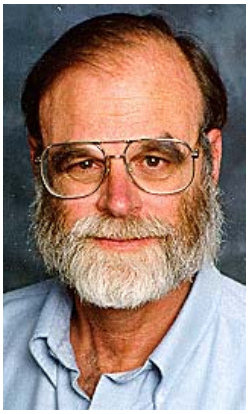
YY哥的技术随笔

——关注Linux、数据库和云计算



个人简介
专业打杂程序员
联系方式
新浪微博 腾讯微博

昵称：YY哥
园龄：7年2个月
粉丝：342
关注：2
+加关注

< 2009年2月 >

日	一	二	三	四	五	六
25	26	27	28	29	30	31
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
1	2	3	4	5	6	7

搜索
找找看
谷歌搜索

常用链接
我的随笔
我的评论
我的参与
最新评论
我的标签
更多链接

随笔分类
c/c++(9)
Linux相关(24)
MySQL(11)
Others(2)
Web技术(12)
数据结构与算法(15)
数据库技术(30)
系统相关(3)
云计算与虚拟化(3)

随笔档案
2014年7月 (4)

SQLite入门与分析(三)---内核概述(1)

写在前面:从本章开始,我们开始进入SQLite的内核。为了能更好的理解SQLite，我先从总的结构上讨论一下内核，从全局把握SQLite很重要。SQLite的内核实现不是很难，但是也不是很简单。总的来说分为三个部分，本章主要讨论虚拟机(Virtual Machine)，但是这里只是从原理上概述，不会太多的涉及实际代码。但是概述完内核之后会仔细讨论源代码的。好了，下面我们来讨论虚拟机(VM)。

1、虚拟机（Virtual Machine）

VDBE是SQLite的核心，它的上层模块和下层模块都是本质上都是为它服务的。它的实现位于vbde.c, vdbe.h, vdbeapi.c, vdbeInt.h, 和vdbemem.c几个文件中。它通过底层的基础设施B+Tree执行由编译器（Compiler）生成的字节代码，这种字节代码程序语言(bytecode programming lauguage)是为了进行查询，读取和修改数据库而专门设计的。
字节代码在内存中被封装成sqlite3_stmt对象(内部叫做Vdbe，见vdbeInt.h)，Vdbe（或者说statement）包含执行程序所需要的一切：

a) a bytecode program
b) names and data types for all result columns
c) values bound to input parameters
d) a program counter
e) an execution stack of operands
f) an arbitrary amount of "numbered" memory cells
g) other run-time state information (such as open BTree objects, sorters, lists, sets)

字节代码和汇编程序十分类似，每一条指令由操作码和三个操作数构成：<opcode, P1, P2, P3>。Opcode为一定功能的操作码，为了理解，可以看成一个函数。P1是32位的有符号整数，p2是31位的无符号整数，它通常是导致跳转(jump)的指令的目标地址（destination），当然这了有其它用途；p3为一个以null结尾的字符串或者其它结构体的指针。和C API不同的是，VDBE操作码经常变化，所以不应该用字节码写程序。
下面的几个C API直接和VDBE交互：

- sqlite3_bind_xxx() functions
- sqlite3_step()
- sqlite3_reset()
- sqlite3_column_xxx() functions
- sqlite3_finalize()

为了有个感性，下面看一个具体的字节码程序：

```
sqlite> .m col
sqlite> .h on
sqlite> .w 4 15 3 3 15
sqlite> explain select * from episodes;
addr  opcode         p1   p2   p3
----  -------------- ---  ---  ----------
0     Goto           0    12
1     Integer        0    0
2     OpenRead       0    2    # episodes
3     SetNumColumns  0    3
4     Rewind         0    10
5     Recno          0    0
6     Column         0    1
7     Column         0    2
8     Callback       3    0
9     Next           0    5
10    Close          0    0
11    Halt           0    0
12    Transaction    0    0
13    VerifyCookie   0    10
14    Goto           0    1
15    Noop           0    0
```

1.1、 栈(Stack)

一个VDBE程序通常由不同完成特定任务的段（section）构成，每一个段中，都有一些操作栈的指令。这是由于不同的指令有不同个数的参数，一些指令只有一个参数；一些指令没有参数；一些指令有好几个参数，这种情况下，三个操作数就不能满足。
考虑到这些情况，指令采用栈来传递参数。(注：从汇编的角度来看，传递参数的方式有好几种，比如：寄存器，全局变量，而堆栈是现代语言常用的方式，它具有很大的灵活性)。而这些指令不会自己做这些事情，所以在它们之前，需要其它一些指令的帮助。VDBE把计算的中间结果保存到内存单元(memory cells)中，其实，堆栈和内存单元都是基于Mem（见vdbeInt.h）数据结构(注：这里的栈，内存单元都是虚拟的，记得一位计算机科学家说过：计算机科学中90%以上的科学都是虚拟化问题。一点不假，OS本质上也是虚拟机，而在这里SQLite，我们也处处可见虚拟化的身影，到后面的OS Interface模块中再仔细讨论这个问题)。

