**第二部分 数据库机器(database machine)**

1,数据流

在本项目中，数据库机是实际操作数据库文件的部分，DBM相当于一个cpu，用于逐条处理dbm指令，操作寄存器和游标来返回sql语句的结果。

dbm对用户透明，用户使用SQL操作数据库，而真正操作具体数据的是dbm指令。

其输入输出如图2-1-1所示

图2-1-1

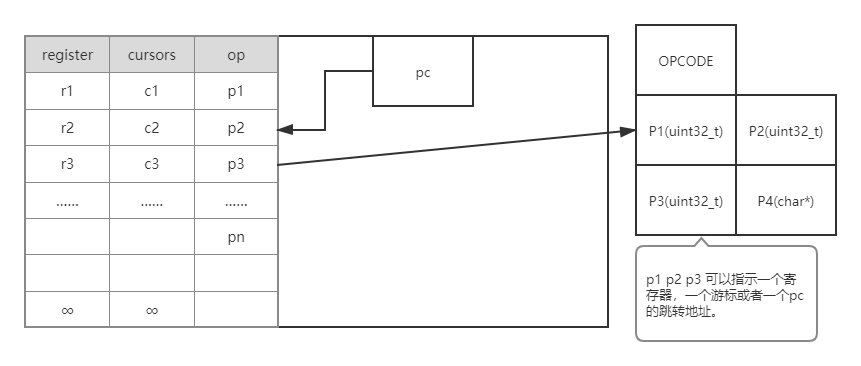


2.dbm中的数据结构,

如图2-2-1所示，一个dbm中含有无穷的寄存器和游标，在dbm运行时pc从op这个数组中逐条取出命令并执行，每条指令都可能对某个寄存器或游标进行操作

1. 寄存器可以储存整数，字符串或者无类型的比特串
2. 游标指向某个表的具体项，并且可以操作cursor快速移动，

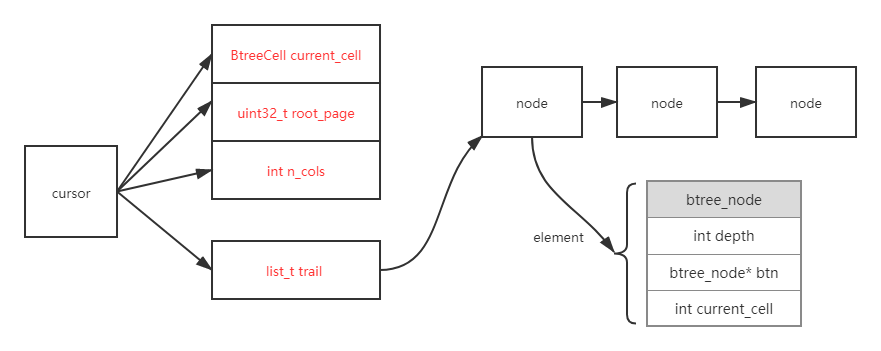
图2-2-1



游标的结构如图2-2-2所示，每个游标都具体指向表中某元组(current\_cell),并且储存该表对应B树的根节点(root\_page)和表所含列数(n\_cols)

最后是一个用于记录从根节点到含有该游标所指元组的路径的一个链表trail，在trail中，其元素记录了当前B树深度(depth),所在的B树节点以及用于指示路径的current\_cell标识

