetRecord` operation supports setting the following values for `resultflag`:

- `0`: the response only contains whether the request succeeded or failed
- `1`: the response contains the same values as the request does
- `2`: the response contains the latest values of all fields of the data that has been modified
- `3`: the response contains the value before the record is modified

Name	Type	Value
x-tcaplus-target	String	Tcaplus.SetRecord
x-tcaplus-version	String	Tcaplus3.32.0

x-tcaplus-pwd-md5	String	MD5 of AppKey(Password)
x-tcaplus-result-flag	Int	2
x-tcaplus-data-version-check	Int	2
x-tcaplus-data-version	Int	-1
x-tcaplus-idl-type	String	protobuf

Sample

URL

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_example/records
```

HTTP request header

```
[  
  "x-tcaplus-target:Tcaplus.SetRecord",  
  "x-tcaplus-version:Tcaplus3.32.0",  
  "x-tcaplus-pwd-md5:c3eda5f013f92c81dda7afc273cf82",  
  "x-tcaplus-result-flag:2",  
  "x-tcaplus-data-version-check:2",  
  "x-tcaplus-data-version:-1",  
  "x-tcaplus-idl-type:Protobuf"  
]
```

JSON request data

```
{  
  "ReturnValues": "Send to tcaplus by calvinshao",  
  "Record": {  
    "name": "calvinshao",  
    "lockid": [  
      50,  
      60,  
      70,  
      80,  
      90,  
      100  
    ]  
  }  
}
```

```
],
  "pay": {
    "pay_times": 3,
    "total_money": 12000,
    "pay_id": 5,
    "auth": {
      "pay_keys": "adqwacsasafasda",
      "update_time": 1528018372
    }
  },
  "region": 101,
  "uin": 100,
  "is_available": false,
  "gamesvrid": 4099,
  "logintime": 404
}
```

JSON response data

```
{
  "ErrorCode": 0,
  "ErrorMsg": "Succeed",
  "RecordVersion": 1,
  "ReturnValues": "Send to tcaplus by calvinshao",
  "Record": {
    "name": "calvinshao",
    "lockid": [
      50,
      60,
      70,
      80,
      90,
      100
    ],
    "pay": {
      "pay_times": 3,
      "total_money": 12000,
      "pay_id": 5,
```

```

    "auth": {
      "pay_keys": "adqwacsasafasda",
      "update_time": 1528018372
    }
  },
  "region": 101,
  "uin": 100,
  "is_available": false,
  "gamesvrid": 4099,
  "logintime": 404
}
}

```

Tcaplus.AddRecord

```
POST /ver1.0/apps/{APP_ID}/zones/{ZONE_ID}/tables/{TABLE_NAME}/records
```

This API is used to insert a record by specifying its `key` information. If the record already exists, an error will be returned.

The `AddRecord` operation supports setting the following values for `resultflag`:

- `0`: the response only contains whether the request succeeded or failed
- `1`: the response contains the same values as the request does
- `2`: the response contains the latest values of all fields of the data that has been modified
- `3`: the response contains the value before the record is modified

Name	Type	Value
x-tcaplus-target	String	Tcaplus.SetRecord
x-tcaplus-version	String	Tcaplus3.32.0
x-tcaplus-pwd-md5	String	MD5 of AppKey(Password)
x-tcaplus-result-flag	Int	1
x-tcaplus-idl-type	String	protobuf

Sample

URL

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_example/records
```

HTTP request header

```
[  
  "x-tcaplus-target:Tcaplus.AddRecord",  
  "x-tcaplus-version:Tcaplus3.32.0",  
  "x-tcaplus-pwd-md5:c3eda5f013f92c81dda7afc273cf82",  
  "x-tcaplus-result-flag:1",  
  "x-tcaplus-idl-type:protobuf"  
]
```

JSON request data

```
{  
  "ReturnValues": "Send to tcaplus by calvinshao",  
  "Record": {  
    "name": "calvinshao",  
    "lockid": [  
      50,  
      60,  
      70  
    ],  
    "pay": {  
      "pay_times": 2,  
      "total_money": 10000,  
      "pay_id": 5,  
      "auth": {  
        "pay_keys": "adqwacsasafasda",  
        "update_time": 1528018372  
      }  
    },  
    "region": 101,  
    "uin": 100,  
    "is_available": true,  
    "gamesvrid": 4099,  
    "logintime": 404  
  }  
}
```

```
}
```

JSON response data

```
{
  "ErrorCode": 0,
  "ErrorMsg": "Succeed",
  "RecordVersion": 1,
  "ReturnValues": "Send to tcaplus by calvinshao",
  "Record": {
    "name": "calvinshao",
    "lockid": [
      50,
      60,
      70
    ],
    "pay": {
      "pay_times": 2,
      "total_money": 10000,
      "pay_id": 5,
      "auth": {
        "pay_keys": "adqwacsasafasda",
        "update_time": 1528018372
      }
    },
    "region": 101,
    "uin": 100,
    "is_available": true,
    "gamesvrid": 4099,
    "logintime": 404
  }
}
```

Tcaplus.DeleteRecord

```
DELETE /ver1.0/apps/{APP_ID}/zones/{ZONE_ID}/tables/{TABLE_NAME}/records
```

This API is used to delete a record by specifying its `key` information. If the data does not exist, an error will be returned.

The `DeleteRecord` operation supports setting the following values for `resultflag` :

- `0` : the response only contains whether the request succeeded or failed
- `1` : the response contains the same values as the request does
- `2` : the response contains the latest values of all fields of the data that has been modified
- `3` : the response contains the value before the record is modified

Name	Type	Value
x-tcaplus-target	String	Tcaplus.SetRecord
x-tcaplus-version	String	Tcaplus3.32.0
x-tcaplus-pwd-md5	String	MD5 of AppKey(Password)
x-tcaplus-result-flag	Int	1
x-tcaplus-idl-type	String	protobuf

Sample

URI

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_example/records
```

HTTP request header

```
[  
  "x-tcaplus-target:Tcaplus.DeleteRecord",  
  "x-tcaplus-version:Tcaplus3.32.0",  
  "x-tcaplus-pwd-md5:c3eda5f013f92c81dda7afcdc273cf82",  
  "x-tcaplus-result-flag:1",  
  "x-tcaplus-idl-type:protobuf"  
]
```

JSON request data

```
{  
  "ReturnValues": "Send to tcaplus by calvinshao",  
  "Record": {
```

```

"region": 101,
"name": "calvinshao",
"uin": 100
}
}

```

JSON response data

```

{
  "ErrorCode": 0,
  "Record": {},
  "RecordVersion": -1,
  "ReturnValues": "Send to tcaplus by calvinshao"
}

```

Tcaplus.FieldGetRecord

```

GET /ver1.0/apps/{APP_ID}/zones/{ZONE_ID}/tables/{TABLE_NAME}/records?
keys={JSONKeysObj}&select={JSONSelectObj}

```

This API is used to query a record by specifying its `key` information in a TcaplusDB pb table. This operation only queries and transfers the values of the field you specify through the `select` variable, which will reduce the network transfer traffic usage and is the biggest difference from the `GetRecord` operation. If the data record does not exist, an error will be returned.

The `keys` and `select` variables must be specified in the URI. `keys` specifies the values of all primary keys. `select` specifies the name of the field whose value you want to display. You can specify the fields in the nested structure by separating the path with a dot, such as `pay.total_money`.

Note:

The variables in the request must be URL-encoded. Please encode the space in the URL as `%20` instead of `+`.

Name	Type	Value
x-tcaplus-target	String	Tcaplus.FieldGetRecord
x-tcaplus-version	String	Tcaplus3.32.0
x-tcaplus-pwd-md5	String	MD5 of AppKey(Password)

x-tcaplus-idl-type	String	protobuf
--------------------	--------	----------

Sample

If URL is not URL-encoded

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_example/records?keys=
{'region': 101, 'name': 'calvinshao', 'uin': 100}&select=['gamesvrid',
'lockid', 'pay.auth.pay_keys', 'pay.total_money']
```

