实时库接口说明

一 CTableOp 类接口说明

CTableOp 类的接口都是对本地实时数据库做操作，包括打开一张表、按域取表数据、修改一张表、向表里写入数据、按关键字删除表的数据和清空一张表。

包含头文件：db_api/odb_tableop.h

连接动态库：libodb_apiall_lib.so

1．打开一张表：Open

注：在使用实时库的接口对某张表操作之前必须先打开这张表

int Open(const int app_no, const int table_no, const short context_no = 0)

接口参数	输入/输出	参数（返回值）说明	备注
int app_no	In	应用号	
int table_no	In	表号	
short context_no	In	态号	缺省为 0，表示当前态
返回值	Out	0：成功；<0：失败	

按应用号，表号，态号打开一张表，态号默认为 0，表示当前态

例：打开实时态 SCADA 应用的遥测定义表（301）：

```
CTableOp tab_op; //建一个 CTableOp 类的对象
int ret_code = tab_op.Open(AP_SCADA, 301, 1);
```

成功则返回 0，否则返回小于 0 的数

2．取表参数：GetTablePara

int GetTablePara(short& field_num, int& record_num, int& record_size);

接口参数	输入/输出	参数（返回值）说明	备注
short field_num	Out	域的个数	
int record_num	Out	记录的个数	
int record_size	Out	每条记录的大小	
返回值	Out	0：成功；<0：失败	

得到该表的域个数、记录个数和每条记录的大小

例：SCADA 下的 220 表

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
   return -1;
}
//赋初值
short field_num = 0;
int record_num = 0;
int record_size = 0;
ret_code = tab_op.GetTablePara(field_num, record_num, record_size);
```
成功则返回 0，否则返回小于 0 的数

3．取域参数：GetFieldPara

```
int GetFieldPara(struct FIELD_BASE_INFO& field_info);
```

接口参数	输入/输出	参数（返回值）说明	备注
FIELD_BASE_INFO& field_info	In/Out	域的信息	需要指定 field_info.field_no
返回值	Out	0：成功；<0：失败	

得到一张表某个域的信息如大小、类型等

例：SCADA 220 表 3 号域

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
    return -1;
}
struct FIELD_BASE_INFO field_base;
field_base.field_no = 3;   //指定域号
ret_code = tab_op.GetFieldPara(field_base);
```
成功则返回 0，否则返回小于 0 的数

4．写入记录：TableWrite

```
int  TableWrite( const char* buf_ptr, const int buf_size)
```

接口参数	输入/输出	参数（返回值）说明	备注
char* buf_ptr	In	写入的数据	数据结构要和表的结构一致
int buf_size	In	写入数据的大小	
返回值	Out	>=0：成功；<0：失败	

写入多条记录，写入的必须是该表中不存在的记录

例：往 SCADA 下 220 号表写入一条记录

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
  return -1;
}
BRK_DEVICE_SCADA  brk_device_struct;  //220 表的表结构类型
……//为这个结构赋值
int buf_size = sizeof(brk_device_struct);
ret_code = tab_op.TableWrite( (char*)&brk_device_struct, buf_size);
```

成功则返回写入的记录个数，否则返回小于 0 的数

5．取表记录：TableGet

(1) int TableGet(const int field_no, CBuffer& buf_base);

接口参数	输入/输出	参数（返回值）说明	备注
int field_no	In	域号	
CBuffer buf_base	Out	存放表记录的类	buf_base.GetBufPtr() buf_base.GetLength()
返回值	Out	>=0：成功；<0：失败	

int TableGet(const char* field_name, CBuffer& buf_base);

接口参数	输入/输出	参数（返回值）说明	备注
char* field_name	In	域名	
CBuffer buf_base	Out	存放表记录的类	buf_base.GetBufPtr() buf_base.GetLength()
返回值	Out	>=0：成功；<0：失败	

取本张表所有记录的某个域（根据域号或域名）

例：取 SCADA 220 表的 2 号域

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
    return -1;
}
int field_no = 2;
CBuffer buf_base;
ret_code = tab_op.TableGet(field_no, buf_base);
```

成功则返回 buf_base 的大小，否则返回小于 0 的数

(2)int TableGet(const std::vector<int>& vec_field_no, CBuffer& buf_base);

