Extending the Functionality of a B+ Tree

Database Applications - Recitation 11
Zeinab Khalifa
April 2nd, 2020
What are we trying to solve?
High-Level Overview
High-Level Overview
High-Level Overview
Understanding the Tree & KeyRecord Structures

```c
struct PageHdr {
    char page_type;    /* 'N' for NonLeaf, 'L' for Leaf */
    Long page_num;
    Long next_leaf_page_num;    /* FOR LEAF PAGES ONLY */
    int num_bytes;
    int num_keys;
    KeyRecord* key_list_ptr;
    Long final_right_page_num;    /* FOR NONLEAF PAGES ONLY */
};
```
Understanding the Tree & KeyRecord Structures

```c
struct PageHdr {
    char page_type;  /* 'N' for NonLeaf, 'L' for Leaf */
    Long page_num;
    Long next_leaf_page_num; /* FOR LEAF PAGES ONLY */
    int num_bytes;
    int num_keys;
    KeyRecord* key_list_ptr;
    Long final_right_page_num; /* FOR NONLEAF PAGES ONLY */
};

struct KeyRecord {
    Long page_num; /* FOR NONLEAF PAGES ONLY */
    int key_len;
    char* stored_key;
    Long posting; /* FOR LEAF PAGES ONLY */
    KeyRecord* next;
};
```
Bree & KeyRecord Example

- PageHdr
  - 'N' or 'L'
  - Page Number
  - Next Leaf Page No
  - NumBytes
  - NumKeys
  - KeyListPtr
  - Ptr to the rightmost child
Bree & KeyRecord Example
Bree & KeyRecord Example

PageHdr
- 'N' or 'L'
- Page Number
- Next Leaf Page No
- NumBytes
- NumKeys
- KeyListPtr
- Ptr to the rightmost child

KeyRecord
- Page Number
- KeyLen
- Key Ptr
- Posting Ptr
- Next

For leaf pages only
- For non-leaf pages only

Page containing keys<"aaa"

"aaa"

"aab"

...
Existing Functionality

- C
- i <doc>
- p <num>
- s <key>
- S <key>
- T
- q
Project structure & Demo
What do you need to implement?

<table>
<thead>
<tr>
<th>Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>f &lt;key1&gt; &lt;keys2&gt;</td>
<td>Print in alphabetical order (forward) the distinct keys that are in the range defined by &lt;key1&gt; and &lt;key2&gt; (including the bounds). If &lt;key1&gt; and &lt;key2&gt; are not in alphabetical order, print &quot;Invalid key order!&quot; If no documents have keys within the given range, print &quot;Keys in the given range not found!&quot;</td>
</tr>
</tbody>
</table>
What do you need to implement?

<table>
<thead>
<tr>
<th>Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{f &lt;key1&gt; &lt;keys2&gt;}</td>
<td>Print in \textit{alphabetical order} (forward) the distinct keys that are in the range defined by \texttt{&lt;key1&gt;} and \texttt{&lt;key2&gt;} (including the bounds). If \texttt{&lt;key1&gt;} and \texttt{&lt;key2&gt;} are not in alphabetical order, print &quot;Invalid key order!&quot; If no documents have keys within the given range, print &quot;Keys in the given range not found!&quot;</td>
</tr>
<tr>
<td>\texttt{b &lt;key1&gt; &lt;keys2&gt;}</td>
<td>Print in reverse \textit{alphabetical order} (backward) the distinct keys that are in the range defined by \texttt{&lt;key1&gt;} and \texttt{&lt;key2&gt;} (including the bounds). If \texttt{&lt;key1&gt;} and \texttt{&lt;key2&gt;} are not in alphabetical order, print &quot;Invalid key order!&quot; If no documents have keys within the given range, print &quot;Keys in the given range not found!&quot;</td>
</tr>
</tbody>
</table>
How to start?

Start Early!

Due on April 18th, 2020
How to start?

Start Early!

Due on April 18th, 2020