If URL is URL-encoded

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_example/records?
keys=%7B%22region%22%3A%20101%2C%20%22name%22%3A%20%22calvinshao%22%2C%2
0%22uin%22%3A%20100%7D&select=%5B%22gamesvrid%22%2C%20%22lockid%22%2C%20
%22pay.auth.pay_keys%22%2C%20%22pay.total_money%22%5D
```

HTTP request header

```
[
  "x-tcaplus-target:Tcaplus.FieldGetRecord",
  "x-tcaplus-version:Tcaplus3.32.0",
  "x-tcaplus-pwd-md5:c3eda5f013f92c81dda7afcdc273cf82",
  "x-tcaplus-idl-type:protobuf"
]
```

JSON response data

```
{
  "ErrorCode": 0,
  "ErrorMsg": "Succeed",
  "RecordVersion": 1,
  "Record": {
    "name": "calvinshao",
    "lockid": [
      50,
      60,
      70
    ],
  },
}
```

```

"pay": {
  "total_money": 10000,
  "auth": {
    "pay_keys": "adqwacsasafasda"
  }
},
"region": 101,
"uin": 100,
"gamesvrid": 4099
}
}

```

Tcaplus.FieldSetRecord

```
PUT /ver1.0/apps/{APP_ID}/zones/{ZONE_ID}/tables/{TABLE_NAME}/records
```

This API is used to modify a record by specifying its `key` information. Different from the `SetRecord` operation, it transfers and sets the values of only the specified fields instead of all fields, which reduces the network traffic usage. If the data record already exists, an update operation will be performed; otherwise, an error will be returned.

The `FieldSetRecord` operation supports setting the following values for `resultflag` :

- `0` : the response only contains whether the request succeeded or failed
- `1` : the response contains the same values as the request does

Name	Type	Value
x-tcaplus-target	String	Tcaplus.SetRecord
x-tcaplus-version	String	Tcaplus3.32.0
x-tcaplus-pwd-md5	String	MD5 of AppKey(Password)
x-tcaplus-result-flag	Int	2
x-tcaplus-data-version-check	Int	2
x-tcaplus-data-version	Int	-1
x-tcaplus-idl-type	String	protobuf

Sample

URL

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_example/records
```

HTTP request header

```
[  
  "x-tcaplus-target:Tcaplus.FieldSetRecord",  
  "x-tcaplus-version:Tcaplus3.32.0",  
  "x-tcaplus-pwd-md5:c3eda5f013f92c81dda7afc273cf82",  
  "x-tcaplus-result-flag:1",  
  "x-tcaplus-data-version-check:1",  
  "x-tcaplus-data-version:-1",  
  "x-tcaplus-idl-type:Protobuf"  
]
```

JSON request data

```
{  
  "ReturnValues": "aaaaaaaaaa",  
  "FieldPath": [ // Explicitly specify the path of the field to be  
updated  
    "gamesvrid",  
    "logintime",  
    "pay.total_money",  
    "pay.auth.pay_keys"  
  ],  
  "Record": { // Set the new value of the field to be updated  
    "name": "calvinshao",  
    "pay": {  
      "total_money": 17190,  
      "auth": {  
        "pay_keys": "bingo"  
      }  
    },  
    "region": 101,  
    "uin": 100,  
  }
```

```
"gamesvrid": 1719,  
"logintime": 1719  
}  
}
```

JSON response data

```
{  
  "ErrorCode": 0,  
  "ErrorMsg": "Succeed",  
  "RecordVersion": 3,  
  "ReturnValues": "aaaaaaaa",  
  "Record": {  
    "name": "calvinshao",  
    "pay": {  
      "total_money": 17190,  
      "auth": {  
        "pay_keys": "bingo"  
      }  
    }  
  },  
  "region": 101,  
  "uin": 100,  
  "gamesvrid": 1719,  
  "logintime": 1719  
}
```

Tcaplus.FieldIncRecord

```
PUT /ver1.0/apps/{APP_ID}/zones/{ZONE_ID}/tables/{TABLE_NAME}/records
```

This API is used to auto-increment the specified fields by specifying the `key` information of a record. It supports fields in `int32`, `int64`, `uint32`, and `uint64` fields and has similar characteristics as `FieldSetRecord`.

The `SetRecord` operation supports setting the following values for `resultflag`:

- `0`: the response only contains whether the request succeeded or failed
- `1`: the response contains the modified value of the specified field

Name	Type	Value
x-tcaplus-target	String	Tcaplus.SetRecord
x-tcaplus-version	String	Tcaplus3.32.0
x-tcaplus-pwd-md5	String	MD5 of AppKey(Password)
x-tcaplus-result-flag	Int	2
x-tcaplus-data-version-check	Int	2
x-tcaplus-data-version	Int	-1
x-tcaplus-idl-type	String	protobuf

Sample

URL

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_example/records
```

HTTP request header

```
[  
  "x-tcaplus-target:Tcaplus.FieldIncRecord",  
  "x-tcaplus-version:Tcaplus3.32.0",  
  "x-tcaplus-pwd-md5:c3eda5f013f92c81dda7afc273cf82",  
  "x-tcaplus-result-flag:1",  
  "x-tcaplus-data-version-check:1",  
  "x-tcaplus-data-version:-1",  
  "x-tcaplus-idl-type:Protobuf"  
]
```

JSON request data

```
{  
  "ReturnValues": "aaaaaaaaaa",  
  "Record": {  
    "name": "calvinshao",  
    "pay": {  
      "total_money": -1,  
    }  
  }  
}
```

```
"auth": {
  "update_time": -1
},
"region": 101,
"uin": 100,
"gamesvrid": 2,
"logintime": -2
}
```

JSON response data

```
{
  "ErrorCode": 0,
  "ErrorMsg": "Succeed",
  "RecordVersion": 9,
  "ReturnValues": "aaaaaaaa",
  "Record": {
    "name": "calvinshao",
    "pay": {
      "total_money": 11999,
      "auth": {
        "update_time": 921
      }
    }
  },
  "region": 101,
  "uin": 100,
  "gamesvrid": 4101,
  "logintime": 98
}
```

Tcaplus.PartkeyGetRecord

```
GET /ver1.0/apps/{APP_ID}/zones/{ZONE_ID}/tables/{TABLE_NAME}/records?
keys={JSONKeysObj}&select={JSONSelectObj}
```

This API is used to query multiple records by specifying the values of certain primary keys. This operation will return multiple data entries and display them by the field name specified by the `select` variable. The premise of this operation is that the specified primary key set must have an index created when the table is created; otherwise, an error will be returned.

The `keys` variable must be specified in the URI, which specifies the values of all primary keys. The optional `select` variable specifies the name of the field whose value you want to display. You can specify the fields in the nested structure by separating the path with a dot, such as `pay.total_money`. `limit` and `offset` are the parameters used to control partial record return.

Note:

The variables in the request must be URL-encoded. Please encode the space in the URL as `%20` instead of `+`. Please specify the index name to be accessed through `x-tcaplus-index-name` in the HEADER. The index name can be found in the table definition file.

Name	Type	Value
<code>x-tcaplus-target</code>	String	Tcaplus.GetRecord
<code>x-tcaplus-version</code>	String	Tcaplus3.32.0
<code>x-tcaplus-pwd-md5</code>	String	MD5 of AppKey(Password)
<code>x-tcaplus-idl-type</code>	String	protobuf
<code>x-tcaplus-index-name</code>	String	{index_name}

Sample

If URL is not URL-encoded

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_example/records?keys=
{'name': 'calvinshao', 'uin': 100}&select=['gamesvrid', 'lockid',
'pay.total_money', 'pay.auth.pay_keys']
```

If URL is URL-encoded

```
http://10.123.9.70/ver1.0/apps/2/zones/1/tables/tb_example/records?
keys=%7B%22name%22%3A%20%22calvinshao%22%2C%20%22uin%22%3A%20100%7D&sele
ct=%5B%22gamesvrid%22%2C%20%22lockid%22%2C%20%22pay.total_money%22%2C%20
%22pay.auth.pay_keys%22%5D
```