1.2、程序体(Program Body)

这是一个打开episodes表的过程。
第一条指令：Integer是为第二条指令作准备的，也就是把第二条指令执行需要的参数压入堆栈，OpenRead从堆栈中取出参数值然后执行。SQLite可以通过ATTACH命令在一个连接中打开多个数据库文件，每当SQLite打开一个数据，它就为之赋一个索引号(

2014年3月 (1)
2013年9月 (1)
2013年8月 (1)
2013年2月 (1)
2012年11月 (4)
2012年1月 (1)
2011年12月 (1)
2011年10月 (1)
2011年3月 (1)
2010年9月 (1)
2010年8月 (1)
2010年7月 (3)
2010年6月 (2)
2010年5月 (7)
2010年4月 (1)
2010年3月 (1)
2010年1月 (1)
2009年12月 (2)
2009年10月 (2)
2009年9月 (14)
2009年8月 (4)
2009年6月 (14)
2009年5月 (3)
2009年4月 (1)
2009年3月 (3)
2009年2月 (11)
2008年10月 (7)
2008年8月 (5)
2008年7月 (1)
2008年6月 (2)
2008年5月 (2)
2008年4月 (5)

kernel

kernel中文社区
LDN
The Linux Document Project
The Linux Kernel Archives

manual

cppreference
gcc manual
mysql manual

sites

Database Journal
Fedora镜象
highscalability
KFUPM ePrints
Linux docs
Linux Journal
NoSQL
SQLite

技术社区

apache
CSDN
IBM-developerworks
lucene中国
nutch中国
oldlinux
oracle's forum

最新评论

1. Re:理解MySQL——架构与概念
我试验了下.数据 5 9 10 13 18begin;select * from asf_execution where num> 5 and num 5 and INSTANCE_ID_<18 lock in share mode;会有 1.行锁 2.间隙所 [5 18)插

index)，main database的索引为0，第一个数据库为1，依次如此。Integer指令数据库索引的值压入栈，而OpenRead从中取出值，并决定打开哪个数据，来看看SQLite文档中的解释：

Open a read-only cursor for the database table whose root page is P2 in a database file.
The database file is determined by an integer from the top of the stack. 0 means the main database and 1 means the database used for temporary tables. Give the new cursor an identifier of P1. The P1 values need not be contiguous but all P1 values should be small integers. It is an error for P1 to be negative.

If P2==0 then take the root page number from off of the stack.

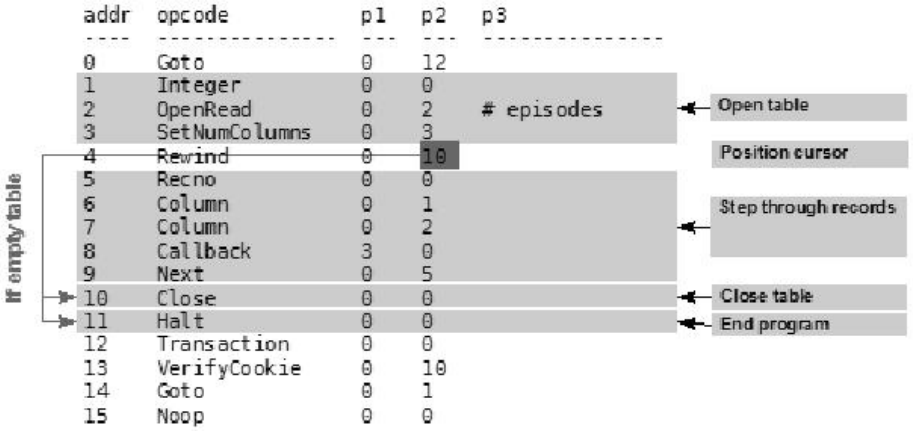
There will be a read lock on the database whenever there is an open cursor. If the database was unlocked prior to this instruction then a read lock is acquired as part of this instruction. A read lock allows other processes to read the database but prohibits any other process from modifying the database. The read lock is released when all cursors are closed. If this instruction attempts to get a read lock but fails, the script terminates with an SQLITE_BUSY error code.

The P3 value is a pointer to a KeyInfo structure that defines the content and collating

sequence of indices. P3 is NULL for cursors that are not pointing to indices.

