### 简介：

seqlite3是一款轻型的数据库，是一款遵守ACID的关系型数据库管理系统，它包含在一个相对小的C库中。它的设计目标是嵌入式的，而且已经在很多嵌入式产品中使用了它，它占用资源非常的低，在嵌入式设备中，可能只需几百K的内存就够了。它能够支持Windows/Linux/Unix等等主流的操作系统，同时能够跟多种程序语言相结合，比如Tcl、C#、PHP、JAVA等，还有ODBC接口，同样比起Mysql、PostgreSQL这两款开源的世界著名数据库管理系统来讲，它的处理速度比他们都快。

### sqlite3的安装

**方式一**

1.拷贝

把文件：sqlite-autoconf-3071601.tar.gz拷贝到ubuntu，

解压：tar xvf sqlite-autoconf-3071601.tar.gz

2.安装

2.1 进入目录：sqlite-autoconf-3071601 cd sqlite-autoconf-3071601

2.2 依次运行以下命令：

./configure

make

sudo make install

以上步骤都成功执行，sqlite3安装成功，可以在命令行下输入：sqlite3 test.db

这个命令用来打开一个数据库文件，成功就可以去操作此数据库，这时候命令行应该是如下样子：

xxx@ubuntu:~/work/sqlite/sqlite\_demo$ sqlite3 test.db

SQLite version 3.7.16.1 2013-03-29 13:44:34

Enter ".help" for instructions

Enter SQL statements terminated with a ";"

sqlite>

如果不成功，提示header and source什么的不匹配，执行下面的操作：

wen@ubuntu:~/sqlite$ sudo cp /usr/local/lib/libsqlite3.\* /usr/lib/i386-linux-gnu/ -a

这句命令的意思：拷贝/usr/local/lib/目录下的所有相关库到/usr/lib/i386-linux-gnu/目录下，记得一定要加 -a。

**方式二**

Ubuntu linux下安装sqlite3

1.介绍：sqlite3是linux上的小巧的数据库，一个文件就是一个数据库。

2.安装：

要安装sqlite3，可以在终端提示符后运行下列命令:

sudo apt-get install sqlite3

检查版本

sqlite3 -version

3.测试

当前目录下建立test.db测试数据库文件

sqlite3 test.db

退出

.exit

4. 安装Sqlite3编译需要的工具包：

apt-get install libsqlite3-dev

5.图形界面

可以选择sqlitebrowser程式（qt3）

sudo apt-get install sqlitebrowser

启动图形界面可以在终端提示符后输入sqlitebrowser

### sqlite3常用命令

当前目录下建立或打开test.db数据库文件，并进入sqlite命令终端，以sqlite>前缀标识：

sqlite3 test.db

查看数据库文件信息命令(注意命令前带字符'.')：

sqlite>.database

查看所有表的创建语句：

sqlite>.schema

查看指定表的创建语句：

sqlite>.schema table\_name

以sql语句的形式列出表内容：

sqlite>.dump table\_name

设置显示信息的分隔符：

sqlite>.separator symble

Example：设置显示信息以‘：’分隔

sqlite>.separator :

设置显示模式：

sqlite>.mode mode\_name

Example:默认为list，设置为column，其他模式可通过.help查看mode相关内容

sqlite>.mode column

输出帮助信息：

sqlite>.help

设置每一列的显示宽度：

sqlite>.width width\_value

Example:设置宽度为2

sqlite>.width 2

列出当前显示格式的配置：

sqlite>.show

退出sqlite终端命令：

sqlite>.quit

或

sqlite>.exit

### sqite3常用指令

**1）建立数据表**

create table 表名 if not exists(field1 type1, field2 type2, ...);

table\_name是要创建数据表名称，fieldx是数据表内字段名称，typex则是字段类型。

例，建立一个简单的学生信息表，它包含学号与姓名等学生信息：

create table student\_info(stu\_no interger primary key, name text);

**2）添加数据记录**

insert into 表名(field1, field2, ...) values(val1, val2, ...);

valx为需要存入字段的值。

例，往学生信息表添加数据：

Insert into student\_info(stu\_no, name) values(0001, 'alex');

Insert into student\_info(stu\_no, name) values(0003, 'lc');

**3）修改数据记录**

update 表名 set field1=val1, field2=val2 where expression;

where是sql语句中用于条件判断的命令，expression为判断表达式

例，修改学生信息表学号为0001的数据记录：

update student\_info set stu\_no=0001, name='hence' where stu\_no=0001;