HTTP request header

```
[
  "x-tcapplus-target:Tcaplus.PartkeyGetRecord",
  "x-tcapplus-version:Tcaplus3.32.0",
  "x-tcapplus-pwd-md5:c3eda5f013f92c81dda7afcdc273cf82",
  "x-tcapplus-idl-type:protobuf"
  "x-tcapplus-index-name:index_name"
]
```

Response data

```
{
  "ErrorCode": 0,
  "ErrorMsg": "Succeed",
  "MultiRecords": [
    {
      "RecordVersion": 9,
      "Record": {
        "name": "calvinshao",
        "lockid": [
          50,
          60,
          70,
          80,
          90,
          100
        ],
        "pay": {
          "total_money": 11999,
          "auth": {
            "pay_keys": "adqwacsasafasda"
          }
        },
        "region": 101,
        "uin": 100,
        "gamesvrid": 4101
      }
    },
  ],
}
```

```
{
  "RecordVersion": 1,
  "Record": {
    "name": "calvinshao",
    "lockid": [
      50,
      60,
      70,
      80
    ],
    "pay": {
      "total_money": 10000,
      "auth": {
        "pay_keys": "adqwacsasafasda"
      }
    },
    "region": 102,
    "uin": 100,
    "gamesvrid": 4100
  },
  {
    "RecordVersion": 1,
    "Record": {
      "name": "calvinshao",
      "lockid": [
        60,
        70,
        80,
        90
      ],
      "pay": {
        "total_money": 10000,
        "auth": {
          "pay_keys": "adqwacsasafasda"
        }
      },
      "region": 103,
      "uin": 100,
      "gamesvrid": 4101
    }
  }
}
```

```
    }  
  }  
],  
  "RemainNum": 0,  
  "TotalNum": 3  
}
```

Go

Last updated: 2024-12-04 10:16:20

To help you use Go to manipulate TcaplusDB tables, we encapsulated TcaplusDB table operation APIs based on RESTful APIs, which support CRUD operations. This document describes how to use such APIs to manipulate TcaplusDB PB tables.

Preparations

1. Create a TcaplusDB table

Create a sample TcaplusDB table `game_players.proto`. For more information, please see [Creating Table](#).

2. Create a CVM instance

- Create a CVM instance to run the SDK demo. We recommend the following configuration: 2 CPU cores, 4 GB memory, and 50 GB disk space. The CVM instance needs to be in the same VPC as that of the TcaplusDB instance.
- Download the installation package of the RESTful API SDK for Go [here](#).

3. Prepare the Go environment

Run the following command to install the Go environment:

```
yum install -y golang
```

4. Compile the program

The SDK demo is compiled by using the `make` command. you can directly run `make build` in the `Makefile` file in the `src` directory. After the compilation, an executable file `example` will be generated. You can try out all sample APIs by directly running the file.

Directions

1. Configure table parameters

View the relevant information of the table created in the [TcaplusDB Console](#) and configure the demo as follows:

```
// TcaplusDB RESTful API connection parameters
const (
```

```
// Service access point, which is also the RESTful API
connection address of the cluster where the table resides. The default
port is 80
EndPoint = "http://172.xx.xx.12/"
// Application access ID, which is also the access ID of the
cluster where the table resides
AccessID = 310
// Application password, which is also the access password of
the cluster where the table resides
AccessPassword = "Tcaplus2020"
// Table group ID
TableGroupID = 1
// Table name
TableName = "game_players"
)
```

2. Create a table connection

Use `NewTcaplusClient` to specify the `EndPoint`, `AccessID`, and `AccessPassword` parameters to create `client`, which is a `TcaplusRestClient` object.

```
// Specify the `EndPoint`, `AccessID`, and `AccessPassword` parameters
to create `client`, which is a `TcaplusClient` object
client, err := tcaplus_client.NewTcaplusClient(EndPoint, AccessID,
AccessPassword)
if err != nil {
    fmt.Println(err.Error())
    return
}
}
```

3. Prepare sample table data

Insert data in JSON format. The definition is as follows:

```
// Call `AddRecord` to insert a record
// You can define `record` as a structure, map, or slice, which must be
convertible to JSON format
record := map[string]interface{}{
    "player_id": 10805514,
    "player_name": "Calvin",
}
```

```

    "player_email": "calvin@test.com",
    "game_server_id": 10,
    "login_timestamp": []string{"2019-12-12 15:00:00"},
    "logout_timestamp": []string{"2019-12-12 16:00:00"},
    "is_online": false,
    "pay": map[string]interface{}{
        "pay_id": 10101,
        "amount": 1000,
        "method": 1,
    },
}
status, resp, err := client.AddRecord(record,
tcaplus_client.RetAllLatestField, "userBuffer", TableGroupID, TableName)
if err != nil {
    fmt.Println(err.Error())
    return
}
}

```

API List

GetRecord

```

/**
    @brief This API is used to query a record based on the `Key`
    field. You can filter fields by `selectFiled`
    @param [IN] key `key` field information of the record to be
    queried. It can be a structure, map, or slice, so that it can be
    converted to JSON. It needs to be in the same type as that defined in
    `proto`
    @param [IN] selectFiled List of fields in the record to be
    queried. If you want to enter a nested structure, use a dotted value.
    `nil` indicates that all fields will be queried
    @param [IN] groupID Table group ID
    @param [IN] tableName Table name
    @retval(3) HTTP response code, HTTP response content, and error
    message
**/
func (c *TcaplusClient) GetRecord(key interface{}, selectFiled []string,
groupID int, tableName string) (string, []byte, error)

```

AddRecord

```
/**
    @brief This API is used to add a record to the table. If the
    record already exists, an error will be reported

    @param [IN] record Record to be added, which can be a structure,
    map, or slice, so that it can be converted to JSON. It needs to be in
    the same type as that defined in `proto`

    @param [IN] resultFlag Position flag of the returned value. Valid
    values:
        RetOnlySucOrFail: the response
        only contains whether the request succeeded or failed
        RetEqualReq: the response
        contains the same values as the request does
        RetAllLatestField: the response
        contains the latest values of all fields of the data that has been
        modified
        RetAllOldField: the response
        contains the value before the record is modified

    @param [IN] userBuffer Custom information, which will be returned
    as-is in the response message. If you do not need it, enter ""

    @param [IN] groupID Table group ID

    @param [IN] tableName Table name

    @retval(3) HTTP response code, HTTP response content, and error
    message
**/
func (c *TcaplusClient) AddRecord(record interface{}, resultFlag int,
userBuffer string, groupID int, tableName string) (string, []byte,
error)
```

SetRecord

```
/**
    @brief This API is used to update/insert a record in the table. If
    the record already exists, it will be updated; otherwise, it will be
    inserted

    @param [IN] record Record to be added, which can be a structure,
    map, or slice, so that it can be converted to JSON. It needs to be in
```

```

the same type as that defined in `proto`
    @param [IN] resultFlag    Position flag of the returned value. Valid
values:
                                RetOnlySucOrFail: the response
only contains whether the request succeeded or failed
                                RetEqualReq: the response
contains the same values as the request does
                                RetAllLatestField: the response
contains the latest values of all fields of the data that has been
modified
                                RetAllOldField: the response
contains the value before the record is modified
    @param [IN] versionPolicy    Policy for checking the record version
number, which is to be used with `version` to implement optimistic
locking. If you do not need it, set it to
`NoCheckDataVersionAutoIncrease`
                                CheckDataVersionAutoIncrease:
the record version number will be checked. The operation will succeed,
and the version number will auto-increment only if `version` is the same
as the version number on the server
                                NoCheckDataVersionOverwrite: the
record version number will not be checked, and it (`version`) will be
forcibly written to the server
                                NoCheckDataVersionAutoIncrease:
the record version number will not be checked, and the version number on
the server will auto-increment
    @param [IN] version    Record version number, which is used for
version number check. If you do not need the check, set it to `-1`
    @param [IN] userBuffer    Custom information, which will be returned
as-is in the response message. If you do not need it, enter ""
    @param [IN] groupID    Table group ID
    @param [IN] tableName    Table name
    @retval(3)    HTTP response code, HTTP response content, and error
message
**/
func (c *TcaplusClient) SetRecord(record interface{}, resultFlag int,
versionPolicy int, version int, userBuffer string, groupID int,
tableName string) (string, []byte, error)

