# 哈爾濱工業大學

数 据 库 实 验 二 报 告

# 实验二：缓冲池管理器（Buffer Manager）


姓　　名 郭　　炼

学　　号 1 1 8 0 1 0 0 2 1 7

老　　师 邹　　兆　　年

专　　业 数据科学与大数据技术


日　　期 2020 年 05 月 25 日

# 目录

# 第1章 实验目的

在本次实验中，我需要基于 BadgerDB 已经实现的存储管理器，采用时钟算法作为缓冲池替换策略管理缓冲页，实现 BadgerDB 缓冲池管理器。

# 第2章 缓冲池替换策略

## 2.1 时钟算法

在本次实验中，我们使用时钟算法作为缓冲池替换策略管理缓冲页。

我们将缓冲池的所有页看成一个环，维护一个时钟指针来指示当前的页。每次申请新页时，我们首先移动时钟指针，之后查看当前页是否空闲：如果为空，就直接返回当前空页；如果不为空，就查看是否空闲。如果空闲就写回缓冲（如果必要）再返回。如果所有页都被占用着，就抛出一个异常。该算法对应的流程图如图2.1所示。

# 第3章 代码实现

我们只需要编辑 buffer.cpp 文件，实现 `Buffer` 类对应的方法就可以完成本次实验。

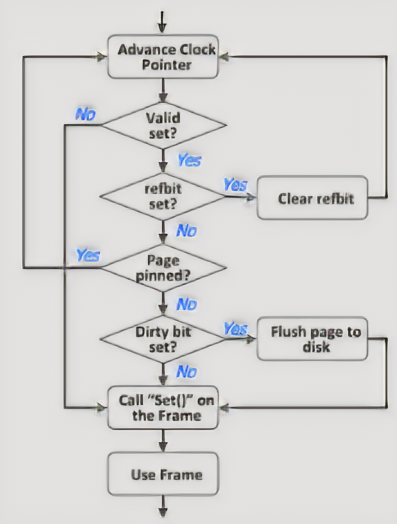


图 2.1: 时钟算法流程图

## 3.1 析构算法

在析构函数中，我们只需要做两件事：写回脏页和释放内存。程序会将所有合法的有效脏页写回到磁盘，之后将占用的所有堆内存释放（如 hashTable，bufPool 等）到堆上。

## 3.2 advanceClock 方法

在 `advanceClock` 方法我们只需要让时钟指针前进一步就可以了。

需要注意的是，由于缓冲页逻辑上是环形的，所以如果指针到达终点后面（`clockHand == numBufs`），我们需要将指针移回至起点。

## 3.3 allocBuf 方法

`allocBuf` 方法需要返回一个空闲的页用于其他程序来使用。在该方法中，我们需要实现时钟算法（参见第二章），同时更新哈希表。

如果程序找到一个可用的页，那么就会设置参数中的 frame 来返回；否则就抛出 `BufferExceededException` 异常。

## 3.4 readPage 方法

`readPage` 方法则是用于读取特定页。其参数为`File* file, const PageId PageNo, Page*& page`，其中 `file, PageNo` 用于定义查询对应的

页。page 则用于返回对应页指针。

程序会首先在哈希表中查询对应的页。如果找到，则表明该页已经在缓冲池中，那么只需要更新描述表中 refbit, pinCnt 两个变量即可，之后返回该页；如果没有找到，那么表明我们需要使用 allocBuf 方法来为该页申请一个新的缓冲池，之后更新描述表，在哈希表中插入对应项。

## 3.5 unPinPage 方法

unPinPage 方法用于释放一个页的占用。具体来说就是将在哈希表中差点对应的缓冲，将其对应描述表的 pinCnt 减一。如果此时 pinCnt 为零，则抛出 PageNotPinnedException 异常。

如果该页不在哈希表中，则表明该页在缓冲池中没有对应的缓冲块，那么就不需要更新任何数据。

## 3.6 allocPage 方法

allocPage 方法用于将给定文件的页读入缓冲池中。我们可以使用 file 的 allocatePage 方法来在对应的文件中申请一个新的空页，之后利用 allocBuf 方法来申请新的缓冲块，将申请的新页与新的缓冲块建立其联系，并在哈希表中插入对应的项，同时更新描述表。

在完成这一切之后，更新 PageNo, page 两个变量来传出新页和缓冲块的数据。

## 3.7 disposePage 方法

disposePage 方法用于将指定页从文件中删除。如果该页在缓冲池中有对应的块，那么同时也需要同步清空对应的描述表，删除哈希表中对应项，释放该块。

## 3.8 flushFile 方法

flushFile 方法将给定文件的全部缓冲页全部写回，并在哈希表中删除对应的项，清空描述表中对应信息。需要注意的是：

- 若删除的块无效，则需要抛出 BadBufferException 异常；
- 若该块正在被其他程序占用，则需要抛出 PagePinnedException 异常。

# 第4章 实验总结

在本次实验中，我对于缓冲池管理器的工作机制有了更为清晰和具体的理解，学习到数据库中缓冲池的工程细节和原理，对于缓冲池的页管理有更深入的理解，同时提升了 C++ 的工程代码能力。

在对自己的代码调试的过程中，我也感受到了数据库系统的复杂性和综合性。数据库缓冲池涉及了许多对于内存和文件的操作，而一旦操作失误，如果没有足够的检查，报错不够及时，那么就会在出现错误时无从查起，需要溯源多级。