**4）删除数据记录**

delete from 表名 where expression;

不加判断条件则清空表所有数据记录。

例，删除学生信息表学号为0001的数据记录：

delete from student\_info where stu\_no=0001;

**5）查询数据记录**

select指令基本格式：

select columns from 表名 [where expression];

a查询输出所有数据记录

select \* from 表名;

b限制输出数据记录数量 val =3 查询前三行

select \* from 表名 limit val;

c升序输出数据记录

select \* from 表名 order by field asc;

d降序输出数据记录

select \* from 表名 order by field desc;

e条件查询

select \* from 表名 where expression;

select \* from 表名 where field in ('val1', 'val2', 'val3');

select \* from 表名 where field between val1 and val2;

f查询记录数目

select count (\*) from 表名;

g区分列数据

select distinct field from 表名;

有一些字段的值可能会重复出现，distinct去掉重复项，将列中各字段值单个列出。

**6) 增加一个字段项addr**

alter table 表名 add column addr type;

例如在food表中添加一个count列

alter table food add column count int;

**7) 改变表名**

alter table 旧表名 rename to 新表名;

**8）建立索引： 一般几百万大容量数据时使用**

当说数据表存在大量记录，索引有助于加快查找数据表速度。

create index index\_name on 表名(field);

例，针对学生表stu\_no字段，建立一个索引：

create index student\_index on student\_table(stu\_no);

建立完成后，sqlite3在对该字段查询时，会自动使用该索引。

**9）删除数据表或索引**

drop table 表名;

drop index 索引名;

**10）时间类型插入数据库表中**

DATETIME TIMESTAMP 用来存储日期/时间的不限制长度的字符串类型.

要求的格式是 ‘YYYY-MM-DD HH:MM:SS’,其他的东西被忽略.

create table tablename (id int,time DATETIME);

insert into tablename values(1003,datetime('now')); 插入1003和当前系统时间到表中

select \* from tablename where time between '2018-01-10' and '2018-01-19'; 查询表中在2018-01-10到208-01-19号之间所有数据

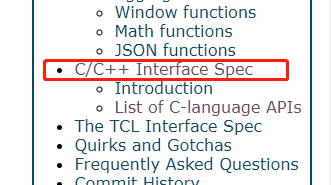
select id from tablename where time > '2018-01-10 15:29:06'; 查询tablename表中时间大于2018-01-10 15:29:06的所有id项数据

