

MANUAL TECNICO

Repositorio en GitHub: <https://github.com/x30061/josq-db>

VERSION DE JAVA UTILIZADA: java 17.0.4.1 2022-08-18 LTS

VERSION DE MAVEN UTILIZADA: Apache Maven 3.8.2

LIBRERIAS UTILIZADAS:

- jdom2 2.0.6.1: indispensable para leer los documentos xml e interactuar masivamente con la base de datos.
- javafx 19: indispensable para que el usuario final interactue de forma amigable con la base de datos.

REQUISITOS DE USO: se necesita tener instalado la JRE y Graphviz.

Para la estructura de datos de la Base de Datos se utilizo un Arbol B+ basado en el documento: B+ Trees, CSE 332 Summer 2021, Instructor: Kristofer Wong. Dicho documento se encuentra en la carpeta "documentacion" con el nombre: "09-BTree.pdf"

W UNIVERSITY of WASHINGTON

L8: AVL Proof; B+ Trees

CSE332, Summer 2021

W UNIVERSITY of WASHINGTON

L8: AVL Proof; B+ Trees

CSE332, Summer 2021

B+ Trees

CSE 332 Summer 2021

Instructor: Kristofer Wong

Teaching Assistants:

Alena Dickmann	Arya GJ	Finn Johnson
Joon Chong	Kimi Locke	Peyton Rapo
Rahul Misal	Winston Jodjana	

B+ Tree Node Structure

Both the textbook and we refer to "B+ Trees" as "B-Trees", but "B-Trees" actually encompass several variants

- Two node types: **internal** and **leaf**
- Each **internal node** contains up to $M-1$ keys (for up to M children)
 - Does not store values, only keys
 - Function as "signposts"
- Each **leaf node** contains up to L items
 - Stores (key, value) pairs
 - As usual, we'll ignore the "along for the ride" value in our examples

3	7	21	84
---	---	----	----

$x < 3$ $3 \leq x < 7$ $7 \leq x < 21$ $21 \leq x < 84$ $84 \leq x$

3	"cat"
7	"apple"
21	"purple"
84	"ideas"

W UNIVERSITY of WASHINGTON

L8: AVL Proof; B+ Trees

CSE332, Summer 2021

W UNIVERSITY of WASHINGTON

L8: AVL Proof; B+ Trees

CSE332, Summer 2021

B+ Tree Parameters

- Two parameters, one for each type of node:
 - M = # of children in an **internal node**
 - The ranges are defined by $M-1$ **keys**
 - L = # of **items** in a **leaf node**
- Picking M and L based on disk-block size maximizes B+ Tree's efficiency
 - Recommend $M^* \approx \text{diskBlockSize} / \text{keySize}$
 - Recommend $L = \text{diskBlockSize} / (\text{keySize} + \text{valueSize})$
- In practice, $M \gg L$
 - Since typically $\text{sizeof}(\text{key}) \gg \text{sizeof}(\text{keyvaluepair})$

k_1	k_2	...	k_{m-1}	
ptr_1	ptr_2	...	ptr_{m-1}	ptr_m

(sorted by key)

k_1	v_1
k_2	v_2
...	...
k_L	v_L

(sorted by key)

* More precisely, we recommend
 $M = (\text{diskBlockSize} + \text{keySize}) / (\text{keySize} + \text{pointerSize})$

W UNIVERSITY of WASHINGTON

L8: AVL Proof; B+ Trees

CSE332, Summer 2021

W UNIVERSITY of WASHINGTON

L8: AVL Proof; B+ Trees

CSE332, Summer 2021

B+ Tree Structure

- Internal nodes**
 - Have between $\lceil M/2 \rceil$ and M children; i.e., at least half full
 - Reminder: no values, just keys
- Leaf nodes**
 - All leaves at the same depth
 - Have between $\lceil L/2 \rceil$ and L items; i.e., at least half full
 - Reminder: keys and values
- Root node** – A Special Case!
 - If tree has $\leq L$ items, root is a **leaf node**
 - Unusual; only occurs when starting up
 - Else, root is an **internal node** and has between 2 and M children
 - i.e., the "at least half full" condition does not apply

Interactuar con la API:

La clase Interacciones ofrece los metodos para realizar operaciones en la base de datos, a continuacion se listan dichos metodos **en el orden en el que deben ser usados**:

1. addTabla(tabla: String)
2. addColumna(tabla: String,columna: String,tipo: String)
3. setLlavePrimaria(tabla: String,columna: String)
4. setRelacion(tabla: String,columna: String,relacion: String) (opcional)
5. addLlave(tabla: String,tipo: String)
6. setDato(tabla: String,llave: String,columna: String)
7. deleteDato(tabla: String,llave: String,columna: String) (opcional)
8. deleteLlave(tabla: String,llave: String) (opcional)

El codigo fuente del proyecto se distribuye en los paquetes:

josq

- ├ controles: contiene las clases Interacciones (la API), Utilidades, CargarXML
- ├ estructuras: contiene las clases ArbolBp, Nodo, NodoHoja, NodoRama
- ├ modelos: contiene las clases MiBD, Tabla, Columna
- ├ vistas: se encuentra la interfaz amigable
- └ App.java: es el detonador de la aplicacion

En la siguiente pagina se encuentra el diagrama de clases que representa la base de datos. Solo estan las clases indispensables para el funcionamiento de la base de datos.

ARQUITECTURA DE LA BASE DE DATOS

