**OceanBase客户端使用指南**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **编号** | **文档版本** | **修订章节** | **修订原因** | **修订日期** | **修订人** |
| **1** | 0.1 | new | 新建文档 | 2010-12-27 | 若海 |
| **2** | 0.2 | 2.3，3.2 | API更新 | 2011-5-6 | 若海 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

[1 OceanBase客户端工作原理 3](#_Toc281229461)

[2 客户端常用数据结构说明 3](#_Toc281229462)

[2.1 rowkey 3](#_Toc281229463)

[2.2 RowData 3](#_Toc281229464)

[2.3 QueryInfo 3](#_Toc281229465)

[2.4 InsertMutator和UpdateMutator 4](#_Toc281229466)

[3 接口使用说明 4](#_Toc281229467)

[3.1 初始化OceanBase java客户端 4](#_Toc281229468)

[3.1.1 配置项 4](#_Toc281229469)

[3.1.2 创建客户端实例 4](#_Toc281229470)

[3.2 接口使用说明 5](#_Toc281229471)

[3.2.1 ResultCode定义 5](#_Toc281229472)

[3.2.2 get 5](#_Toc281229473)

[3.2.3 query 5](#_Toc281229474)

[3.2.4 insert 6](#_Toc281229475)

[3.2.5 update 6](#_Toc281229476)

[3.2.6 delete 6](#_Toc281229477)

本文档描述OceanBase客户端的使用方法，由于当前仅提供Java客户端，所以暂时只包含Java客户端的使用方法。

文档主要包括以下几个部分：

1. 客户端的工作原理
2. 客户端常用数据结构说明
3. 客户端的初始化和接口使用说明

# OceanBase客户端工作原理

OceanBase是一个分布式表格存储系统，主要包括以下模块：

1. rootserver
2. updateserver
3. chunkserver
4. mergeserver

rootserver负责维护系统数据的分布信息，客户端根据用户请求的rowkey，从rootserver查询负责该rowkey对应数据的mergeserver。

updateserver负责整个系统的数据更新，更新的数据会定期合并到chunkserver。

chunkserver负责静态数据的存储。

mergeserver提供系统的查询服务，当接收到查询时，mergeserver会根据相应的rowkey（或者范围）请求chunkserver上存储的静态数据，以及updateserver上更新的数据，然后做合并，返回结果数据。

mergeserver还支持对数据的过滤，排序等操作。

# 客户端常用数据结构说明

## rowkey

OceanBase中所有的数据都需要有一个rowkey，数据在存储时按照rowkey排序。RowKey为字节流，rowkey的结构在系统的schema中指定。

## RowData

OceanBase客户端使用RowData来表示一行数据，该数据结构通常用在查询返回时。数据结构包括：

1. 表名
2. rowkey
3. 各列数据，列的数据由列名和值组成，提供按照列名取值的接口

## QueryInfo

OceanBase提供查询功能，查询的参数使用QueryInfo对象封装，QueryInfo主要包含以下信息：

|  |  |
| --- | --- |
| **字段名** | **含义** |
| startKey | 查询范围的开始rowKey |
| endKey | 查询范围的结束rowKey |
| Flag | 查询的标识位：INCLUSIVE\_START （结果包含startKey）；INCLUSIVE\_END（结果包含endKey）；MIN\_VALUE（startKey是最小值，即比任何用户数据小）；MAX\_VALUE（endKey是最大值，即比任何用户数据大） |
| orderBy | 排序字段列表，支持多个排序字段，每个字段包括字段名和排序规则（升序/降序） |
| columns | 查询换回的列名列表 |
| pageSize | 分页的单页大小 |
| pageNum | 分页时当前的页码 |
| Limit | 单次查询返回的最大条目数 |
| filter | 过滤条件，比如price > 1.0等 |
| groupbyParam | groupBy的条件参数 |

## InsertMutator和UpdateMutator

Mutator封装了对一条记录的修改操作，客户端把要修改的数据填充在Mutator中，作为请求参数发送到updateserver，updateserver执行相应的操作。

