Final Project: B+ tree

孫聖授 電信所 R04942031

吳冠融 電信所 R04942070

游肇輝 電信所 R05942046

* 建構**Relation**的Class, 被建構時,同時生成B-plustree與Slotted\_Page兩個class

Slotted\_Page

Relation

BplusTree

Relation<T>::Relation(const string& relationName, const string& keyType, int recordLength) : \_bpt(), \_sp(recordLength)

1. InsertRecord



Slotted\_Page

2) deleteRecord



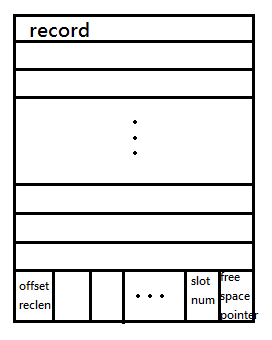
BplusTree

在B-plustree中用key來找到此筆record並刪除, 在slotted page中用rid來找到此筆record並刪除

1. Implementation choices and Slotted Page management :

* SLOTTED PAGE
* Slotted Page Formate:

作業中以C++物件的形式來formate slotted page，物件內容包含pageSize = 512 bytes, free space pointer, slot number, offset, reclen以及儲存record的slot，而record以及對應到的RID是以vector的形式來實作。

* Slot\_Number計算方式：由右圖

令Slot最大個數為n

已知Free Space Pointer Size = 2 bytes

Slot Number Size = 2 bytes

Offset Size = 2 bytes

Reclen Size = 2 bytes

Reclen在建立Relation時決定

因此可列出以下式子：

(Page Size - Free Space Pointer Size – Slot Number Size) / (Reclen Size + Offset Size + Reclen) ≥ Maximum Slot Num

* (512 -2 -2) / (2 +2 + reclen) ≥ n

又因int除法為無條件捨去

n = 508 / ( 4 + reclen)

* Insert and Delete Function:

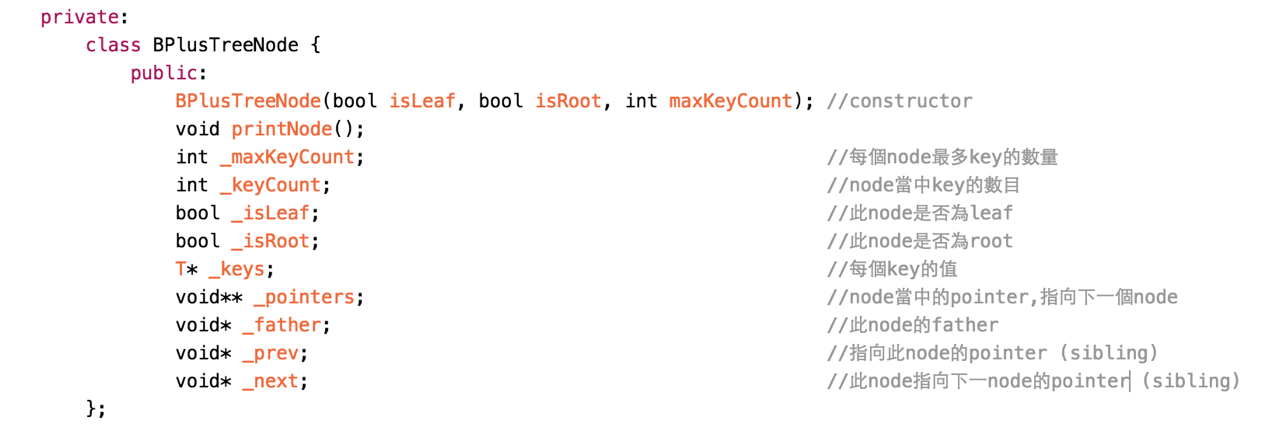
實作Insert和Delete時，以填滿目前最新開的page後再回填前面被Delete掉的部分為原則來實作。

Example1:資料已經Insert到第5頁了，然後接下來對1, 2, 3, 4頁中的資料做Delete的動作，那之後再繼續Insert時，會先將第5頁填滿後，再回填前面1, 2, 3, 4頁的空缺處。

Example2: 接續Example1，如果今天已經回填到3頁的空缺處了，此時若再Delete掉第1頁當中的資料，則先繼續填滿3, 4, 5頁，之後再繼續檢查前面是否有需要回填的部分。

