[**SQLite数据库简介和使用**](http://www.cnblogs.com/zxf-rxf/p/3166011.html)

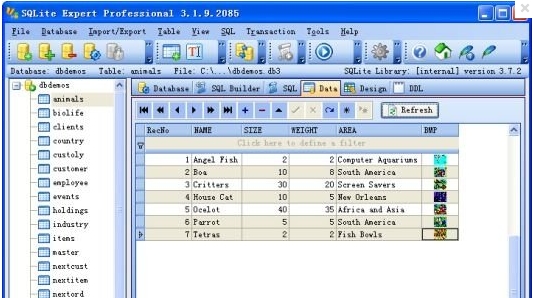
**一、Sqlite简介：**   
　　SQLite (http://www.sqlite.org/)，是一款轻型的数据库，是遵守ACID的关联式数据库管理系统，它的设计目标是嵌入式的，而且目前已经在很多嵌入式产品中使用了它，它占用资源非常的低，在嵌入式设备中，可能只需要几百K的内存就够了。它能够支持Windows/Linux/Unix等等主流的操作系统，同时能够跟很多程序语言相结合，比如 Tcl、C#、PHP、Java等，还有ODBC接口，同样比起Mysql、PostgreSQL这两款开源世界著名的数据库管理系统来讲，它的处理速度比他们都快。SQLite第一个Alpha版本诞生于2000年5月. 至今已经有10个年头，SQLite也迎来了一个版本 SQLite 3已经发布。

**二、Sqlite作者赠言：**  
　　o May you do good and not evil. 愿你行善莫行恶   
　　o May you find forgiveness for yourself and forgive others. 愿你原谅自己宽恕他人   
　　o May you share freely, never taking more than you give. 愿你宽心与人分享，所取不多于你所施予

**三、Sqlite的命令：**  
　　sqlite3 too.db 创建名为too的数据库，其后缀不一定用db  
　　.Help 求助   
　　.quit 离开

**四、Sqlite的客户端工具：**

　　SQLiteExpertPers



**六、Sqlite的sql语句：**

建表：create table table\_name(field1, field2, field3, ...);   
　　例子：创建名为film的数据库表  
　　　　create table film(\_id Integer primaray key autoincrement , title, length, year, starring);   
　　注意：语句要以分号结尾，字段不用指定类型，它会适时的自动转换  
　　　　可以存储文字、数字、大文本(blub)

创建索引：create index index\_name on table\_name(field\_to\_be\_indexed);  
　　例子：针对表film的title字段创建名为film\_title\_index的索引  
　　　　create index film\_title\_index on film(title);  
　　注意：当表的数据较多时，索引能加快查询速度(前提是根据建有索引的字段查询)

添加数据：insert into table\_name(field1,field2,field3,...) values(data1, data2, data3, ...);  
　　例子：向表film中添加一条记录  
　　　　insert into film(title, length, year, starring) values('Contact',153,1997,'Jodie Foster');  
　　注意：可以省略语句中的字段名部分，前提是数据个数与字段个数一样  
　　　　如果某个字段没有添加值则其值为null,也可手动添加null值

查询数据：select columns from table\_name where expression;  
　　例子：从表film中查询数据  
　　　　1 显示表里所有字段的所有数据   
　　　　　　select \* from film;   
　　　　2 如果资料太多了，我们或许会想限制笔数：   
　　　　　　select \* from film limit 10;   
　　　　3 照着电影年份来排列：   
　　　　　　select \* from film order by year limit 10;   
　　　　4 年份比较近的电影先列出来：   
　　　　　　select \* from film order by year desc limit 10;   
　　　　5 我们只想看电影名称跟年份：   
　　　　　　select title, year from film order by year desc limit 10;   
　　　　6 查所有茱蒂佛斯特演过的电影：   
　　　　　　select \* from film where starring='Jodie Foster';   
　　　　7 查所有演员名字开头叫茱蒂的电影('%' 符号便是 SQL 的万用字符）：   
　　　　　　select \* from film where starring like 'Jodie%';   
　　　　8 查所有演员名字以茱蒂开头、年份晚于1985年、年份晚的优先列出、最多十笔，只列出电影名称和年份：   
　　　　　　select title, year from film where starring like 'Jodie%' and year >= 1985   
　　　　　　order by year desc limit 10;   
　　　　9 查看数据库一共有多少条记录：   
　　　　　　select count(\*) from film;   
　　　　10 查看1985年以后的电影有几部：   
　　　　　　select count(\*) from film where year >= 1985;

更新数据：update film set starring='Jodie Foster' where starring='Jodee Foster';   
　　把主角字段为'Jodee Foster'的所有记录改成Jodie Foster。