```

DeleteRecord

```

/**
    @brief This API is used to delete a record. If the record does not
    exist, an error will be reported.

    @param [IN] record Record to be deleted, which only needs to
    contain the `key` field. It can be a structure, map, or slice, so that
    it can be converted to JSON. It needs to be in the same type as that
    defined in `proto`

    @param [IN] resultFlag Position flag of the returned value. Valid
    values:

        RetOnlySucOrFail: the response
        only contains whether the request succeeded or failed

        RetEqualReq: the response
        contains the same values as the request does

        RetAllLatestField: the response
        contains the latest values of all fields of the data that has been
        modified

        RetAllOldField: the response
        contains the value before the record is modified

    @param [IN] userBuffer Custom information, which will be returned
    as-is in the response message. If you do not need it, enter ""

    @param [IN] groupID Table group ID

    @param [IN] tableName Table name

    @retval(3) HTTP response code, HTTP response content, and error
    message
**/
func (c *TcaplusClient) DeleteRecord(record interface{}, resultFlag int,
userBuffer string, groupID int, tableName string) (string, []byte,
error)

```

FieldGetRecord

```

/**
    @brief This API is used to query a record based on specified
    fields. The `key` field is used to query the record, and `selectFiled`
    is used to filter fields

```

```

    @note    The difference between this API and `GetRecord` lies in that
    `GetRecord` queries the entire record first and then filters the fields
    by `selectFiled`, while `FieldGetRecord` filters the fields on the
    server, which has a lower traffic load

```

```

    @param [IN] key    `key` field information of the record to be
    queried. It can be a structure, map, or slice, so that it can be
    converted to JSON. It needs to be in the same type as that defined in
    `proto`

```

```

    @param [IN] selectFiled    List of fields in the record to be
    queried, which cannot be empty. If you want to enter a nested structure,
    use a dotted value

```

```

    @param [IN] groupID    Table group ID

```

```

    @param [IN] tableName    Table name

```

```

    @retval(3)    HTTP response code, HTTP response content, and error
    message

```

```

    **/

```

```

func (c *TcaplusClient) FieldGetRecord(key interface{}, selectFiled
[]string, groupID int, tableName string) (string, []byte, error)

```

FieldSetRecord

```

/**

```

```

    @brief    This API is used to update a record. If the record does not
    exist, an error will be reported

```

```

    @param [IN] record    Record to be added, which can be a structure,
    map, or slice, so that it can be converted to JSON. It needs to be in
    the same type as that defined in `proto`

```

```

    @param [IN] setField    List of fields to be updated, which cannot be
    empty. If you want to enter a nested structure, use a dotted value

```

```

    @param [IN] resultFlag    Position flag of the returned value. Valid
    values:

```

```

        RetOnlySucOrFail: the response
        only contains whether the request succeeded or failed

```

```

        RetEqualReq: the response
        contains the same values as the request does

```

```

    @param [IN] versionPolicy    Policy for checking the record version
    number, which is to be used with `version` to implement optimistic

```

locking. If you do not need it, set it to

```
`NoCheckDataVersionAutoIncrease`
```

`CheckDataVersionAutoIncrease`:

the record version number will be checked. The operation will succeed, and the version number will auto-increment only if ``version`` is the same as the version number on the server

`NoCheckDataVersionOverwrite`: the

record version number will not be checked, and it (``version``) will be forcibly written to the server

`NoCheckDataVersionAutoIncrease`:

the record version number will not be checked, and the version number on the server will auto-increment

`@param [IN] version` Record version number, which is used for version number check. If you do not need the check, set it to ``-1``

`@param [IN] userBuffer` Custom information, which will be returned as-is in the response message. If you do not need it, enter ""

`@param [IN] groupID` Table group ID

`@param [IN] tableName` Table name

`@retval (3)` HTTP response code, HTTP response content, and error message

```
**/
```

```
func (c *TcaplusClient) FieldSetRecord(record interface{}, setField []string, resultFlag int, versionPolicy int, version int, userBuffer string, groupID int, tableName string) (string, []byte, error)
```

FieldIncRecord

```
/**
```

`@brief` This API is used to auto-increment/decrement an integer field in a record. This command word supports fields only in ``int32``, ``int64``, ``uint32``, or ``uint64`` type

`@param [IN] record` Record to be updated. If the field value in the record is positive, it indicates auto-increment, and this field will be incremented; if the field value in the record is negative, it indicates auto-decrement, and this field will be decremented

`@param [IN] resultFlag` Position flag of the returned value. Valid values:

```

        RetOnlySucOrFail: the response
only contains whether the request succeeded or failed
        RetEqualReq: the response
contains the same values as the request does
        @param [IN] versionPolicy Policy for checking the record version
number, which is to be used with `version` to implement optimistic
locking. If you do not need it, set it to
`NoCheckDataVersionAutoIncrease`
        CheckDataVersionAutoIncrease:
the record version number will be checked. The operation will succeed,
and the version number will auto-increment only if `version` is the same
as the version number on the server
        NoCheckDataVersionOverwrite: the
record version number will not be checked, and it (`version`) will be
forcibly written to the server
        NoCheckDataVersionAutoIncrease:
the record version number will not be checked, and the version number on
the server will auto-increment
        @param [IN] version Record version number, which is used for
version number check. If you do not need the check, set it to `-1`
        @param [IN] userBuffer Custom information, which will be returned
as-is in the response message. If you do not need it, enter ""
        @param [IN] groupID Table group ID
        @param [IN] tableName Table name
        @retval(3) HTTP response code, HTTP response content, and error
message
    **/
func (c *TcaplusClient) FieldIncRecord(record interface{}, resultFlag
int, versionPolicy int, version int, userBuffer string, groupID int,
tableName string) (string, []byte, error)

```

PartkeyGetRecord

```

/**
    @brief This API is used to query records in batches with an index
    @param [IN] key Record to be queried, which needs to be set to the
`key` field in the index. It can be a structure, map, or slice, so that

```

```

it can be converted to JSON. It needs to be in the same type as that
defined in `proto`
    @param [IN] indexName    Index name
    @param [IN] selectFiled  List of fields in the record to be
queried. If you want to enter a nested structure, use a dotted value.
`nil` indicates that all fields will be queried
    @param [IN] limit       Maximum number of records to be returned in
batches, which will be valid if the value is greater than 0
    @param [IN] offset      Offset of the records to be returned in
batches, which will be valid if the value is greater than or equal to 0
    @param [IN] groupID     Table group ID
    @param [IN] tableName   Table name
    @retval(3)             HTTP response code, HTTP response content, and error
message
**/
func (c *TcaplusClient) PartKeyGetRecord(key interface{}, indexName
string, selectFiled []string, limit int, offset int, groupID int,
tableName string) (string, []byte, error)

```

For the `PartKeyGetRecord` API, the maximum size of a packet returned by one request is `256 KB`, and the `limit` value is subject to the size of one single record. We recommend you set it as follows:

- If the size of one single record is smaller than 256 KB, you can set `limit` to `256 KB / record size`; for example, if the record size is 10 KB, we recommend you set `limit` to a value between 20 and 25.
- If the size of one single record is greater than or equal to 256 KB, set `limit` to 1, i.e., only one record will be returned by one request.

In scenarios where `limit` and `offset` are set, to get the full data based on the index key, you need to judge whether the obtained data is complete based on the `TotalNum` and `RemainNum` flags returned in the response packet.

Java

Last updated: 2024-12-04 10:16:20

To help you use Java to manipulate TcaplusDB tables, we encapsulated the SDK for Java based on RESTful APIs. This document describes how to use RESTful APIs for Java to manipulate TcaplusDB PB tables (based on the protobuf protocol).

Preparations

1. Create a TcaplusDB table

Create a sample TcaplusDB table `game_players.proto`. For more information, please see [Creating Table](#).