再来看看SetNumColumns指令设置游标将指向的列。P1为游标的索引（这里为0，刚刚打开），P2为列的数目，episodes表有三列。

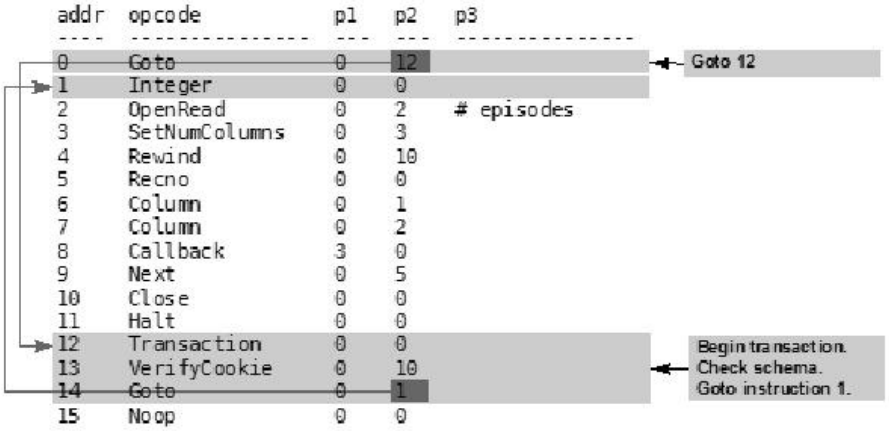
继续Rewind指令，它将游标重新设置到表的开始，它会检查表是否为空（即没有记录），如果没有记录，它会导致指令指针跳到P2指定的指令处。在这里，P2为10，即Close指令。一旦Rewind设置游标，接下就执行5-9这几条指令，它们的主要功能是遍历结果集，Recno把由游标P1指定的记录的关键字压入堆栈。Column指令从由P1指定的游标，P2指定的列取值。5,6,7三条指令分别把id(primary key),season和name字段的值压入栈。接下来，Callback指令从栈中取出三个值（P1），然后形成一个记录数组，存储在内存单元中(memory cell)。Callback会停止VDBE的操作，把控制权交给sqlite3_stemp()，该函数返回SQLITE_ROW。



一旦VDBE创建了记录结构，我们就可以通过sqlite3_column_xxx() functions从记录结构的域内取出值。当下次调用sqlite3_step()时，指令指针会指向Next指令，而Next指令会把游标向移向下一行，如果有其它的记录，它会跳到由P2指定的指令，在这里为指令5，创建一个新的记录结构，一直循环，直到结果集的最后。Close指令会关闭游标，然后执行Halt指令，结束VDBE程序。

1.3、程序开始与停止

现在来看看其余的指令，Goto指令是一条跳转指令，跳到P2处，即第12条指令。指令12是Transaction，它开始一个新的事务；然后执行VerifyCookie，它的主要功能VDBE程序编译后，数据库模式是否改变（即是否进行过更新操作）。这在SQLite中是一个很重要的概念，在SQL被sqlite3_prepare()编译成VDBE代码至程序调用sqlite3_step()执行字节码的这段时间，另一个SQL命令可能会改变数据库模式（such as ALTER TABLE, DROP TABLE, or CREATE TABLE）。一旦发生这种情况，之前编译的statement就会变得无效，数据库模式信息记录在数据库文件的根页面中。类似，每一个statement都有一份用来比较的在编译时刻该模式的备份，VerifyCookie的功能就是检查它们是否匹配，如果不匹配，将采取相关操作。



如果两者匹配，会执行下一条指令Goto；它会跳到程序的主要部分，即第一条指令，打开表读取记录。这里有两点值得注意：
(1)Transaction指令自己不会获取锁（ The Transaction instruction doesn't acquire any locks in itself）。它的功能相当于BEGIN，而实际是由OpenRead指令获取share lock的。当事务关闭时释放锁，这取决于Halt指令，它会进行扫尾工作。
(2)statement对象（VDBE程序）所需的存储空间在程序执行前就已经确定。这有原于两个重要事实：首先，栈的深度不会比指令的数目还多（通常少得多）。其次，在执行VDBE程序之前，SQLite可以计算出为分配资源所需要的内存。

1.4指令的类型(Instruction Types)

每条指令都完成特定的任务，而且通常和别的指令有关。大体上来说，指令可分为三类：
（1)Value manipulation：这些指令通常完成算术运算，比如：add, subtract, divide；逻辑运算，比如：AND和OR；还有字符串操作。