1. B+ tree Implementation

* 定義BPlustree的class, 其中我們以nested class的結構來實現, 在BPlustree中的class中包含著另一class為BPlusTreeNode, 也就是說BPlustree的private member為BPlusTreeNode class, 而主要以BPlusTreeNode來操作tree當中的operations.
* BPlusTreeNode當中的資料有以下:



每個node (page)的大小為512 Bytes, 由key的大小( 可能為int: 4 Bytes, string: 10 Bytes), 可以去計算出\_maxKeyCount 的大小,也就是每個node當中最多可以有幾個key。

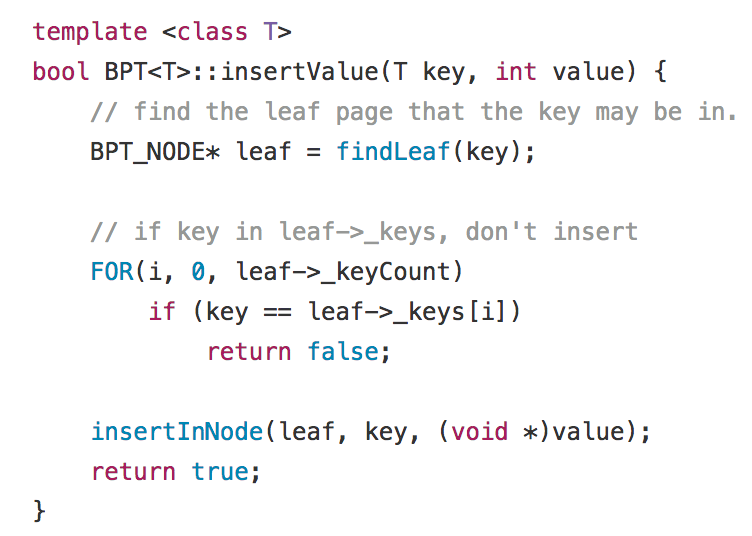
Key 的大小若為x, \_maxKeyCount為n,

n\*x+ (n+1)\*4+4(\_prev)+4(\_next) <= 512

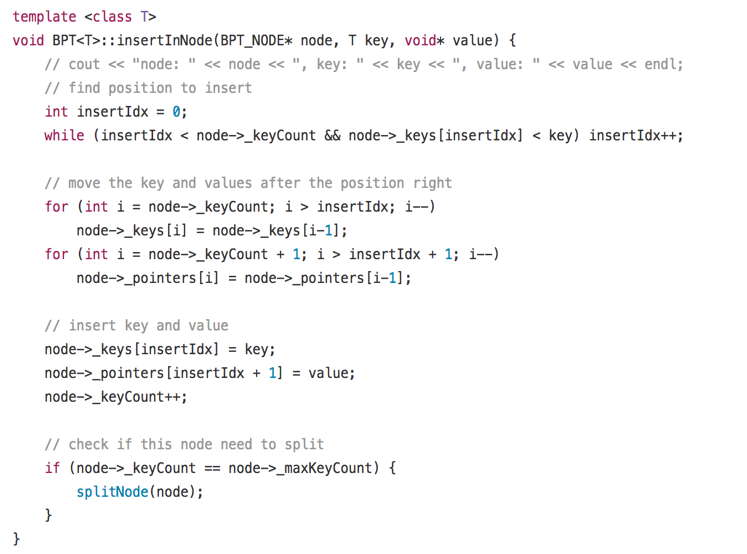
**(n+1): pointer 會比key的數量再多一個**

n=(500/(x+4)), x=4 o 10 => n= 62 or 35

* **BPlusTreeNode 中的insert**

 必須先找到要插入的key所在的leaf node,並排除重複的key (此project下假設並無重複的key)

並把此leaf node,key value傳入insertInNode



在insertInNode當中,

必須先找到此key在leaf node 正確的位置,並且要將此位置右邊的key都右移

移完後就可以開始插入key和value,必須注意(底線處)的是在leaf node當中的pointer與key數量是相等的

而當node中key數量滿的時候必須進行splitNode.