Mutator的主要数据结构包括：

|  |  |
| --- | --- |
| **字段名** | **含义** |
| Table | 更新的表名 |
| rowKey | 待更新的数据的rowKey |
| Columns | 待更新的数据列表，结构为List<Tuple>，其中Tupe为一个name/value对 |

# 接口使用说明

## 初始化OceanBase java客户端

### 配置项

OceanBase客户端支持3个配置项

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 默认值 | 含义 |
| IP | String | 无 | Rootserver的地址 |
| Port | Int | 无 | Rootserver的端口 |
| Timeout | Int | 1000 | 客户端请求的超时时间，单位为毫秒（ms） |

### 创建客户端实例

Client obclient = **null**;

// 创建一个客户端实例

ClientImpl clientInstance = **new** ClientImpl();

clientInstance.setIp("192.168.0.1"); // 设置rootserver地址

clientInstance.setPort(1234); // 设置rootserver端口

clientInstance.setTimeout(2000); // 设置超时时间

obclient = clientInstance;

## 接口使用说明

OceanBase的客户端只要提供读取、查询和更新的接口。

### ResultCode定义

OceanBase客户端使用返回值来标识请求的结果，可能的返回值包括：

|  |  |
| --- | --- |
| **返回值** | **含义** |
| OB\_ERROR | 未知错误，该错误通常为服务器端内部错误 |
| OB\_SUCCESS | 请求成功 |
| OB\_ERROR\_FUNC\_VERSION | 接口版本错误，使用的客户端和服务器端版本不一致 |
| OB\_INVALID\_ARGUMENT | 参数错误 |
| OB\_NOT\_INIT | 服务器状态错误，服务未初始化成功 |
| OB\_ERROR\_OUT\_OF\_RANGE | 查询超过范围 |
| OB\_NOT\_SUPPORTED | 操作不支持，这个错误通常发生在请求的功能未实现，比如当前请求DB语义的操作 |
| OB\_SCHEMA\_ERROR | 请求的schema和服务器端的不匹配 |
| OB\_DATA\_NOT\_SERVE | 请求的数据不由该服务器负责。通常发生在根据rowKey找到的MergeServer所在的chunkserver已经不负责该rowKey对应的数据。该错误不影响请求的结果。 |

### get

get接口的定义为：

**public** Result<RowData> get(String table, Rowkey rowkey, Set<String> columns);

含义：get接口用于获取特性行的数据

参数定义：

|  |  |
| --- | --- |
| Table | 表名 |
| rowKey | 需要获取的数据的rowKey |
| Columns | 需要返回的列名 |

返回值：成功返回相应的数据，失败返回相应的ResultCode

### query

query接口的定义为：

**public** Result<List<RowData>> query(String table, QueryInfo query);

含义：query接口用于查询指定rowKey所对应范围的数据

参数定义：

|  |  |
| --- | --- |
| Table | 查询的表名 |
| QueryInfo | 查询参数对象，具体参见数据结构说明 |

返回：成功，返回结果列表，失败返回相应的ResultCode

### insert

insert接口的定义为：

**public** Result<Boolean> insert(InsertMutator mutator);

含义：insert接口用于新增一条数据

参数：mutator，insert的数据对象，具体参见数据结构说明

返回值：是否成功，成功返回True，否则返回False

### update

update接口的定义为：

**public** Result<Boolean> update(UpdateMutator mutator);

含义：更新数据

参数：mutator，update的数据对象，具体参见数据结构说明

返回值：是否成功，成功返回True，否则返回False

### delete

delete接口的定义为：

**public** Result<Boolean> delete(String table, Rowkey rowkey);

含义：删除数据

参数：

|  |  |
| --- | --- |
| Table | 表名 |
| rowKey | 要删除的数据的rowKey |

返回值：是否成功，成功返回True，否则返回False

### scanTabletList

scanTableList接口的定义为：

public Result<List<RowData>> scanTabletList(String table, QueryInfo query);

含义：从rootserver查询tablet信息，返回的结果是一个RowData列表，每行代表一个Tablet信息，主要关心的数据是负责这个Tablet的mergeserver地址

参数：

|  |  |
| --- | --- |
| Table | 查询的表名 |
| QueryInfo | 查询参数对象，具体参见数据结构说明 |