图2-2-2

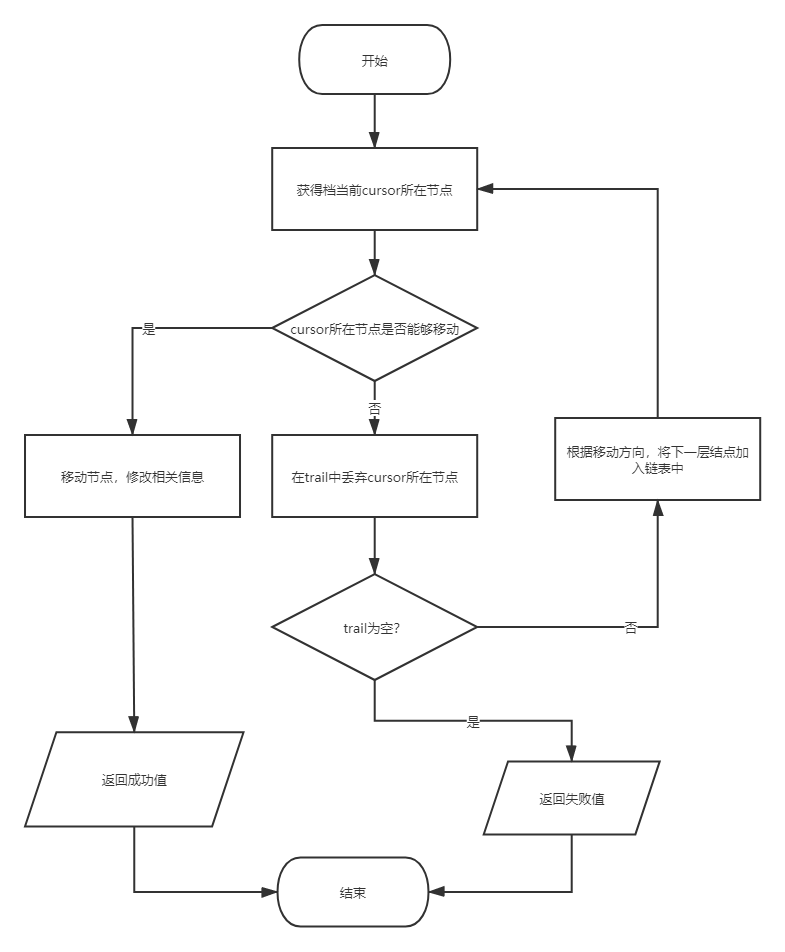


3.cursor移动的实现

虽然在一个表中的每一项记录在逻辑上是连续的，但在物理储存结构上两条记录可能不相连，所以在对游标进行移动操作时会涉及到切换结点的问题，在游标中使用链表记录从根节点到对应B树叶节点的路径。根据链表中的记录，能够快速找到其父节点，从而能够快速完成叶节点的切换。