### sqlite3相关函数

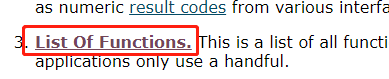
#### 1、sqlite3官网查询

我们可以直接去sqlite3官网查看相关信息：<https://www.sqlite.org/index.html>

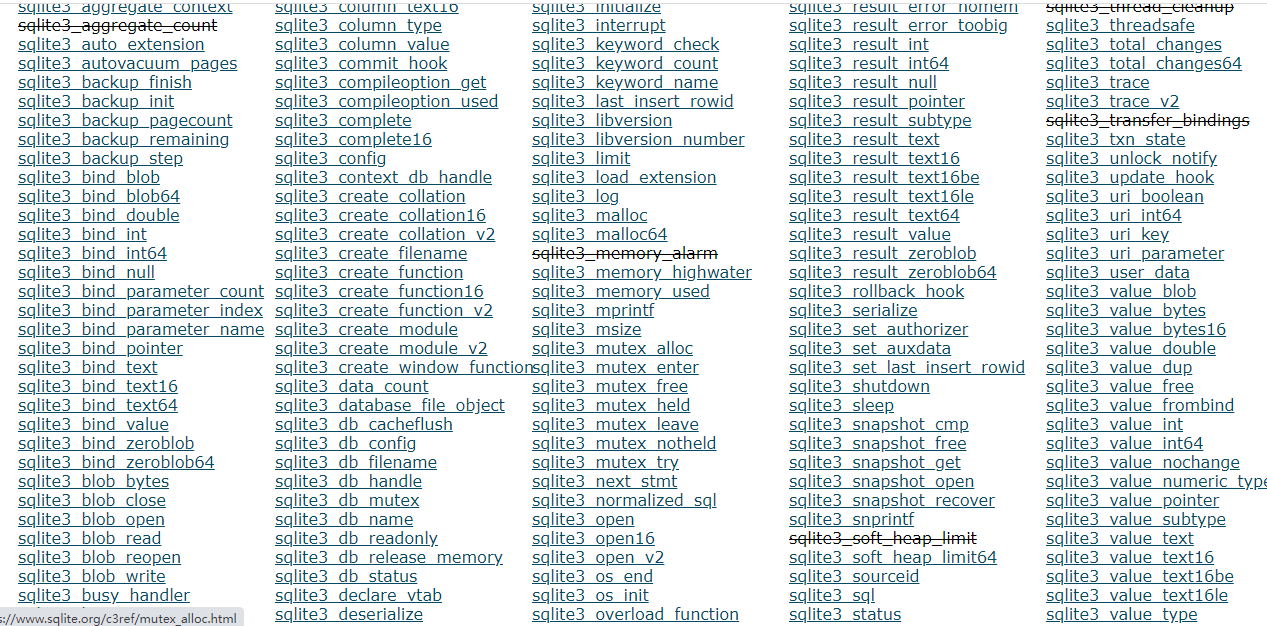
然后点击下图红框部分：



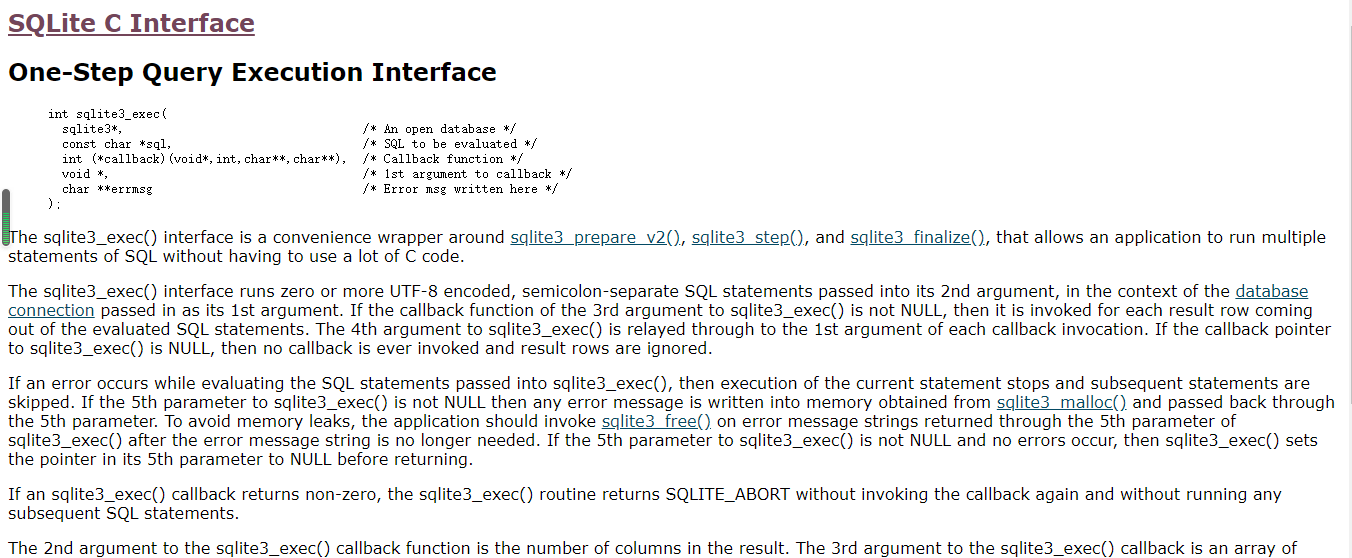
再然后点击下图红框：



接下来你将会看到非常多的函数：



随便进入一个比较常用的，比如sqlite3\_exec()函数：



#### 下载sqlite3包

下载命令：apt-get download sqlite3



解压命令：dpkg -x sqlite3\_3.11.0-1ubuntu1.5\_amd64.deb sqlite3\_package

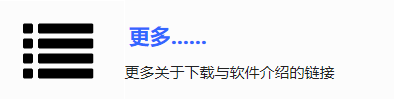


解压后进入到里面的usr文件夹查看，发现只有bin文件夹和share文件夹，并没有找 到想要的include文件夹，没有我们需要的sqlite3的库。



我们可以进入到debian官网，<https://www.debian.org/>

找到网页中的更多选项点进去：



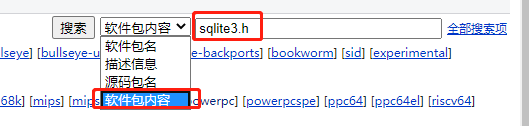
进去之后点击软件：



找到搜索软件包目录并搜索sqlite3



如果没找到就会进到下个界面，搜索框选择软件包内容，并搜索sqlite3.h：



最后就可以顺利找到我们需要下载的库是libsqlite3-dev：



回到Ubuntu终端输入命令：sudo apt-get install libsqlite3-dev



然后就可以查看到/usr/include/sqlite3.h确实存在了。



#### 代码编写

|  |
| --- |
| #include <stdio.h>  #include "sqlite3.h"  #include <stdlib.h>  #include <string.h>  int show\_message(void \*arg,int column,char \*\*value,char \*\*name)  {  int i;  for(i = 0;i < column;i++)  {  printf("%s\t = %s\t",name[i],value[i]);  }  printf("\n");  /\*注意：这里必须return 0，不然会报错\*/  return 0;  }  int main(void)  {  int ret = -1;  sqlite3 \*ppdb = NULL;  char sql[128] = {0};  int id,age;  char name[35];  //打开或创建数据库  ret = sqlite3\_open("student.db",&ppdb);  if(ret != SQLITE\_OK)  {  printf("sqlite3\_open:%s\n",sqlite3\_errmsg(ppdb));//注意这里不能用perror，因为perror()函数对应的每个错误都是库里面的错误码，而sqlite3是外来的。  