2. Create a CVM instance

- Create a CVM instance to run the SDK demo. We recommend the following configuration: 1 CPU core, 2 GB memory, and CentOS 7. The CVM instance needs to be in the same VPC as that of the TcaplusDB instance.
- Download the installation package of the RESTful API SDK for Java [here](#).

3. Prepare the Java environment

The SDK is dependent on Java 1.8 or above. If the OS of the CVM instance is CentOS 7 or above, you can simply run `yum install -y java`.

4. API description

Currently, the SDK supports eight APIs, which are as detailed below:

API Name	Description
addRecord	Adds TcaplusDB record, which must contain the values of all primary key fields
getRecord	Queries TcaplusDB record. A primary key field must be specified
setRecord	Updates TcaplusDB record. A primary key field must be specified for the update. If the record does not exist, a new one will be inserted
deleteRecord	Deletes TcaplusDB record. A primary key field must be specified

fieldSetRecord	Updates specified field. A primary key field must be specified
fieldGetRecord	Gets specified field. A primary key field must be specified
fieldIncRecord	Updates specified field, which can be used only to auto-increment/decrement numeric fields. A primary key field must be specified
partkeyGetRecord	Gets one or more TcaplusDB records based on specified index name and corresponding index field value. The corresponding index field value must be specified

Directions

Upload the entire SDK package `tcaplusdb-restapi-java-sdk-1.0-assembly.zip` to the CVM instance directory. For more information, please see [Copying Local File to CVM](#). After decompression, the directory name will be `tcaplusdb-restapi-java-sdk-1.0`.

1. Prepare configuration

The configuration file is in `tcaplusdb-restapi-java-sdk-1.0/conf/config.properties`, and its content is as follows:

```
#replace with your table endpoint
endpoint=http://10.xx.xx.16

#replace with your table access id
access_id=60

#replace with your table access password
access_password=Tcaxxx19

#replace with your table group id
table_group_id=1

#replace with your table name
table_name=game_players

#optional, configure the returning fields for GetRecord API, multiple
fields with comma delimiter
get_select_fields=
```

```

#required, configure the returning fields for FieldGetRecord API
field_get_select_fields=game_server_id,is_online

#optional, configure the returning fields for PartkeyGetRecord API
part_key_get_select_fields=game_server_id,is_online

#required, configure the updating fields for FieldSetRecord API
field_set_fields=game_server_id,pay.amount

#required, configure the index keys for PartkeyGetRecord API
part_key_index_keys=player_id,player_name

#required, configure the index name for PartkeyGetRecord API
part_key_index_name=index_1

#optional, configure the limit number of records per request to return
for PartkeyGetRecord API, value range: >0
part_key_limit = 10

#optional, configure the offset position to return for PartkeyGetRecord
API, value range: >=0
part_key_offset = 0

```

2. Prepare data

The demo provides separately prepared testing data for each RESTful API, which is stored in the `tcaplus` `db-restapi-java-sdk-1.0/conf` directory as JSON files.

Filename	File Description
add_data.json	Data for <code>addRecord</code> API, which is used to add a record
set_data.json	Data for <code>updateRecord</code> API, which is used to update a record value
delete_data.json	Data for <code>deleteRecord</code> API, which is used to delete the primary key value of a record
get_data.json	Data for <code>getRecord</code> API, which is used to get the primary key value of a record
field_get_data.json	Data for <code>fieldGetRecord</code> API, which is used to get the values of specified fields

field_set_data.json	Data for <code>fieldSetRecord</code> API, which is used to update the values of specified fields
field_inc_data.json	Data for <code>fieldIncRecord</code> API, which is used to increase the values of specified numeric fields
partkey_get_data.json	Data for <code>partkeyGetRecord</code> API, which is used to get a record value based on the index

3. Execute the script

The main execution scripts are in `tcaplusdb-restapi-java-sdk-1.0/bin/run.sh` and can be run as follows:

```
# Add a record
sh bin/run.sh add
# Get a record
sh bin/run.sh get
# Update a record
sh bin/run.sh set
# Delete a record
sh bin/run.sh delete
# Update specified fields
sh bin/run.sh field_set
# Get specified fields
sh bin/run.sh field_get
# Update the values of specified fields (in numeric type)
sh bin/run.sh field_inc
# Get a record based on the index
sh bin/run.sh partkey_get
```

API Data Description

addRecord

This API is used to add a record. The inserted data is in JSON format as shown below:

```
{"player_id":1,"player_name":"test","player_email":"test@email.com",
"game_server_id":2,"login_timestamp":[],"logout_timestamp":
```

```
[], "is_online": true, "pay": {"pay_id": 1, "amount": 20, "method": 3}}
```

getRecord

This API is used to query a record by specifying the primary key field. The data is in JSON format as shown below:

```
{"player_id": 1, "player_name": "test", "player_email": "test@email.com"}
```

The query API also allows you to specify the list of fields to be returned. The `get_select_fields` configuration item defined in the `config.properties` configuration file is used to specify the names of the fields to be returned. If it is left empty, all fields of the record will be returned by default; otherwise, the values of the specified fields will be returned.

setRecord

This API is used to update a record, and the record to be updated must contain a primary key field. The data is in JSON format as shown below:

```
{"player_id": 1, "player_name": "test", "player_email": "test@email.com",  
"game_server_id": 2, "login_timestamp": [], "logout_timestamp":  
[], "is_online": false, "pay": {"pay_id": 1, "amount": 30, "method": 3}}
```

deleteRecord

This API is used to delete a record. The entered data must contain a primary key field. The data is in JSON format as shown below:

```
{"player_id": 1, "player_name": "test", "player_email": "test@email.com"}
```

fieldSetRecord

This API is used to update the values of specified fields. The entered data must contain a primary key field. The data is in JSON format as shown below:

```
{"player_id": 1, "player_name": "test", "player_email": "test@email.com",  
"game_server_id": 3, "is_online": false, "login_timestamp":  
[], "logout_timestamp": [], "pay": {"pay_id": 1, "amount": 40, "method": 3}}
```

At the same time, you need to specify the names of the fields to be updated in the `field_set_fields` configuration item defined in the `config.properties` configuration file. Multiple fields should be separated with commas, and the value **cannot be empty**.

fieldGetRecord

This API is used to get the values of specified fields. The entered data must contain a primary key field. The data is in JSON format as shown below:

```
{"player_id":1,"player_name":"test","player_email":"test@email.com"}
```

At the same time, you need to specify the names of the fields to be obtained in the `field_get_select_fields` configuration item defined in the `config.properties` configuration file, and the value **cannot be empty**.

fieldIncRecord

This API is used to update the values of specified fields. Different from the `fieldSetRecord` API, it can update field values only in numeric type, such as increasing or decreasing a value. The entered data must contain a primary key field and the numeric fields to be updated. The data is in JSON format as shown below:

```
{"player_id":1,"player_name":"test","player_email":"test@email.com",  
"pay":{"amount":50}}
```

Suppose the initial value of `pay.amount` is 100, and the value of the input parameter in the request is 50, then the final value will be 150.

partkeyGetRecord

This API is used to get the corresponding record based on the specified index. Multiple records may be matched, and you can specify the fields to be returned. The entered data must contain an index field. The data is in JSON format as shown below:

```
{"player_id":1,"player_name":"test","player_email":"test@email.com"}
```

At the same time, you need to specify the corresponding index name and index field names in the `part_key_index_name` and `part_key_index_keys` configuration items defined in the `config.properties` configuration file, respectively. **The two configuration items cannot be empty**.

To return specified fields, you can configure `part_key_get_select_fields` in the configuration file; otherwise, leave it empty.

For this API, the maximum size of a packet returned by one request is `256 KB`, and the `limit` value is subject to the size of one single record. We recommend you set it as follows:

- If the size of one single record is smaller than 256 KB, you can set `limit` to `256 KB / record size`; for example, if the record size is 10 KB, we recommend you set `limit` to a value between 20 and 25.
- If the size of one single record is greater than or equal to 256 KB, set `limit` to 1, i.e., only one record will be returned by one request.