接口参数	输入/输出	参数（返回值）说明	备注
vector<int>vec_field_no	In	域号	
CBuffer buf_base	Out	存放表记录的类	buf_base.GetBufPtr() buf_base.GetLength()
返回值	Out	>=0：成功；<0：失败	

取本张表所有记录的某些域（根据域号）

例：取 SCADA220 表的 2 号域和 3 号域

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
    return -1;
}
vector<int> vec_field_no(2);
vec_field_no[0] = 2;
vec_field_no[1] = 3;
CBuffer buf_base;
ret_code = tab_op.TableGet(vec_field_no, buf_base);
```

成功则返回 buf_base 的大小，否则返回小于 0 的数

(3) int TableGet(const int field_no, char** field_buf_ptr, int& buf_size) ;
int TableGet(const char* field_name, char** field_buf_ptr, int& buf_size);
int TableGet(const std::vector<int>& vec_field_no, char** field_buf_ptr, int& buf_size);

用 char*或 char**及其大小取代 CBuffer 的上述各个接口，使用方法与上述接口基本相同，不再举例

6．按关键字取某张表的某条记录：TableGetByKey

(1) int TableGetByKey(const char* key_ptr, char* buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	
char* buf_ptr	Out	存放取出数据的指针	
int buf_size	In	取出数据的大小	
返回值	Out	>=0：成功；<0：失败	

取出某张表关键字为 key_ptr 的记录

例：取 SCADA 下 220 表关键字为 220000001 的记录

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
    return -1;
}
//赋初值
int key = 220000001;
BRK_DEVICE_SCADA   brk_device_struct;   //220 表的表结构类型
……//为这个结构赋初值
int buf_size = sizeof(brk_device_struct);
int buf_size = sizeof(brk_device_struct);
ret_code = tab_op.TableGetByKey((char*)&key, (char*)&brk_device_struct, buf_size);
```

成功则返回记录的大小，否则返回小于 0 的数

(2).int TableGetByKey(const char* key_ptr, const int field_no, char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	
int field_no	In	域号	
char* field_buf_ptr	Out	存放取出数据的指针	
int buf_size	In	取出数据的大小	
返回值	Out	>=0：成功；<0：失败	

int TableGetByKey(const char* key_ptr, const char* field_name, char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	
char* field_name	In	域名	
char* field_buf_ptr	Out	存放取出数据的指针	
int buf_size	In	取出数据的大小	
返回值	Out	>=0：成功；<0：失败	

int TableGetByKey(const char* key_ptr, const std::vector<int>& vec_field_no, char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	
vector<int> vec_field_no	In	域号	
char* field_buf_ptr	Out	存放取出数据的指针	
int buf_size	In	取出数据的大小	
返回值	Out	>=0：成功；<0：失败	