exit(1);  }  //创建表  sprintf(sql,"create table if not exists student (id integer,name text,age interger);");  ret = sqlite3\_exec(ppdb,sql,NULL,NULL,NULL);  if(ret != SQLITE\_OK)  {  printf("sqlite3\_exec1:%s\n",sqlite3\_errmsg(ppdb));//注意这里不能用perror，因为perror()函数对应的每个错误都是库里面的错误码，而sqlite3是外来的。  exit(1);  }    //插入数据  while(1)  {  fprintf(stderr,"请输入学号，姓名，年龄:");  scanf("%d %s %d",&id,name,&age);  if(id == -1)  {  break;  }  memset(sql,0,128);  sprintf(sql,"insert into student values(%d,'%s',%d);",id,name,age);  ret = sqlite3\_exec(ppdb,sql,NULL,NULL,NULL);  if(ret != SQLITE\_OK)  {  printf("sqlite3\_exec2:%s",sqlite3\_errmsg(ppdb));  exit(1);  }  }  //查询  memset(sql,0,sizeof(sql));  sprintf(sql,"select \* from student;");  #if 0  ret = sqlite3\_exec(ppdb,sql,show\_message,NULL,NULL);  if(ret != SQLITE\_OK)  {  printf("sqlite3\_exec3:%s\n",sqlite3\_errmsg(ppdb));  exit(1);  }  #else  char \*\*result;/\*用来保存结果\*/  int row,column;/\*行和列\*/  ret = sqlite3\_get\_table(ppdb,sql,&result,&row,&column,NULL);  if(ret != SQLITE\_OK)  {  printf("sqlite3\_get\_table:%s\n",sqlite3\_errmsg(ppdb));  exit(1);  }  int i,j;  int index = column;  for(i = 0;i < row;i++)/\*row行\*/  {  for(j = 0;j < column;j++)/\*column列\*/  {  printf("%s\t = %s\t",result[j],result[index]);  index++;  }  printf("\n");  }  #endif  return 0;  } |

这里重点讲一下sqlite3\_get\_table(ppdb,sql,&result,&row,&column,NULL)函数

函数描述：获取数据库中的某个表中的存储内容信息。

参数描述：1、ppdb：数据库指针

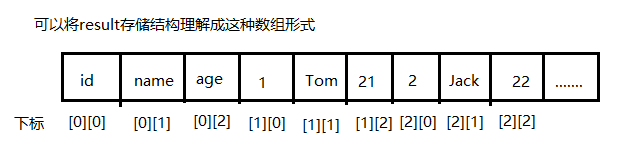
1. sql：其原型为

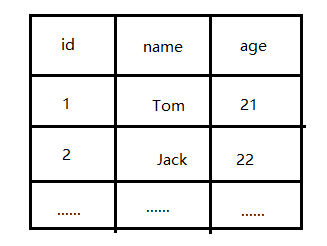
char sql[128] = {0};

sprintf(sql,"select \* from student;");

这个参数里存放了一条数据库指令，其指令就是”select \* from student;”

1. &result：用来存放最终结果





1. &row：表有多少行
2. &column：表有多少列
3. NULL：这里本来是存放错误信息的，也可以指定为NULL