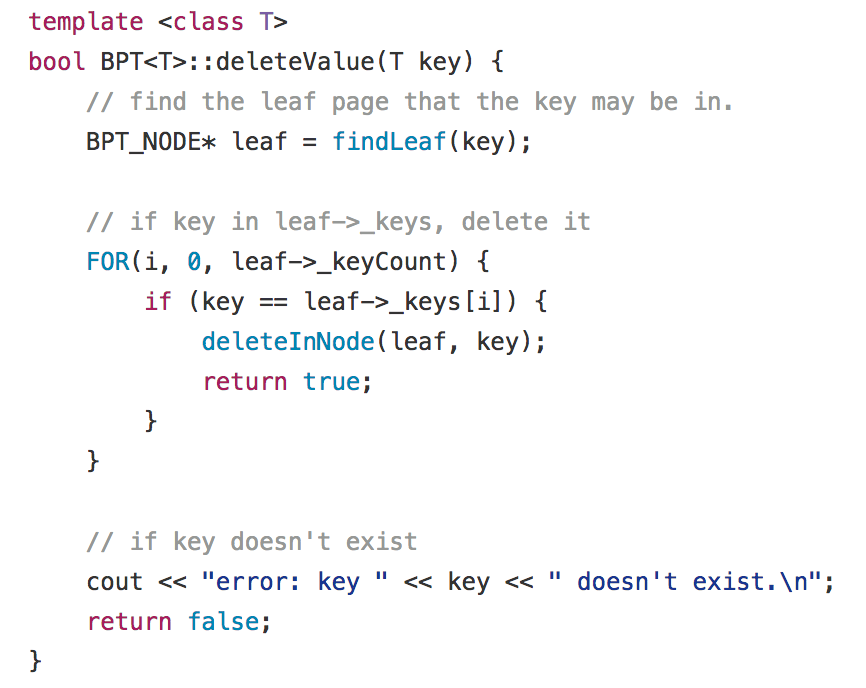
* **BPlusTreeNode 中的splitNode**

必須是創造一個新的node去放右邊的key,移動在最右邊的 (\_maxKeyCount-\_maxKeyCount/2)個key 到新的node上, 並調整新的node上key的pointer所指向的位置

如果此node不是root的話,必須去更新node的father pointer

若node 為root, 必須再去再創新的root

* **BPlusTreeNode 中的delete**



1. Test Design:

* Insert, Delete實作：

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 順序 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| insert |  |  |  |  |  |  |  |  |  |  |
| delete |  |  |  |  |  |  |  |  |  |  |
| 順序 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| insert |  |  |  |  |  |  |  |  |  |  |
| delete |  |  |  |  |  |  |  |  |  |  |
| 順序 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| insert |  |  |  |  |  |  |  |  |  |  |
| delete |  |  |  |  |  |  |  |  |  |  |

Record值：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Page \ slot | 0 | 1 | 2 | 3 | 4 | 5 |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |

* 無意義字處理：
* 指令格式錯誤處理：

在B+ tree當中

insert ( <T> key, int RID) delete (key) 需要檢查的幾種狀況

1. Duplicate key

insert (1, 12), insert (1,34)

1. 刪除不存在的key

insert (1,12), insert (2, 34), delete (3)

1. Split the node
2. Case Study: Duplicate Key

* FUNCTION DISCUSSION:

實作B+ Tree Index系統時，如果以這次作業的Function來看的話，在做一些基本的Scan, Display Data Page of a Relation, Range Query Using Index以及File, Index Statistics的功能時，是否考量Duplicate Keys的狀況，對於上述功能似乎沒有太大的影響，不過Range Query Using Index若加入Duplicate Keys的話，同一個Key值可能同時對應到多個RID，在這種狀況底下如果要進行one-to-one Search的話，可能會失去其功能性。

* FUNCTION INFLUENCE:

考慮Duplicate Keys的狀況之後，主要影響較大的功能為：Single Value Index Search以及Delete Record。

* Single Value Index Search

起因於同一個Key值會對應到許多個Record，所以若單純對Key值作Search的話會產生Error，系統不曉得該回傳哪一筆結果。

解決辦法：考慮到Duplicates Key的狀況時，當針對同一個Key值有多筆insert且要對key值作search的動作時，則在實作上需要增加搜尋條件，如：插入的時間序、RID或者Record的關鍵字。

* Delete Record

同上，起因於同一個Key值會對應到許多個Record，因此對Key值作delete的動作時，系統無法判斷該刪除哪一筆資料。

解決辦法：考慮到Duplicates Key的狀況時，當針對同一個Key值有多筆Insert且要對Key值作Delete的動作時，則在實作上需要增加其他刪除條件，如：插入的時間序、RID或者Record的關鍵字。