[SQLite3 C/C++编程接口介绍](http://blog.csdn.net/woxiaozhi/article/details/6530529)

分类： [C++学习笔记](http://blog.csdn.net/woxiaozhi/article/category/800150) [linux学习笔记](http://blog.csdn.net/woxiaozhi/article/category/800141)2011-06-07 22:09 1309人阅读 [评论](http://blog.csdn.net/woxiaozhi/article/details/6530529#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/woxiaozhi/article/details/6530529#report)

本文介绍一下SQLite C/C++接口。

早期的SQLite C/C++接口只有5个接口函数, 很容易学习。 新版本的SQLite新增了很多功能, 目前大概有185个API接口。本文介绍一些核心的API接口和对象。

1. 核心对象   
数据库连接对象: sqlite3  
prepared\_statement对象: sqlite3\_stmt

严格的讲 sqlite3\_stmt对象也不是必须的， 另外两个封装了qlite3\_stmt的接口sqlite3\_exec和sqlite3\_get\_table可以代替sqlite3\_stmt对象完成数据存取工作。 但是理解sqlite3\_stmt对于充分使用SQLite有很大帮助。

以下是一些操作 sqlite3和sqlite3\_stmt对象的接口。下面这些名字只是提供一个概念，实际的函数可能会有很多版本比如sqlite3\_open() 有，sqlite3\_open\_v16(), sqlite3\_open\_v2()  
sqlite3\_open()  
sqlite3\_prepare()  
sqlite3\_step()  
sqlite3\_column()  
sqlite3\_finalize()  
sqlite3\_close()

函数说明  
sqlite3\_open() 打开一个数据库连接, 返回sqlite3对象  
sqlite3\_prepare() 此函数将SQL转换成sqlite3\_stmt对象, 通常使用sqlite3\_prepare\_v2()  
sqlite3\_step()  此函数单步执行sqlite3\_stmt  
sqlite3\_column()   返回 sqlite3\_stmt所在行的指定column的值， 有如下具体函数  
    \* sqlite3\_column\_blob()  
    \* sqlite3\_column\_bytes()  
    \* sqlite3\_column\_bytes16()  
    \* sqlite3\_column\_count()  
    \* sqlite3\_column\_double()  
    \* sqlite3\_column\_int()  
    \* sqlite3\_column\_int64()  
    \* sqlite3\_column\_text()  
    \* sqlite3\_column\_text16()  
    \* sqlite3\_column\_type()  
    \* sqlite3\_column\_value()

sqlite3\_finalize()  销毁 sqlite3\_stmt对象, 所有sqlite3\_stmt对象都应该销毁以防止内存泄漏  
sqlite3\_close()  关闭数据库连接,  销毁sqlite3对象, 所有与这个sqlite3对象相关的sqlite3\_stmt对象都应该在调用这个函数之前销毁。

2. 绑定和重新执行sqlite3\_stmt   
sqlite3\_reset():  此函数使得执行过sqlite3\_step()的sqlite3\_stmt重新执行, 相当于将游标返回到开始位置重新读取数据, sqlite3\_reset()的效率比重新创建一个sqlite3\_stmt搞很多。

sqlite3\_bind(): 此函数用于INSERT SQL, 当同一个INSERT SQL要插入一系列数据时使用, 每次sqlite3\_step()后重新bind数据。

3. 一般用法和步骤   
(1) Create a prepared statement using sqlite3\_prepare().  
(2) Evaluate the prepared statement by calling sqlite3\_step() one or more times.  
(3) For queries, extract results by calling sqlite3\_column() in between two calls to sqlite3\_step().  
(4) Destroy the prepared statement using sqlite3\_finalize().

4. 简易接口   
sqlite3\_get\_table()  
sqlite3\_exec()

5. 例子

view plaincopy to clipboardprint?  
01.void basic\_interface()     
02.{     
03.    sqlite3 \*db;     
04.    char \*zErrMsg = 0;     
05.    int rc;     
06.    char sql[]="select \* from addr;";     
07.    sqlite3\_stmt \*stmt;     
08.    int id;     
09.    unsigned char \*street, \*city;     
10.    rc = sqlite3\_open("D:/VCWork/test.db", &db);     
11.    if( rc ){     
12.        fprintf(stderr, "Can't open database: %s/n", sqlite3\_errmsg(db));     
13.        sqlite3\_close(db);     
14.        return;     
15.    }     
16.    rc= sqlite3\_prepare\_v2(db,sql, strlen(sql), &stmt,0);       
17.    if( rc ){     
18.        fprintf(stderr, "Can't open statement: %s/n", sqlite3\_errmsg(db));     
19.        sqlite3\_close(db);     
20.        return;     
21.    }     
22.    //right align output format     
23.    printf("%10s %10s %10s/n", "id", "street", "city");     
24.    printf("%10s %10s %10s/n", "---", "---", "---");     
25.    while(sqlite3\_step(stmt)==SQLITE\_ROW ) {     
26.        id = sqlite3\_column\_int(stmt, 0);     
27.        street = (unsigned char\*)sqlite3\_column\_text(stmt,1);     
28.        city = (unsigned char\*)sqlite3\_column\_text(stmt, 2);     
29.        printf("%10d %10s %10s/n", id, street, city);     
30.    }     
31.    sqlite3\_finalize(stmt);     
32.    sqlite3\_close(db);     
33.}    
void basic\_interface()  
{  
 sqlite3 \*db;  
 char \*zErrMsg = 0;  
 int rc;  
 char sql[]="select \* from addr;";  
 sqlite3\_stmt \*stmt;  
 int id;  
 unsigned char \*street, \*city;  
 rc = sqlite3\_open("D:/VCWork/test.db", &db);  
 if( rc ){  
  fprintf(stderr, "Can't open database: %s/n", sqlite3\_errmsg(db));  
  sqlite3\_close(db);  
  return;  
 }  
 rc= sqlite3\_prepare\_v2(db,sql, strlen(sql), &stmt,0);    
 if( rc ){  
  fprintf(stderr, "Can't open statement: %s/n", sqlite3\_errmsg(db));  
  sqlite3\_close(db);  
  return;  
 }  
 //right align output format  
 printf("%10s %10s %10s/n", "id", "street", "city");  
 printf("%10s %10s %10s/n", "---", "---", "---");  
 while(sqlite3\_step(stmt)==SQLITE\_ROW ) {  
  id = sqlite3\_column\_int(stmt, 0);  
  street = (unsigned char\*)sqlite3\_column\_text(stmt,1);  
  city = (unsigned char\*)sqlite3\_column\_text(stmt, 2);  
  printf("%10d %10s %10s/n", id, street, city);  
 }  
 sqlite3\_finalize(stmt);  
 sqlite3\_close(db);  
}

参考：

An Introduction To The SQLite C/C++ Interface

<http://www.sqlite.org/cintro.html>

<http://blog.csdn.net/woxiaozhi/article/details/6530529>