TP Bases de Datos II

Parte 4

(ENTREGA 4)
En esta parte del trabajo práctico vamos a comenzar con la implementación del B+ Tree. Este va a ser un cambio bastante grande ya que implica modificar por completo la estructura de datos de las páginas. Dada la magnitud del cambio vamos a encararlo por partes, empezando en este caso por la creación del nodo raíz, y limitando las inserciones al máximo admitido por este nodo. En las próximas entregas seguiremos con los diferentes casos de splits y merges según insertemos o borremos registros.

Introducción

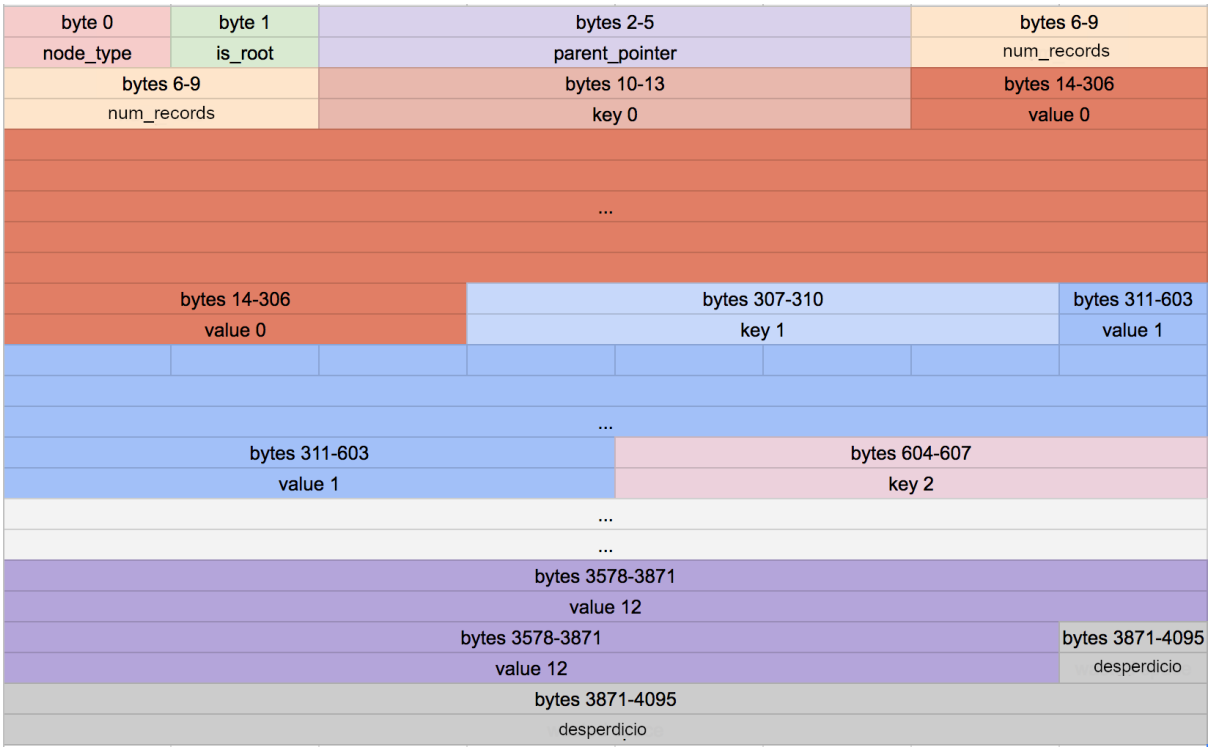
Hasta ahora representamos una tabla como una lista de registros desordenados, organizados en páginas que contenían un máximo de 14 registros cada una. Cada página contenía solamente una lista de registros serializados, uno inmediatamente seguido del otro.
Sin embargo, al cambiar la representación por un B+ Tree cada página representará un nodo del árbol (ya sea interno u hoja). Todas las páginas tendrán un header al comienzo de la misma (cuyo formato varía según si es un nodo interno u hoja), y seguido inmediatamente el cuerpo del nodo.
El cuerpo de un nodo hoja contendrá una lista de pares clave-valor serializados, donde la clave corresponde al id del registro, y el valor es el registro serializado (tal cual lo guardabamos hasta el momento)
El cuerpo de un nodo interno contendrá una lista de pares puntero-clave, donde el puntero será simplemente el número de página hijo al que se está referenciando, y la clave será la clave de mayor valor contenida en el hijo del anterior puntero (o sea el hijo a la izquierda.
En esta entrega sin embargo solo nos preocupamos por los nodos hoja, específicamente el caso del nodo raíz como hoja.
Dado que ahora los nodos tienen metadatos, ya no necesitamos calcular el número de registros a partir del tamaño del archivo, por lo tanto ya no necesitamos guardar páginas parciales en disco. Ahora todos los nodos se guardarán en su totalidad en el archivo, ocupando cada uno 4096 bytes exactamente.

Formato Nodo Hoja

- Header
    - **node_type** (1 byte)
        - 0 para nodos internos
        - 1 para nodos hoja
    - **is_root** (1 byte)
        - 0 para false
        - 1 para true
    - **parent_pointer** (4 bytes)

        - número de página en la que está el nodo padre
    - **num_records** (4 bytes)
- Body
    - *Lista de pares clave-valor (por el espacio ocupado por el header, y el overhead de las claves, ahora solo vamos a poder guardar 13 registros)*
        - **key #** (4 bytes)
            - el valor de la columna id, del registro
        - **value #** (291 bytes)
            - el registro completo serializado

(3835 bytes = 13 * 295)



Requerimientos

Nodo Raíz

- Implementar funciones de acceso para cada uno de los campos del header de un nodo hoja (una página) según la definición de la estructura dada (node_type, is_root, parent_pointer y num_records), que dado un nodo hoja devuelve el valor de en ese campo
- Implementar funciones de acceso (getters y setters) para los valores de un nodo hoja, que dado un nodo (una página) y una posición en la lista de clave-valor (las posiciones empezando de 0), devuelvan o seteen según corresponda la clave alojada en esa posición
- Implementar funciones de acceso (getters y setters) para los claves de un nodo hoja, que dado un nodo (una página) y una posición en la lista de clave-valor (las

posiciones empezando de 0), devuelvan o seteen según corresponda el valor alojado en esa posición
- Implementar un constructor de un nuevo nodo en blanco , de tipo hoja

Base de datos

Teniendo ya implementadas las funcionalidades para crear y modificar un nodo hoja, vamos a hacer el cambio en la base de datos para empezar a guardar los datos de manera estructurada en el B+ tree.

- Ante una base de datos nueva se creará un nuevo nodo hoja con el flag de is_root seteado
- Actualizar el `INSERT` para que inserte en el nodo hoja:
    - Se podrán insertar como máximo el número de registros soportados por un nodo hoja (13 en este caso).
    - Si se intenta insertar un nuevo registro superado el límite se lanzará un error de "Split no implementado"
- Actualizar el `SELECT` para que lea los registros del nodo hoja
- Actualizar el metacomando .table-metadata para que lea los datos del nodo hoja