In scenarios where `limit` and `offset` are set, to get the full data based on the index key, you need to judge whether the obtained data is complete based on the `TotalNum` and `RemainNum` flags returned in the response packet.

You can set `limit` and `offset` in the following steps:

```
# Step 1. Prepare batch demo data and set the values of `limit` and
`offset` as instructed in `conf/batch_add_data.json`.
part_key_limit = 2
part_key_offset = 0

# Step 2. Add demo data in batches. Run the `batch_add` command to add
three records.
sh bin/run.sh batch_add

# Step 3. Call the `partkey_get` command, and `partkeyGetRecord` will be
executed twice: in the first execution, `limit` and `offset` are not
set; in the second execution, `limit` and `offset` are set. Check the
differences between the two execution results.
sh bin/run.sh partkey_get

# Step 4. The response data with `limit` and `offset` not set is as
follows, which contains four data entries in total:
{"MultiRecords":[{"RecordVersion":3,"Record":
{"game_server_id":3,"is_online":true,"player_email":"test@email.com","p
layer_id":1,"player_name":"test"}},{"RecordVersion":1,"Record":
{"game_server_id":3,"is_online":true,"player_email":"test1@email.com","p
layer_id":1,"player_name":"test"}},{"RecordVersion":1,"Record":
{"game_server_id":4,"is_online":true,"player_email":"test2@email.com","p
layer_id":1,"player_name":"test"}},{"RecordVersion":1,"Record":
{"game_server_id":5,"is_online":true,"player_email":"test3@email.com","p
layer_id":1,"player_name":"test"}]},{"TotalNum":4,"RemainNum":0,"ErrorCod
e":0,"ErrorMsg":"Succeed"}
```

Step 5. The response data with `limit` and `offset` set is as follows, which contains only two data entries subject to the `limit` value:

```
{"MultiRecords":[{"RecordVersion":3,"Record":{"game_server_id":3,"is_online":true,"player_email":"test@email.com","player_id":1,"player_name":"test"}},{"RecordVersion":1,"Record":{"game_server_id":3,"is_online":true,"player_email":"test1@email.com","player_id":1,"player_name":"test"}}],"TotalNum":4,"RemainNum":2,"ErrorCode":0,"ErrorMsg":"Succeed"}
```

PHP

Last updated: 2024-12-04 10:16:20

Preparations

1. Create a TcaplusDB table

Create a sample TcaplusDB table `game_players.proto`. For more information, please see [Creating Table](#).

2. Create a CVM instance

- Create a CVM instance to run the SDK demo. We recommend the following configuration: 1 CPU core, 2 GB memory, and CentOS 7. The CVM instance needs to be in the same VPC as that of the TcaplusDB instance.
- Download the installation package of the RESTful API SDK for PHP [here](#).

3. Prepare the PHP environment

PHP is not installed on the new instance. You need to run `yum install php` to install it on the default version (PHP 5.4.16). The PHP version must be above 5.3.

Directions

1. Configuration modification

All basic configuration items are in `config.php` in the SDK installation directory, and you need to modify them based on the information of the table applied for.

- Connection configuration: modify parameters such as `url`, `endpoint`, `table_group_id`, `access_id`, and `access_passwd` to the corresponding values. For more information on how to get the connection information, please see [Getting Access Point Information](#).
- Default data source: all variables in this section are in uppercase. It can be used as the data source of specific API functions in `demo.php`, so the source information can be modified here in a unified manner.

2. Call method

To facilitate the feature testing of individual APIs, all SDK API functions are built as separate programs in TcaplusDB, which correspond to the eight API functions in the demo, and their data sources are managed separately from the program code.

The data sources of all API functions are from two places: default data sources are uniformly stored in `config.php`, and the own data sources of each API function are stored in the `data` subdirectory. You can modify a data file to run the corresponding API function, which helps eliminate your need to define data

sources in the program code during batch operations. If the data file does not exist, the default data source will be loaded, where all data is organized in the form of list and can be loaded in batches.

Program files in `sample` can be executed by running a command similar to the one below:

```
php -f <Specific API function file>
```

3. API functions

All APIs are encapsulated as functions in `demo.php`. There are eight API functions in total, all of which can load the data source from the configuration for batch operations. The specific functions are as detailed below:

API Function Name	Feature
GetRecord	Gets table record
AddRecord	Inserts table record
SetRecord	Updates table record
DeleteRecord	Deletes record
FieldGetRecord	Gets an attribute field of specified record
FieldSetRecord	Updates an attribute field value of specified record
FieldIncRecord	Updates the numeric value of an attribute field of specified record, such as increasing/decreasing a value in numeric type
PartKeyGetRecord	Gets an attribute field value of specified record based on specified index

SDK API List

add

This API is used to insert a record into a table. If the record already exists, an error will be reported.

```
@param $table_group_id `table_group_id` of target table, which is required
@param $table_name `table_name` of target table, which is required
@param $record Array of target records to be inserted, which is required
```

```

@param $ReturnValues Custom value to be returned, which is optional
@param $resultflag Content to be returned in response packet, which is
optional. Valid values:
    0: the response only contains whether the request succeeded
or failed
    1: the response contains the same values as the request does
    2: the response contains the latest values of all fields of
the data that has been modified
    3: the response contains the value before the record is
modified

public function add($table_group_id, $table_name, array $record,
$returnValues = 'TcaplusDB', $resultflag = 1)

```

get

This API is used to query a record based on the primary key. It allows you to specify the list of attributes fields to be returned as detailed in `$params["select"]`.

```

@param $table_group_id Game region ID, which is required
@param $table_name Table name, which is required
@param $params Required
    $params["select"] Array of the fields to be queried, which is
optional
    $params ["keys"] Primary key field for querying target record,
which is required
    $params ["limit"] Limit on the number of records to be returned,
which is optional
    $params ["offset"] Offset of the number of records to be
returned, which is optional

public function get($table_group_id, $table_name, $params)

```

fieldGet

This API is used to query and return the value of a specified field based on the primary key. You can specify the list of fields as detailed in `$param["select"]`, which cannot be empty.

```

@param $table_group_id Game region ID, which is required
@param $table_name Table name, which is required

```

```

@param $params Required
    $params\["select"\] Array of the fields to be queried, which is
required
    $params\["keys"\] Primary key field for querying target record,
which is required
    $params\["limit"\] Limit on the number of records to be returned,
which is optional
    $params\["offset"\] Offset of the number of records to be
returned, which is optional

public function fieldGet($table_group_id, $table_name, $params)

```

partKeyGet

This API is used to query and return one or multiple records based on the index defined in the table. It can return the list of specified fields and allow you to use the `limit` and `offset` parameters to control the number of records in the returned packet.

```

@param $table_group_id Game region ID, which is required
@param $table_name Table name, which is required
@param $params Required
    $params\["select"\] Array of the fields to be queried, which is
optional
    $params\["index"\] Name of the index for query, which is required
    $params\["keys"\] Primary key field for querying target record,
which is required
    $params\["limit"\] Limit on the number of records to be returned,
which is optional. Default value: -1
    $params\["offset"\] Offset of the number of records to be
returned, which is optional

public function partKeyGet($table_group_id, $table_name, $params)

```

For the `partkeyGet` API, the maximum size of a packet returned by one request is `256 KB`, and the `limit` value is subject to the size of one single record. We recommend you set it as follows:

- If the size of one single record is smaller than 256 KB, you can set `limit` to 256 KB / **record size**; for example, if the record size is 10 KB, we recommend you set `limit` to a value between 20 and 25.
- If the size of one single record is greater than or equal to 256 KB, set `limit` to 1, i.e., only one record will be returned by one request.