游标进行next或prev的流程如图2-3-1所示

图2-3-1



对cursor进行seek操作的流程图如图2-3-2所示

图2-3-2

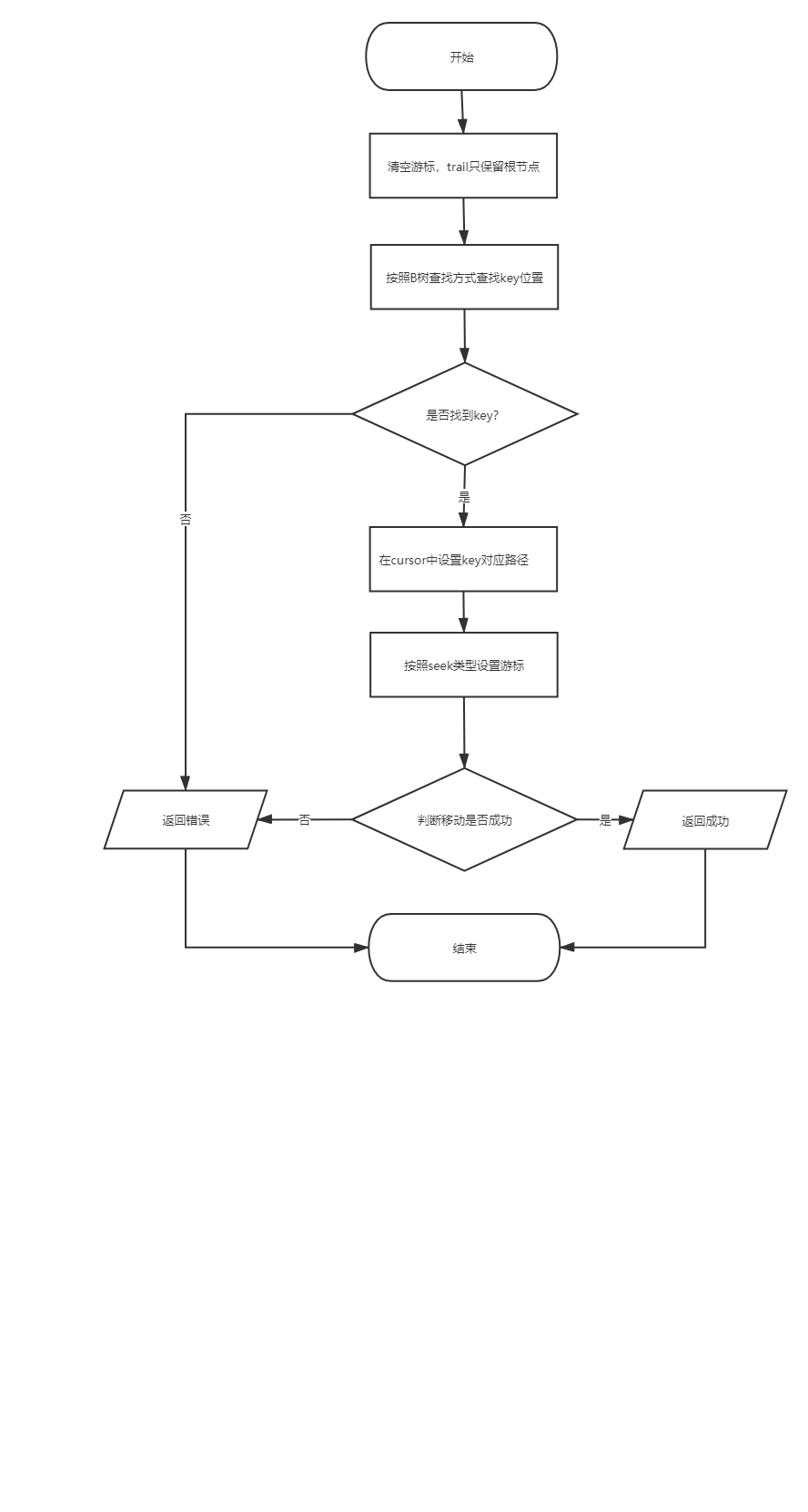


图2-3-2

4.dbm向上提供的接口函数及参数

所有的dbm操作函数都只有两个参数stmt 和 op但不同的指令格式不同

**//通过op->opcode 返回执行对应命令的函数**

int chidb\_dbm\_op\_handle (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*寄存器赋值操作类**

**\*Integer String Null Copy SCopy**

**\*与数据库无关，仅仅是根据命令参数向对应的寄存器赋值**

**\*Integer op->p1 value op->p2 r**

**\*将p1 中的值放入 指示的寄存器中**

**\*String op->p1 length op->p2 r op->p4 string**

**\*将长度为Length的字符串存入寄存器r**

**\*/**

int chidb\_dbm\_op\_Integer (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_String (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_Null (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*寄存器操作函数**

**\*op->p1 r1 op->p2 r2**

**\*将寄存器r1 中的内容拷贝到r2 中 Scopy 是浅拷贝\*/**

int chidb\_dbm\_op\_Copy (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_SCopy (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*指令跳转相关函数**

**\*Eq Ne Lt Le Gt Ge**

**\*与数据库无关，仅仅是对于指令是否按照给定的值跳转**

**\*op->p1 r1 op->p3 r2 op->p2 address\_j**

**\*比较r1 r2 两个寄存器中的值，如果比较结果与对应命令相同，则将指令跳转到address\_j 的位置\*/**

int chidb\_dbm\_op\_Eq (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_Ne (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_Lt (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_Le (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_Gt (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_Ge (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*根据给定的page打开\关闭游标**

**\*op->p1 cursor 1 op->p2 column\_number**

**\*打开指定游标，并记录对应的行数\*/**

int chidb\_dbm\_op\_OpenRead (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_OpenWrite (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_Close (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*重置游标,使其指向第一个单元**

**\*Rewind**

**\*如果游标对应的B树为空，则执行跳转**

**\*op->p1 cursor c op->p2 address\_j\*/**

int chidb\_dbm\_op\_Rewind (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*移动游标**

**\*Next Prev Seek SeekGt SeekLt SeekGe SeekLe**

**\*op->p1 cursor c op->p2 address\_j op->p3 key**

**\*按照给定的key对游标进行一次移动操作**

**\*移动失败则进行指令跳转\*/**

int chidb\_dbm\_op\_Next (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_Prev (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_Seek (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_SeekGt (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_SeekGe (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_SeekLt (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_SeekLe (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*以索引为判断条件的跳转函数**

**\*op->p1 cursor c op->p2 address\_j op->p3 Idkey**

**\*判断成功后执行跳转\*/**

int chidb\_dbm\_op\_IdxGt (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_IdxGe (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_IdxLt (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_IdxLe (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_IdxPKey (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*返回表中的结果**

**\*Column op->p1 cursor c op->p2 column n op->p3 register r**

**\*将游标c所指的某个单元格中第n列数据存入寄存器r中 (n从0开始)\*/**

int chidb\_dbm\_op\_Column (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*返回cursor所指项的关键字值**

**\*op->p1 cursor c op->p3 register r**

**\*如果cursor所指向的项是索引项，则返索引对应的值Pkey\*/**

int chidb\_dbm\_op\_Key (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\***

**\*记录要返回的其实行号和行数**

**\*op->p1 StartRow op->p2 nRR\*/**

int chidb\_dbm\_op\_ResultRow (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\***

**\*向表中添加记录**

**\*op->p1 startregister r op->p2 number n op->p3 register r2**

**\*向表中添加从r1 到 r1+n-1 中所记录的数据并且将这些记录存入r2中\*/**

int chidb\_dbm\_op\_MakeRecord (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*向表中添加一个新的记录**

**\*op->p1 cursor op->p2 register r1 op->p3 r2**

**\*获得r1所存的数据，并将其存入r2所指示的关键字的表中，并且cursor指向新的单元\*/**

int chidb\_dbm\_op\_Insert (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_IdxInsert (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**/\*创建表**

**\*op->p1 register r1**

**\*在B树中添加一个新的节点，并且将这个节点对应的页号存入r1 中\*/**

int chidb\_dbm\_op\_CreateTable (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

int chidb\_dbm\_op\_CreateIndex (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)

**//中断DBM，退出程序**

int chidb\_dbm\_op\_Halt (chidb\_stmt \*stmt, chidb\_dbm\_op\_t \*op)