The design of the JDBM database scheme of the indexer

Overview of Database Components

The design is composed of several JDBM-backed maps (HTrees) that store the various pieces of data required by the indexer. Below is a list of the main components:

1. Mapping Tables: URL ↔ Page ID
2. Mapping Tables: Word ↔ Word ID
3. Forward Index for Page Details and Child Pages
4. Inverted Indexes
5. Word Order Maps for Phrase Search
6. Counter Map for Unique ID Generation

Detailed Database Schema

Mapping Tables: URL ↔ Page ID

urlToPageIdMap

	Field	Type	Description
Key	url	String	Unique webpage URL
Value	pageID	Integer	Corresponding page identifier

This table provides the conversion from a webpage URL to its unique page identifier, which is checked before adding a new page. The indexer checks with this map to determine whether the URL has already been crawled, helping prevent duplicate entries and ensuring efficient lookups. By converting URL strings to more efficient numerical IDs, the design enhances performance and minimizes redundancy.

pageIdToUrlMap

	Field	Type	Description
Key	pageId	Integer	Unique page identifier
Value	url	String	Webpage URL corresponding to the page ID

This table maps a unique numerical page identifier back to its corresponding URL, supporting reverse lookups when details about a page are required. It is utilized during retrieval and display operations to provide human-readable URLs based on numeric identifiers.

Mapping Tables: Word ↔ Word ID

wordToWordId

	Field	Type	Description
Key	word	String	Processed word text
Value	wordId	Integer	Corresponding unique word ID

This table converts processed words to unique numeric identifiers, providing a quick method to check if a word has already been encountered. The indexer uses this mapping during tokenization to either retrieve an existing ID or assign a new one, ensuring that each word is indexed only once. This approach minimizes duplication and facilitates efficient lookup and management of textual data in the inverted indexes.

wordIdToWord

	Field	Type	Description
Key	wordId	Integer	Unique word identifier
Value	word	String	Word text corresponding to the identifier

This table serves as the reverse mapping from a unique word identifier back to the corresponding text, enabling the indexer to convert numerical IDs back to user-readable words during searches and result displays. It is used when generating reports or processing query results, ensuring that the stored textual data can be efficiently retrieved.

Forward Index for Page Details and Child Pages

pageInfo

	Field	Type	Description
Key	pageId	Integer	Unique page identifier
Value	url	String	Page URL
Value	title	String	Page title
Value	lastModifiedDate	Long	Timestamp of the last modification
Value	size	Long	Size of the page in bytes
Value	childPageIds	List<Integer>	List of child page IDs representing links from the current page

Note: Although the table lists the fields of the PageInfo object separately for clarity, they are stored together as a single serialized value in the database, associated with the pageId key.

This forward index records metadata for each indexed page. Metadata including URL, title, modification date, size, and linked child pages are stored as values under a single key (pageId). This structure simplified metadata retrieval and maintains navigational hierarchies efficiently, while effectively managing both static and dynamic page details.

Inverted Indexes

bodyInvertedIndex

	Field	**Type**	**Description**
Key	wordId	Integer	Corresponding unique word ID
Value	postings	HashMap<Integer, Integer> (pageId → frequency)	Posting list for words in the page body

titleInvertedIndex

	Field	**Type**	**Description**
Key	wordId	Integer	Corresponding unique word ID
Value	postings	HashMap<Integer, Integer> (pageId → frequency)	Posting list for words in the page body

These two tables contain the inverted index for page text (the body and the title). Each posting list is associated with a word identifier, mapping to a frequency-count map where each entry pairs a pageId with the frequency of that word. This enables efficient, targeted searches and ranking based on frequency.

Word Order Maps for Phrase Search

pageIdToBodyWords

	Field	**Type**	**Description**
Key	pageId	Integer	Unique page identifier
Value	bodyWords	List<String>	Ordered list of words from the page body

pageIdToTitleWords

	Field	**Type**	**Description**
Key	pageId	Integer	Unique page identifier
Value	titleWords	List<String>	Ordered list of words from the page title

These two tables maintains the sequential arrangement of words for each page, with the pageId serving as the key, and the ordered lists of words (for the body and title) as values. These word order maps support exact phrase searches using a sliding-window approach. This design provides advanced search capability while avoiding the complexity and overhead of building a complete positional index.

Counter Map for Unique ID Generation

counter

	Field	**Type**	**Description**
Key	counterType	String	Type of counter (either "pageId" or "wordId")
Value	counterValue	Integer	Next available unique identifier for the given type

This table maintains counters for generating new unique identifiers for pages and words. The counter type (either "pageId" or "wordId") is associated with a counter. Every time a new page or word is processed, the corresponding counter is incremented by one. Centralized counter management is crucial for preventing identifier collisions, preserving data integrity, and supporting scalable index growth.