The following are sample responses for different `limit` and `offset` settings:

```
# If `limit=-1, offset=0` is set in the request
{"MultiRecords":[{"RecordVersion":1,"Record":{"pay":{"amount":1000,"pay_id":10101},"player_email":"kosh@test.com","player_id":1,"player_name":"kosh"}}, {"RecordVersion":1,"Record":{"pay":{"amount":1000,"pay_id":10101},"player_email":"kosh@163.com","player_id":1,"player_name":"kosh"}}, {"RecordVersion":1,"Record":{"pay":{"amount":1000,"pay_id":10101},"player_email":"kosh@gmail.com","player_id":1,"player_name":"kosh"}}, {"RecordVersion":1,"Record":{"pay":{"amount":1000,"pay_id":10101},"player_email":"kosh@126.com","player_id":1,"player_name":"kosh"}}],"TotalNum":4,"RemainNum":0,"ErrorCode":0,"ErrorMsg":"Succeed"}

# If `limit=2, offset=0` is set in the request
{"MultiRecords":[{"RecordVersion":1,"Record":{"pay":{"amount":1000,"pay_id":10101},"player_email":"kosh@test.com","player_id":1,"player_name":"kosh"}}, {"RecordVersion":1,"Record":{"pay":{"amount":1000,"pay_id":10101},"player_email":"kosh@163.com","player_id":1,"player_name":"kosh"}}],"TotalNum":4,"RemainNum":2,"ErrorCode":0,"ErrorMsg":"Succeed"}
```

As shown above, if `limit` and `offset` are set, the number of entries in the returned result will be the same as that set in `limit`.

To get the full index data, you can judge whether the obtained data is complete based on the `RemainNum` and `TotalNum` flags returned in the response packet.

set

This API is used to update or insert a record. If the record already exists, it will be updated; otherwise, it will be inserted.

```
@param $table_group_id Game region ID, which is required
@param $table_name Table name, which is required
@param $record Target record to be set, which is required. This operation involves insertion and modification semantics
@param $ReturnValues Check string, which is optional. The server will return the same string
@param $resultflag Content mode in response package, which is optional. Default value: 1. Valid values:
```

```

    0: the response only contains whether the request succeeded
or failed
    1: the response contains the same values as the request does
    2: the response contains the latest values of all fields of
the data that has been modified
    3: the response contains the value before the record is
modified

public function set($table_group_id, $table_name, array $record,
$returnValues = 'TcaplusDB', $resultflag = 1)

```

fieldSet

This API is used to update the value of a specified field based on the primary key. You can specify the field to be updated as detailed in `$field_path`.

```

@param $table_group_id Game region ID, which is required
@param $table_name Table name, which is required
@param $record Target record to be set, which is required. This
operation involves insertion and modification semantics
@param $field_path Array of field names (paths) to be set, which is
required. You can use a dotted path to specify nested fields
@param $returnValues Check string, which is optional. The server will
return the same string
@param $resultflag Content mode in response package, which is
optional. Default value: 1. Valid values:
    0: the response only contains whether the request succeeded
or failed
    1: the response contains the same values as the request does
    2: the response contains the latest values of all fields of
the data that has been modified
    3: the response contains the value before the record is
modified

public function fieldSet($table_group_id, $table_name, array $record,
$field_path, $returnValues = 'TcaplusDB', $resultflag = 1)

```

fieldInc

This API is used to update (auto-increment/decrement) the value of a specified numeric field based on the primary key. For example, if the original value of `pay.method` is 200, the requested value is 50, then the final value will be 250; if the requested value is -50, then the final value will be 150.

Note:

Auto-decrement only takes effect for signed type such as `int32` and `int64`.

```
@param $table_group_id Game region ID, which is required
@param $table_name Table name, which is required
@param $record Record to be auto-incremented/decremented, which is
required. It must contain the primary key, and the target record must be
in integer type
@param $ReturnValues Check string, which is optional. The server will
return the same string
@param $resultflag Content mode in response package, which is
optional. Default value: 1. Valid values:
    0: the response only contains whether the request succeeded
or failed
    1: the response contains the same values as the request does
    2: the response contains the latest values of all fields of
the data that has been modified
    3: the response contains the value before the record is
modified
@param $dataVersion Version number, which is optional
@param $dataVersionCheck Data version check policy, which is optional.
Valid values:
    1: data will be written only if the version numbers are the same
    2: the version number will not be checked. The client version number
will be forcibly set at the storage layer.
    3: the version number will not be checked. The version number at the
storage layer will be increased by 1 each time the write operation is
performed. This is the default value

public function fieldInc($table_group_id, $table_name, array $record,
$returnValues = 'TcaplusDB', $resultflag = 2, $dataVersion = -1,
$dataVersionCheck = 3)
```

delete

This API is used to delete a record of a specified primary key.

```
@param $table_group_id Game region ID, which is required
@param $table_name Table name, which is required
@param $record Array of target records to be deleted, which is
required. You only need to set a primary key field
@param $ReturnValues Check string, which is optional. The server will
return the same string
@param $resultflag Content mode in response package, which is
optional. Default value: 1. Valid values:
    0: the response only contains whether the request succeeded
or failed
    1: the response contains the same values as the request does
    2: the response contains the latest values of all fields of
the data that has been modified
    3: the response contains the value before the record is
modified

public function delete($table_group_id, $table_name, array $record,
$returnValues = 'TcaplusDB', $resultflag = 1)
```

demo.php Usage

To help you quickly try out TcaplusDB SDK features, all APIs have been encapsulated as eight functions in `demo.php`. You can run the corresponding function as needed and comment out unneeded ones by running the following command:

```
php -f "demo.php"
```

Python

Last updated: 2024-12-04 10:16:20

This SDK is encapsulated for Python based on RESTful APIs and used to perform CRUD operations on TcaplusDB PB tables.

Preparations

1. Create a TcaplusDB table

Create a sample TcaplusDB table `game_players.proto`. For more information, please see [Creating Table](#).

2. Create a CVM instance

- Create a CVM instance to run the SDK demo. We recommend the following configuration: 2 CPU cores, 4 GB memory, and 50 GB disk space. The CVM instance needs to be in the same VPC as that of the TcaplusDB instance.
- Download the installation package of the RESTful API SDK for Python [here](#).

Directions

1. Specify the `endpoint`, `access_id`, and `access_passwd` parameters to create `client`, which is a `TcaplusRestClient` object.
2. Use the `SetTargetTable` method of the `client` object to specify the table to be accessed.
3. Call an API function to send a data access request.

```
# Create a `TcaplusRestClient` object
#endpoint: restful API, such as http://x.x.x.x:80
client = TcaplusRestClient(endpoint, access_id, access_passwd)
# Use `SetTargetTable` to specify the table to be accessed
client.SetTargetTable(table_group_id=1, table_name='game_players')
# Send a data access request. The following uses `AddRecord` as an
example:
record = {'player_id': 10805514, 'player_name': 'Calvin',
         'player_email': 'calvin@test.com', 'game_server_id': 10,
         'login_timestamp': ['2019-12-12 15:00:00'],
         'logout_timestamp': ['2019-12-12 16:00:00'], 'is_online': False,
         'pay': {'pay_id': 10101, 'amount': 1000, 'method': 1}}
```

```
status, resp = client.AddRecord(record, custom_headers=None,
return_values=None, table_group_id=None, table_name=None)
```

4. After using the `SetTargetTable` method to specify the target table, you can use the `GetTargetTable` method to get it.

```
table_group_id, table_name = client.GetTargetTable()
```

You can also use the `table_group_id` and `table_name` parameters of the API to directly specify the target table. This method will not change the target table set in the `SetTargetTable` function.

```
status, resp = client.AddRecord(record, custom_headers=None,
return_values=None, table_group_id=1, table_name='game_players')
```

API List

GetRecord

This API is used to query a table record based on a primary key field. One record will be returned at a time.

```
@param keys Dictionary of primary key fields for querying target
record, which is required
@param select_fields Array of non-primary key field names to be
returned, which is optional. You can use a dotted path to specify nested
fields
@param custom_headers HTTP header to be specified, which is optional
@param table_group_id ID of the cluster table group where the table
resides, which is optional
@param table_name `table_name` of target table, which is optional

def GetRecord(self, keys, select_fields=[], custom_headers=None,
table_group_id=None, table_name=None)
```

AddRecord

This API is used to insert a record into a table. If the record already exists, an error will be reported.

```
@param record JSON object of the target record to be inserted, which
is required
```

```
@param custom_headers HTTP header to be specified, which is optional
@param table_group_id ID of the cluster table group where the table
resides, which is optional
@param table_name `table_name` of target table, which is optional

def AddRecord(self, record, custom_headers=None, return_values=None,
table_group_id=None, table_name=None)
```

