Sqlite3 使用教程

OS X 自从 10.4 后把 SQLite 这套相当出名的数据库软件，放进了作业系统工具集里。OS X 包装的是第三版的 SQLite，又称 SQLite3。这套软件有几个特色：

- 软件属于公共财（public domain），SQLite 可说是某种「美德软件」（virtueware），作者本人放弃着作权，而给使用 SQLite 的人以下的「祝福」（blessing）：
    - May you do good and not evil. 愿你行善莫行恶
    - May you find forgiveness for yourself and forgive others. 愿你原谅自己宽恕他人
    - 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
- 支援大多数的 SQL 指令（下面会简单介绍）。
- 一个档案就是一个数据库。不需要安装数据库服务器软件。
- 完整的 Unicode 支援（因此没有跨语系的问题）。
- 速度很快。

目前在 OS X 10.4 里，SQLite 是以/usr/bin/sqlite3 的形式包装，也就说这是一个命令列工具，必须先从终端机（Terminal.app 或其他程序）进入 shell 之后才能使用。网络上有一些息协助使用 SQLite 的视觉化工具，但似乎都没有像 CocoaMySQL（配合 MySQL 数据库使用）那般好用。或许随时有惊喜也未可知，以下仅介绍命令列的操作方式。

SQLite 顾名思议是以 SQL 为基础的数据库软件，SQL 是一套强大的数据库语言，主要概念是由「数据库」、「资料表」（table）、「查询指令」（queries）等单元组成的「关联性数据库」（进一步的概念可参考网络上各种关于 SQL 及关联性数据库的文件）。因为 SQL 的查询功能强大，语法一致而入门容易，因此成为现今主流数据库的标准语言（微软、Oracle 等大厂的数据库软件都提供 SQL 语法的查询及操作）。

以下我们就建立数据库、建立资料表及索引、新增资料、查询资料、更改资料、移除资料、sqlite3 命令列选项等几个项目做简单的介绍。

目录

- 1 建立数据库档案
- 2 在 sqlite3 提示列下操作
- 3 SQL 的指令格式
- 4 建立资料表
- 5 建立索引
- 6 加入一笔资料
- 7 查询资料
- 8 如何更改或删除资料
- 9 其他 sqlite 的特别用法
- 10 小结

建立数据库档案

用 sqlite3 建立数据库的方法很简单，只要在 shell 下键入（以下$符号为 shell 提示号，请勿键入）：

```
$ sqlite3 foo.db
```

如果目录下没有 foo.db，sqlite3 就会建立这个数据库。sqlite3 并没有强制数据库档名要怎么取，因此如果你喜欢，也可以取个例如 foo.icannameitwhateverilike 的档名。

在 sqlite3 提示列下操作

进入了 sqlite3 之后，会看到以下文字：

```
SQLite version 3.1.3
Enter ".help" for instructions
sqlite>
```

这时如果使用.help 可以取得求助，.quit 则是离开（请注意：不是 quit）

SQL 的指令格式

所以的 SQL 指令都是以分号（;）结尾的。如果遇到两个减号（--）则代表注解，sqlite3 会略过去。

建立资料表

假设我们要建一个名叫 film 的资料表，只要键入以下指令就可以了：

```
create table film(title, length, year, starring);
```

这样我们就建立了一个名叫 film 的资料表，里面有 name、length、year、starring 四个字段。

这个 create table 指令的语法为：

```
create table table_name(field1, field2, field3, ...);
```

table_name 是资料表的名称，fieldx 则是字段的名字。sqlite3 与许多 SQL 数据库软件不同的是，它不在乎字段属于哪一种资料型态：sqlite3 的字段可以储存任何东西：文字、数字、大量文字（blub），它会在适时自动转换。

建立索引

如果资料表有相当多的资料，我们便会建立索引来加快速度。好比说：

```
create index film_title_index on film(title);
```

意思是针对 film 资料表的 name 字段，建立一个名叫 film_name_index 的索引。这个指令的语法为

```
create index index_name on table_name(field_to_be_indexed);
```

一旦建立了索引，sqlite3 会在针对该字段作查询时，自动使用该索引。这一切的操作都是在幕后自动发生的，无须使用者特别指令。

加入一笔资料

接下来我们要加入资料了，加入的方法为使用 insert into 指令，语法为：

```
insert into table_name values(data1, data2, data3, ...);
```

