sqlite3命令行工具

sqlite3命令行工具是基于终端的SQLite软件，可以查询交互，并且可以多种格式显示结果。它也可以用在脚本中。

打开终端，键入如下命令：

```
~  sqlite3 test.db
SQLite version 3.8.5 2014-08-15 22:37:57
Enter ".help" for usage hints.
sqlite>
```

.help命令是sqlite3的元命令工具之一,它列出了sqlite3所有的元命令。

.exit 和 .quit 命令退出sqlite3会话。我们还可以使用组合键Ctrl + D 退出sqlite3。

.databases命令显示了连接数据库。

.tables表命令列出可用的表。

创建一个数据库

使用sqlite3命令行工具来创建一个新的数据库文件：

```
$ sqlite3 test.db
```

这里我们创建一个新的test.db数据库。如果文件存在，则打开数据库。

创建数据库表

```
CREATE TABLE Cars(Id INTEGER PRIMARY KEY, Name TEXT, Price INTEGER);
INSERT INTO "Cars" VALUES(1,'Audi',52642);
INSERT INTO "Cars" VALUES(2,'Mercedes',57127);
INSERT INTO "Cars" VALUES(3,'Skoda',9000);
INSERT INTO "Cars" VALUES(4,'Volvo',29000);
INSERT INTO "Cars" VALUES(5,'Bentley',350000);
INSERT INTO "Cars" VALUES(6,'Citroen',21000);
INSERT INTO "Cars" VALUES(7,'Hummer',41400);
INSERT INTO "Cars" VALUES(8,'Volkswagen',21600);
```

基本sqlite3元命令

```
sqlite> .tables
Cars
```

.tables命令显示可用的表。

```
sqlite> SELECT * FROM cars;
1|Audi|52642
2|Mercedes|57127
3|Skoda|9000
4|Volvo|29000
5|Bentley|350000
6|Citroen|21000
7|Hummer|41400
8|Volkswagen|21600
```

SELECT查询语句的输出默认情况下是按行输出，行分隔符是`|`。

```
sqlite> .separator :
sqlite> SELECT * FROM cars;
1:Audi:52642
2:Mercedes:57127
3:Skoda:9000
4:Volvo:29000
5:Bentley:350000
6:Citroen:21000
7:Hummer:41400
8:Volkswagen:21600
```

上面使用了新的分隔符`:`

还有其他几个可用的输出模式。下面的示例显示列的输出模式：

```
sqlite> .mode column
sqlite> .headers on
sqlite> SELECT * FROM cars;
Id          Name        Price
----------  ----------  ----------
1           Audi        52642
2           Mercedes    57127
3           Skoda       9000
4           Volvo       29000
5           Bentley     350000
6           Citroen     21000
7           Hummer      41400
8           Volkswagen  21600
```

```
sqlite> .width 15 18
sqlite> SELECT Name, Price FROM cars;
Name             Price
---------------  ------------------
Audi             52642
Mercedes         57127
Skoda            9000
Volvo            29000
Bentley          350000
Citroen          21000
Hummer           41400
Volkswagen       21600
```

在这里，我们改变列宽。第一列将15个字符宽,第二列是18。

```
sqlite> .show
echo: off
      eqp: off
  explain: off
  headers: on
     mode: column
nullvalue: ""
   output: stdout
separator: ":"
    stats: off
    width: 15 18
```

.show 命令列出各种设置。这些包括输出模式，在列表模式下使用的分隔符和是否显示头部。

```
sqlite> .schema Cars
CREATE TABLE Cars(Id INTEGER PRIMARY KEY, Name TEXT, Price INTEGER);
```

.schema 命令显示表的结构。它返回创建表的DDL SQL。

.prompt 命令可以改变的sqlite3的提示。

```
sqlite> .prompt "> " ". "
> SELECT * FROM Cars
. LIMIT 2;
Id               Name                Price
---------------  ------------------  ----------
1                Audi                52642
2                Mercedes            57127
```

有两个提示。最主要是第一个提示，另一个是继续提示。默认的主提示符是'sqlite>' 和默认的继续提示'...>'。

从shell执行SQL

```
$ sqlite3 test.db "SELECT * FROM Cars;"
```

Dumping tables

我们将使用.dump命令来转储。

```
sqlite> .dump Cars
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE Cars(Id INTEGER PRIMARY KEY, Name TEXT, Price INTEGER);
INSERT INTO "Cars" VALUES(1,'Audi',52642);
INSERT INTO "Cars" VALUES(2,'Mercedes',57127);
INSERT INTO "Cars" VALUES(3,'Skoda',9000);
INSERT INTO "Cars" VALUES(4,'Volvo',29000);
INSERT INTO "Cars" VALUES(5,'Bentley',350000);
INSERT INTO "Cars" VALUES(6,'Citroen',21000);
INSERT INTO "Cars" VALUES(7,'Hummer',41400);
INSERT INTO "Cars" VALUES(8,'Volkswagen',21600);
COMMIT;
```

.dump 命令显示了我们所必需的SQL创建表的信息。

```
sqlite> .output cars2.sql
sqlite> .dump Cars
```

我们也可以将输出重定向到一个文件中。

.output命令会将输出重定向cars2.sql文件。

Reading SQL

我们可以从一个文件名读取SQL语句 `.read` 命令。

```
sqlite> .tables Cars
Cars
sqlite> DROP TABLE Cars;
sqlite> .tables Cars
sqlite> .read cars.sql
sqlite> .tables Cars
Cars
sqlite> SELECT * FROM Cars WHERE Id=1;
Id          Name        Price
----------  ----------  ----------
1           Audi        52642
```

The .sqlite_history file

命令和语句中存档于 .sqlite_history文件。该文件位于主目录。

```
$ tail -5 ~/.sqlite_history
```

使用tail命令,我们显示最后五项。

Resource file

sqlite3 工具有一个资源文件 .sqliterc 。 它位于主目录。如果没有这些文件，我们可以简单地创建它。 这些文件可以包含元命令，或常规的SQL语句。

```
$ cat .sqliterc
.mode column
.header on
.nullvalue NULL
```

这里是一个简单的资源文件的例子，它有三个元命令。有了资源文件，我们不需要再次执行已有的元命令。当我们开始 sqlite3 工具，他们将自动执行。