SetRecord

This API is used to update or insert a record based on the primary key. If the record already exists, its value will be updated; otherwise, it will be inserted.

```
@param record JSON object of the target record to be set, which is
required. This operation involves insertion and modification semantics
@param custom_headers HTTP header to be specified, which is optional
@param table_group_id ID of the cluster table group where the table
resides, which is optional
@param table_name `table_name` of target table, which is optional

def SetRecord(self, record, custom_headers=None, return_values=None,
table_group_id=None, table_name=None)
```

DeleteRecord

This API is used to delete a record based on the primary key.

```
@param record JSON object of the target record to be deleted, which is
required. You only need to set a primary key field
@param custom_headers HTTP header to be specified, which is optional
@param table_group_id ID of the cluster table group where the table
resides, which is optional
@param table_name `table_name` of target table, which is optional

def DeleteRecord(self, record, custom_headers=None, return_values=None,
table_group_id=None, table_name=None)
```

FieldGetRecord

This API is used to get the values of specified fields based on the primary key. You need to specify the list of the fields to be returned (which must be non-primary key fields). You can use a dotted value to represent nested fields, i.e., `pay.amount`.

```
@param keys    Dictionary of primary key fields for querying target
record, which is required
@param select_fields  Array of non-primary key field names to be
returned, which is required. You can use a dotted path to specify nested
fields
@param custom_headers  HTTP header to be specified, which is optional
@param table_group_id  ID of the cluster table group where the table
resides, which is optional
@param table_name    `table_name` of target table, which is optional

def FieldGetRecord(self, keys, select_fields, custom_headers=None,
table_group_id=None, table_name=None)
```

FieldSetRecord

This API is used to update the values of specified fields based on the primary key. You need to specify the list of the fields to be updated (which must be non-primary key fields). You can use a dotted value to represent nested fields, i.e., `pay.amount`.

```
@param record    JSON object of the target record to be set, which is
required. This operation involves insertion and modification semantics
@param field_path  Array of field names (paths) to be set, which is
required. You can use a dotted path to specify nested fields
@param custom_headers  HTTP header to be specified, which is optional
@param table_group_id  ID of the cluster table group where the table
resides, which is optional
@param table_name    `table_name` of target table, which is optional

def FieldSetRecord(self, record, field_path, custom_headers=None,
return_values=None, table_group_id=None, table_name=None)
```

FieldIncRecord

This API is used to update (auto-increment/decrement) the value of a specified numeric field based on the primary key. For example, if the original value of `pay.method` is 200, the requested value is 50, then the final value will be 250; if the requested value is -50, then the final value will be 150.

Note:

To auto-decrement, the type of the target field passed into the request must be a signed numeric, such as `int32` or `int64`.

```
@param record    Record to be auto-incremented/decremented, which is
required. It must contain the primary key, and the target record must be
in integer type
@param custom_headers    HTTP header to be specified, which is optional
@param table_group_id    ID of the cluster table group where the table
resides, which is optional
@param table_name    `table_name` of target table, which is optional

def FieldIncRecord(self, record, custom_headers=None,
return_values=None, table_group_id=None, table_name=None)
```

PartkeyGetRecord

This API is used to query data based on the index defined in the table and return one or multiple records matching the index keys. It can return the list of specified fields and allow you to specify `limit` and `offset` to control the size of the packet returned by each request.

```
@param index_keys    Dictionary of index keys for querying target record,
which is required
@param index_name    Name of the index for query, which is required
@param select_fields    Array of non-primary key field names to be
returned, which is optional. You can use a dotted path to specify nested
fields
@param custom_headers    HTTP header to be specified, which is optional
@param table_group_id    ID of the cluster table group where the table
resides, which is optional
@param table_name    `table_name` of target table, which is optional
@param limit    Maximum number of records to be returned, which is
optional
@param offset    Initial offset of the number of records to be returned,
which is optional

def PartkeyGetRecord(self, index_keys, index_name, select_fields=[],
custom_headers=None, table_group_id=None, table_name=None, limit=None,
```

```
offset=None)
```

For this API, the maximum size of a packet returned by one request is `256 KB`, and the `limit` value is subject to the size of one single record. We recommend you set it as follows:

- If the size of one single record is smaller than 256 KB, you can set `limit` to `256 KB / record size`; for example, if the record size is 10 KB, we recommend you set `limit` to a value between 20 and 25.
- If the size of one single record is greater than or equal to 256 KB, set `limit` to 1, i.e., only one record will be returned by one request.

The following are the response packets returned with and without `limit` and `offset` parameters set, respectively:

```
# Index key settings
{'player_id': 39775502, 'player_name': 'Sara'}

# The response packet will be as follows if `limit` and `offset` are not
set in the request:
{'ErrorCode': 0, 'ErrorMsg': u'Succeed', 'MultiRecords':
[{'RecordVersion': 1, 'Record': {'logout_timestamp': [u'2019-12-12
16:00:00'], 'player_name': u'Sara', 'login_timestamp': [u'2019-12-12
15:00:00'], 'pay': {'amount': 100, 'pay_id': 1, 'method': 1},
u'game_server_id': 1, 'player_email': u'sara@test.com', u'is_online':
True, u'player_id': 39775502}}, {'RecordVersion': 1, 'Record':
{'logout_timestamp': [u'2019-12-12 16:10:00'], 'player_name': u'Sara',
u'login_timestamp': [u'2019-12-12 15:10:00'], 'pay': {'amount': 200,
u'pay_id': 2}, u'game_server_id': 2, 'player_email': u'sara@163.com',
u'is_online': True, u'player_id': 39775502}}, {'RecordVersion': 1,
u'Record': {'logout_timestamp': [u'2019-12-12 16:20:00'],
u'player_name': u'Sara', u'login_timestamp': [u'2019-12-12 15:20:00'],
u'pay': {'amount': 300, u'pay_id': 3, u'method': 1}, u'game_server_id':
3, u'player_email': u'sara@gmail.com', u'is_online': True, u'player_id':
39775502}}], u'RemainNum': 0, u'TotalNum': 3}

# The response packet will be as follows if `limit` and `offset` are set
in the request and `limit` is set to 2:
{'ErrorCode': 0, 'ErrorMsg': u'Succeed', 'MultiRecords':
[{'RecordVersion': 1, 'Record': {'logout_timestamp': [u'2019-12-12
16:00:00'], 'player_name': u'Sara', 'login_timestamp': [u'2019-12-12
15:00:00'], 'pay': {'amount': 100, 'pay_id': 1, 'method': 1},
u'game_server_id': 1, 'player_email': u'sara@test.com', u'is_online':
```

```
True, u'player_id': 39775502}}, {u'RecordVersion': 1, u'Record':  
{u'logout_timestamp': [u'2019-12-12 16:10:00'], u'player_name': u'Sara',  
u'login_timestamp': [u'2019-12-12 15:10:00'], u'pay': {u'amount': 200,  
u'pay_id': 2}, u'game_server_id': 2, u'player_email': u'sara@163.com',  
u'is_online': True, u'player_id': 39775502}}], u'RemainNum': 1,  
u'TotalNum': 3}
```

As shown above, if `limit` and `offset` are set, the number of entries in the returned result will be the same as that set in `limit`.

To get the full index data, you can judge whether the obtained data is complete based on the `RemainNum` and `TotalNum` flags returned in the response packet.

Downloading RESTful API Samples in Multiple Languages

Last updated: 2024-12-04 10:16:20

RESTful SDK Download for Various Languages

Language	Installation Package Name	Download
GO	tcaplusdb-restapi-go-sdk.zip	Download
Java	tcaplusdb-restapi-java-sdk.zip	Download
PHP	tcaplusdb-restapi-php-sdk.zip	Download
Python 2.x	tcaplusdb-restapi-python-sdk-2.x.zip	Download
Python 3.x	tcaplusdb-restapi-python-sdk-3.x.zip	Download