取出某张表关键字为 key_ptr 的记录的一个或几个域

例：取 SCADA 下 220 表 220000001 记录的 2 号域和 3 号域

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
    return -1;
}
int key = 220000001;
vector<int> vec_field_no(2);
vec_field_no[0] = 2;
vec_field_no[1] = 3;
//构造一个由二号域和三号域数据类型组成的结构
struct TEMP_STRU
{
    int fac_id;
    char brk_name[64];
} temp_struct;
……//为这个结构对象赋初值
int buf_size = sizeof(struct TEMP_STRU);
ret_code = tab_op.TableGetByKey((char*)&key, vec_field_no, (char*)&temp_struct, buf_size);
```

成功则返回 field_buf_ptr 的大小，否则返回小于 0 的数

(3) int TableGetByKey(const char* key_ptr, const int keybuf_size, const std::vector<int>& vec_field_no, char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	

int keybuf_size	In	关键字的大小
vector<int> vec_field_no	In	域号
char* field_buf_ptr	In	存放取出数据的指针
int buf_size	In	取出数据的大小
返回值	Out	>=0：成功；<0：失败

上述接口的变体，参数中包括关键字的大小，用于复合关键字的表，使用方法与上述接口相同

7．修改表数据：TableModify

int TableModify(const int field_no, const char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
int field_no	In	域号	
char* field_buf_ptr	In	修改数据的指针	
int buf_size	In	修改数据的大小	
返回值	Out	>=0：成功；<0：失败	

int TableModify(const char* field_name, const char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* field_name	In	域名	
char* field_buf_ptr	In	修改数据的指针	
int buf_size	In	修改数据的大小	
返回值	Out	>=0：成功；<0：失败	

int TableModify(const std::vector<int>& vec_field_no, const char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
vector<int> vec_field_no	In	域号	

char* field_buf_ptr	In	修改数据的指针
int buf_size	In	修改数据的大小
返回值	Out	>=0：成功；<0：失败

修改某张表的一个或几个域

例：修改 SCADA 220 表的 2 号域

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
    return -1;
}
struct FIELD_BASE_INFO field_base;
int field_no = 2;
field_base.field_no = field_no;
ret_code = tab_op.GetFieldPara(field_base);
if(ret_code < 0)
{
    return -1;
}
short field_num = 0;
int record_num = 0;
int record_size = 0;
ret_code = tab_op.GetTablePara(field_num, record_num, record_size);
if(ret_code < 0)
{
    return -1;
}

int buf_size = field_base.field_length* record_num;
char* field_buf_ptr = (char*)malloc(buf_size);
……//为 field_buf_ptr 赋值
ret_code = tab_op.TableModify(field_no, field_buf_ptr, buf_size);
//成功则返回成功修改的记录条数，否则返回小于 0 的数
```

8．根据关键字修改表记录：TableModifyByKey

(1) int TableModifyByKey(const char* key_ptr, const int field_no, const char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	

int field_no	In	域号
char* field_buf_ptr	In	修改数据的指针
int buf_size	In	修改数据的大小
返回值	Out	>=0：成功；<0：失败

int TableModifyByKey(const char* key_ptr, const char* field_name, const char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	
char* field_name	In	域名	
char* field_buf_ptr	In	修改数据的指针	
int buf_size	In	修改数据的大小	
返回值	Out	>=0：成功；<0：失败	

int TableModifyByKey(const char* key_ptr, const std::vector<int>& vec_field_no, const char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	
vector<int> vec_field_no	In	域号	
char* field_buf_ptr	In	修改数据的指针	
int buf_size	In	修改数据的大小	
返回值	Out	>=0：成功；<0：失败	

根据关键字修改某张表的一条或几条记录的一个或几个域

例：修改 SCADA 220 表的关键字为 220000001 的记录的 2 号域

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
   return -1;
}
```

int key = 220000001;

int fac_id = 210000001;　　//220 表 2 号域是厂站 id

ret_code = tab_op.TableModifyByKey((char*)&key, field_no, (char*)&fac_id, sizeof(int));

//成功则返回成功修改的记录条数，否则返回小于 0 的数

(2) int TableModifyByKey(const char* key_ptr, const int keybuf_size, const int field_no, const char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	
int keybuf_size	In	关键字的大小	
int field_no	In	域号	
char* field_buf_ptr	In	修改数据的指针	
int buf_size	In	修改数据的大小	
返回值	Out	>=0：成功；<0：失败	

int TableModifyByKey(const char* key_ptr, const int keybuf_size, const char* field_name, const char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	
int keybuf_size	In	关键字的大小	
char* field_no	In	域名	
char* field_buf_ptr	In	修改数据的指针	
int buf_size	In	修改数据的大小	
返回值	Out	>=0：成功；<0：失败	

int TableModifyByKey(const char* key_ptr, const int keybuf_size, const std::vector<int>& vec_field_no, const char* field_buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	关键字	
int keybuf_size	In	关键字的大小	
vector<int> vec_field_no	In	域号	

char* field_buf_ptr	In	修改数据的指针
int buf_size	In	修改数据的大小
返回值	Out	>=0：成功；<0：失败

带 keybuf_size 参数的上述介绍的几个接口，使用方法与上述接口基本相同

9．更新表记录：TableUpdate

注：如果有这条记录就修改，没有就写入

int TableUpdate(const char* buf_ptr, const int buf_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* buf_ptr	In	更新的数据	数据结构要和表的结构一致
int buf_size	In	更新数据的大小	
返回值	Out	>=0：成功；<0：失败	

int TableUpdate(const char* buf_ptr, const int record_num, const int record_size);

接口参数	输入/输出	参数（返回值）说明	备注
char* buf_ptr	In	更新的数据	数据结构要和表的结构一致
int record_num	In	更新的记录的条数	
int record_size	In	每条记录的大小	
返回值	Out	>=0：成功；<0：失败	

使用方法与 TableModify 相同，只不过 TableModify 只修改存在的记录，并不写入新的记录

10．删除表记录：DeleteRecord

int DeleteRecord(const char* key_ptr);

接口参数	输入/输出	参数（返回值）说明	备注
char* key_ptr	In	要删除的记录的关键字	
返回值	Out	>=0：成功；<0：失败	

根据关键字删除一条或几条记录

例：删除 SCADA 220 表的 220000001 记录

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 220);
if(ret_code < 0)
{
    return -1;
}
int key = 220000001;
ret_code = tab_op.DeleteRecord((char*)&key);
//成功则返回删除记录的个数，否则返回小于 0 的数
```