入INSERT I.....
--麒麟飞
2. Re:理解MySQL——架构与概念
例1-5
insert into t(i) values(1);
这句话应该是可以插入的.
不会被阻塞
--麒麟飞
3. Re:理解MySQL——架构与概念
注：SELECT ... FOR UPDATE仅在自动提交关闭(即手动提交)时才会对元组加锁，而在自动提交时，符合条件的元组不会被加锁。

这个是错误的.自动提交的,也会尝试获取排它锁.
你可以试验下.
--麒麟飞
4. Re:浅谈mysql的两阶段提交协议
YY哥 偶像啊!细腻文笔 配有说服力的代码和图 我崇拜你 !!!
之前sqlite的深入分析帮了我大忙..现在做mysql相关 有来你的博客找东西 哈哈哈!!
--hark.perfe
5. Re:(i++)+(i++)与(++i)+(++i)
@arrowcat
这类语句本身没什么意义，但是楼主思考的角度让我豁然开朗。
--HJWAJ

阅读排行榜

1. 理解MySQL——索引与优化(77627)
2. SQLite入门与分析(一)---简介(48610)
3. 理解MySQL——复制(Replication)(26209)
4. libevent源码分析(19048)
5. SQLite入门与分析(二)---设计与概念(16977)

评论排行榜

1. (i++)+(i++)与(++i)+(++i)(40)
2. SQLite入门与分析(一)---简介(30)
3. 浅谈SQLite——实现与应用(20)
4. 一道算法题,求更好的解法(18)
5. 理解MySQL——索引与优化(16)

推荐排行榜

1. SQLite入门与分析(一)---简介(12)
2. 理解MySQL——索引与优化(12)
3. 浅谈SQLite——查询处理及优化(10)
4. 乱谈服务器编程(9)
5. libevent源码分析(6)

（2)Data management：这些指令操作在内存和磁盘上的数据。内存指令进行栈操作或者在内存单元之间传递数据。磁盘操作指令控制B-tree和pager打开或操作游标，开始或结束事务，等等。

（3)Control flow：控制指令主要是移动指令指针。

1.5、程序的执行(Program execution)

最后我们来看VM解释器是如何实现以及字节代码大致是如何执行的。在vdbe.c文件中有一个很关键的函数：

```
//执行VDBE程序
int sqlite3VdbeExec(
  Vdbe *p                    /* The VDBE */
)
```

该函数是执行VDBE程序的入口。来看看它的内部实现：

```
/*从这里开始执行指令
**pc为程序计数器(int)
*/
for(pc=p->pc; rc==SQLITE_OK; pc++){
  //取得操作码
  pOp = &p->aOp[pc];
  switch( pOp->opcode ){
  case OP_Goto: {           /* jump */
     CHECK_FOR_INTERRUPT;
     pc = pOp->p2 - 1;
     break;
    }
    ... ...
  }
}
```

从这段代码，我们大致可以推出VM执行的原理：VM解释器实际上是一个包含大量switch语句的for循环，每一个switch语句实现一个特定的操作指令。

分类: 数据库技术

YY哥
关注 - 2
粉丝 - 342
+加关注

« 上一篇：SQLite入门与分析(二)---设计与概念(续)
» 下一篇：SQLite入门与分析(三)---内核概述(2)

posted @ 2009-02-15 16:55 YY哥 阅读(11180) 评论(8) 编辑 收藏

评论列表

#1楼 2009-02-15 17:24 Welshem

不错啊！

支持(0) 反对(0)

#2楼 2009-02-15 21:01 梁逸晨

楼主请原谅我打开文章后就直接拉到底来回帖了，不是我不看正文，而是正文太高深了，暂时还没有能力看得懂，但是我本着无比尊敬楼主的态度为你顶贴

支持(0) 反对(0)

#3楼 2009-02-15 21:47 x.tsai[未注册用户]

http://www.sqlite3.org.cn SQLite3 中国组织转载了博主的文章 非常感谢分享

#4楼[楼主] 2009-02-15 22:21 YY哥

@梁逸晨
谢谢支持。其实这些也不是什么高深的东西，只要学过汇编，再看看相关的文档，我想是应该可以读懂的。

支持(0) 反对(0)

#5楼[楼主] 2009-02-15 22:27 YY哥

@x.tsai
欢迎转载.

支持(0) 反对(0)

#6楼 2009-03-01 16:25 Soli

很好哈！

支持(0) 反对(0)

#7楼 2009-03-01 22:21 Soli

Not bad!

支持(0) 反对(0)

#8楼 2011-11-12 12:20 cugfsx

感谢楼主的讲解，我学习中，争取能理解完整，对我的项目有帮助，非常感谢!

支持(0) 反对(0)