删除数据：delete from film where year < 1970;   
　　删除所有年代早于1970 年（不含）的电影记录

注释：注释单行：--  
　　注释多行：/\* \*/

创建视图：CREATE VIEW view-name AS select-statement

模糊匹配：like %  
  
sqlite日间日期函数：  
　　datetime() 产生日期和时间  
　　date() 产生日期  
　　time() 产生时间  
　　strftime() 对以上三个函数产生的日期和时间格式化  
　　可用的字符串参数：  
　　now 产生现在的时间  
　　YYYY-MM-DD  
　　YYYY-MM-DD HH:MM  
　　YYYY-MM-DD HH:MM:SS  
　　YYYY-MM-DD HH:MM:SS.SSS  
　　HH:MM  
　　HH:MM:SS  
　　HH:MM:SS.SSS  
　　例子：  
　　　　select datetime('now');  
　　　　select datetime('2011-06-12');  
　　　　select datetime('2006-10-17 00:20:00','+1 hour','-12 minute');  
　　　　select date('2006-10-17','+1 day','+1 year');  
　　　　select datetime('now', 'localtime');

--修改表结构  
　　--添加一个字段   
　　　　alter table film add column director2;   
　　--删除一个字段 不行  
　　　　alter table film drop (column director2);

*--删除一个表*

*drop table test;*

sqlite的特别用法   
　　sqlite可以在shell底下直接执行命令：   
　　　　输出 HTML 表格： sqlite3 -html film.db "select \* from film;"   
　　　　将数据库「倒出来」： sqlite3 film.db ".dump" > output.sql   
　　　　利用输出的资料，建立一个一模一样的数据库（加上以上指令，就是标准的SQL数据库备份了）：   
　　　　sqlite3 film.db < output.sql   
　　　　在大量插入资料时，你可能会需要先打这个指令： begin;   
　　　　插入完资料后要记得打这个指令，资料才会写进数据库中： commit;

**七、练习：**  
　　--创建雇员表  
　　　　CREATE TABLE EMPLOYEES(  
　　　　　　employee\_id Integer PRIMARY KEY,   
　　　　　　department\_id Integer,   
　　　　　　location\_id Integer,   
　　　　　　first\_name,   
　　　　　　last\_name,   
　　　　　　salary,  
　　　　　　hire\_date date   
　　　　);  
　　--创建部门表  
　　　　CREATE TABLE DEPARTMENT(  
　　　　　　department\_id Integer primary key,   
　　　　　　name  
　　　　);  
　　创建职位表  
　　　　CREATE TABLE LOCATION (  
　　　　　　location\_id Integer PRIMARY KEY,   
　　　　　　name  
　　　　);

　　添加测试数据  
　　　　insert into [employees](department\_id,location\_id , first\_name, last\_name, salary,hire\_date)  
　　　　　　values (1,1, 'A', 'z', 50000, '2005-02-21');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (1,2, 'B', 'x', 20000, '2009-03-21');   
　　　　insert into [employees](department\_id,location\_id , first\_name, last\_name, salary,hire\_date)  
　　　　　　values (2,3, 'C', 'v', 10000, '2009-08-23');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (4,2, 'D', 'n', 30000, '2004-09-28');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (3,5, 'E', 'm', 3000, '2009-04-11');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (5,5, 'F', 'l', 5000, '2008-03-11');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (5,3, 'G', 'p', 20000, '2005-05-09');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (5,4, 'H', 'o', 8000, '2006-07-21');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (5,5, 'I', 'u', 6000, '2006-09-21');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (5,5, 'J', 'y', 5500, '2007-08-21');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (5,5, 'K', 't', 6500, '2006-12-21');   
　　　　insert into [employees](department\_id,location\_id, first\_name, last\_name, salary,hire\_date)  
　　　　　　values (5,1, 'L', 'r', 100000, '2001-05-21');   
  
insert into department(name) values ('人事');  
　　　　insert into department(name) values ('财务');  
　　　　insert into department(name) values ('后勤');  
　　　　insert into department(name) values ('公关');  
　　　　insert into department(name) values ('研发');

　　　　insert into location(name) values ('总经理');  
　　　　insert into location(name) values ('经理');  
　　　　insert into location(name) values ('主管');  
　　　　insert into location(name) values ('组长');  
　　　　insert into location(name) values ('职员');

　　--查研发部的职员的员工信息  
　　　　select \* from employees e where e.location\_id =(select l.location\_id from location l where l.name='职员')  
　　　　and  
　　　　e.[department\_id]=(select d.department\_id from department d where d.name='研发');

　　--根据查询结果创建一个表  
　　　　create table TEMP\_EMPLOYEES AS select employee\_id, first\_name, last\_name from EMPLOYEES where salary>6000;

　　--查询可以进行计算  
　　　　select salary\*13 年薪 from employees where 年薪 !=260000;  
　　　　select salary\*13 年薪 from employees where 年薪 between 50000 and 100000;  
　　--first\_name 是A, B, C的职员信息  
　　　　select \* from employees where first\_name in ('A', 'B', 'C');

　　--测试is null  
　　　　select \* from film where title is null;

　　--查询月薪大于10000的主管  
　　　　select \* from employees where salary>10000 and location\_id=3;

　　--查询月薪大于10000的主管或者理解  
　　　　select \* from employees where salary>10000 and (location\_id=3 or location\_id=2);  
　　　　select \* from employees where salary>10000 and location\_id not in(4,5,1);

　　--测试order by  
　　　　select \* from employees order by location\_id,salary;