11．利用 sql 语句取表数据：SqlGet

```
int SqlGet(const char* str_sql, CBuffer& buf_base);
```

接口参数	输入/输出	参数（返回值）说明	备注
char* str_sql	In	Sql 语句	
CBuffer buf_base	Out	存放取出数据的类	buf_base.GetBufPtr() buf_base.GetLength()
返回值	Out	>=0：成功；<0：失败	

```
int SqlGet(const char* str_sql, char** buf_ptr, int& buf_size);
```

接口参数	输入/输出	参数（返回值）说明	备注
char* str_sql	In	Sql 语句	
char** buf_ptr	Out	存放取出数据二重指针	
int buf_size	Out	取出数据的大小	
返回值	Out	>=0：成功；<0：失败	

例：取 SCADA 遥测定义表的遥测 id 为 301000001 的记录

```
CTableOp tab_op;
int ret_code = tab_op.Open(AP_SCADA, 301);
if(ret_code < 0)
{
    return -1;
}
char sql_str[200] = “select * from yc_define where yc_id = 301000001”;
CBuffer buf_base;
ret_code = tab_op.SqlGet(sql_str, buf_base);
//成功则返回 buf_base 的大小，否则返回小于 0 的数
```

12．清空表记录：TableClear

int TableClear();

接口参数	输入/输出	参数（返回值）说明	备注
返回值	Out	0：成功；<0：失败	

把一张表清空。

2.2 CTableNet 类接口说明

CTableNet 类的接口和 CTableOp 类的接口大致相同，使用方法也是一样的。所不同的是 CTableOp 类的接口是对本地实时库操作，而 CTableNet 类的接口是对该应用的当前主机的实时库进行操作，或者某个指定的主机进行操作。与 CTableOp 类相同的接口的使用方法请参阅上一节，这里只介绍一个指定主机的接口。

包含头文件：db_api/odb_tablenet.h

连接动态库：libodb_apiall_lib.so

int SetHostMode(const char* host_name);

接口参数	输入/输出	参数（返回值）说明	备注
char* host_name	In	指定的主机名	
返回值	Out	0：成功；<0：失败	

如果不指定主机则 CTableNet 的接口是对应用的当前主机进行操作，指定了主机就是对指定的服务器进行操作。

例：假设 SCADA 应用有两台服务器 kf08-1 和 kf02-1，kf08-1 是主机，指定对 kf02-1 的实时库操作，取 SCADA 应用的 220 表 2 号域

```
CTableNet tab_net;
int ret_code = tab_net.SetHostMode( “kf02-1”);
if(ret_code < 0)
{
    return -1;
}
tab_net.Open( AP_SCADA, 220);
if(ret_code < 0)
{
    ……
    return -1;
}
CBuffer buf_base;
int field_no = 2;
ret_code = tab_net.TableGet(field_no, buf_base);
```

成功则返回 buf_base 的大小，否则返回小于 0 的数

注：如果要循环使用 CTableNet 类的接口，则调用这个接口的 CTableNet 类的对象一定要是全局变量，不可以是局部变量！

例：正确的方法

```
CTableNet  table_net;
……
while(1)
{
      ret_code = table_net.TableGet(…);
      usleep(2000);
}
```

错误的方法

```
while(1)
{
      CTableNet table_net;
      ……
      ret_code = table_net.TableGet(…);
      usleep(2000);
}
```