例如我们可以加入

```
insert into film values ('Silence of the Lambs, The', 118, 1991, 'Jodie Foster');
insert into film values ('Contact', 153, 1997, 'Jodie Foster');
insert into film values ('Crouching Tiger, Hidden Dragon', 120, 2000, 'Yun-Fat Chow');
insert into film values ('Hours, The', 114, 2002, 'Nicole Kidman');
```

如果该字段没有资料，我们可以填 NULL。

查询资料

讲到这里，我们终于要开始介绍 SQL 最强大的 select 指令了。我们首先简单介绍 select 的基本句型：

```
select columns from table_name where expression;
```

最常见的用法，当然是倒出所有数据库的内容：

```
select * from film;
```

如果资料太多了，我们或许会想限制笔数：

```
select * from film limit 10;
```

或是照着电影年份来排列：

```
select * from film order by year limit 10;
```

或是年份比较近的电影先列出来：

```
select * from film order by year desc limit 10;
```

或是我们只想看电影名称跟年份：

```
select title, year from film order by year desc limit 10;
```

查所有茱蒂佛斯特演过的电影：

```
select * from film where starring='Jodie Foster';
```

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%';
```

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;
```

有时候我们只想知道数据库一共有多少笔资料：

```
select count(*) from film;
```

有时候我们只想知道 1985 年以后的电影有几部：

```
select count(*) from film where year >= 1985;
```

（进一步的各种组合，要去看 SQL 专书，不过你大概已经知道 SQL 为什么这么流行了：这种语言允许你将各种查询条件组合在一起——而我们还没提到「跨数据库的联合查询」呢！）

如何更改或删除资料

了解 select 的用法非常重要，因为要在 sqlite 更改或删除一笔资料，也是靠同样的语法。

例如有一笔资料的名字打错了：

```
update film set starring='Jodie Foster' where starring='Jodee Foster';
```

就会把主角字段里，被打成'Jodee Foster'的那笔（或多笔）资料，改回成 Jodie Foster。

```
delete from film where year < 1970;
```

就会删除所有年代早于 1970 年（不含）的电影了。

其他 sqlite 的特别用法

sqlite 可以在 shell 底下直接执行命令：

```
sqlite3 film.db "select * from film;"
```

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;
```

小结

以上我们介绍了 SQLite 这套数据库系统的用法。事实上 OS X 也有诸于 SQLiteManagerX 这类的图形接口程序，可以便利数据库的操作。不过万变不离其宗，了解 SQL 指令操作，SQLite 与其各家变种就很容易上手了。

至于为什么要写这篇教学呢？除了因为 OS X Tiger 大量使用 SQLite 之外（例如：Safari 的 RSS reader，就是把文章存在 SQLite 数据库里！你可以开开看~/Library/Syndication/Database3 这个档案，看看里面有什么料），OpenVanilla 从 0.7.2 开始，也引进了以 SQLite 为基础的词汇管理工具，以及全字库的注音输入法。因为使用 SQLite，这两个模块不管数据库内有多少笔资料，都可以做到「瞬间启动」以及相当快速的查询回应。

将一套方便好用的数据库软件包进 OS X 中，当然也算是 Apple 相当相当聪明的选择。再勤劳一点的朋友也许已经开始想拿 SQLite 来记录各种东西（像我们其中就有一人写了个程序，自动记录电池状态，写进 SQLite 数据库中再做统计.....）了。想像空间可说相当宽广。

目前支援 SQLite 的程序语言，你能想到的大概都有了。这套数据库 2005 年还赢得了美国 O'Reilly Open Source Conference 的最佳开放源代码软件奖，奖评是「有什么东西能让 Perl, Python, PHP, Ruby 语言团结一致地支援的？就是 SQLite」。由此可见 SQLite 的地位了。而 SQLite 程序非常小，更是少数打 "gcc -o sqlite3 *"，不需任何特殊设定就能跨平台编译的程序。小而省，小而美，SQLite 连网站都不多赘言，直指 SQL 语法精要及 API 使用方法，原作者大概也可以算是某种程序设计之道（Tao of Programming）里所说的至人了。