**Search engine system design**

**Group 19:**

Student 1: SHI Jianhua 20517310

Student 2:

Student 3:

1. **RocksDB Database Schema**

|  |  |
| --- | --- |
| **RocksDB Name** | **Mapping** |
| Word\_ID\_Bi | word -> wordID and wordID -> word   * word: String * wordID: String (as an integer start from 1) * A keyword is converted into the corresponding wordID by the order of appearance from the indexer.   Explanation/Notes:   * Before a sentence is processed, it will first be tokenized to get a list of words so that phrases can be supported. * The word inserted into this database is not the original keyword in the document but the stemmed version. * The same tokenizer and stemming algorithm will be used for queries to ensure finding correct wordID. * Given a specific wordID, the corresponding word is returned. |
| Page\_ID\_Bi | URL -> pageID and pageID -> URL   * URL: String * pageID: String (as an integer start from 1) * A URL is converted into the corresponding pageID by the order of appearance from the indexer. * Given a specific pageID, the corresponding URL is returned. |
| PageID\_PageInfo | pageID -> page<page title, URL, last modified date, size>   * pageID: String (as an integer start from 1) * page: String   + title: String   + url: String   + modDate: String   + size: int * Given a pageID, the corresponding page is returned.   Explanation/Notes:   * The PageInfo class is used to store the information of a web page and page is a subset of the PageInfo class. Accessing these information directly from pageID is easy by using this database. |
| PageID\_Links | pageID -> pageID…(childLinks), pageID…(parentLinks)   * pageID: String (as an integer start from 1) * Given a specific pageID, the corresponding parent pageID and children pageID is returned. Different parent pageID and children pageID separated by common and parent pageID and children pageID separated by a space |
| InvertedIndex  (separated by TitleInvertedFile and BodyInvertedFile) | wordID -> {pageID, <word position>}   * wordID: String (as an integer start from 1) * pageID: String (as an integer start from 1) * word position: String (as an integer located position in a page)   + The value corresponding to the key is the positions of the wordID in this specific pageID. <word position> is represented using a String containing the positions separated by comma.   + Each set of {pageID, <word position>} separated by a space * Given a specific wordID, the corresponding set of pageIDs and their positions (after stemming) contained in the page is returned. |
| ForwardIndex | pageID -> {wordID, <word position>}   * pageID: String (as an integer start from 1) * wordID: String (as an integer start from 1) * word position: String (as an integer located position in a page)   + The value corresponding to the key is the positions of the specific wordID in this pageID. <word position> is represented using a String containing the positions separated by comma.   + Each set of {wordID, <word position>} separated by a space * Given a specific pageID, the corresponding set of wordIDs and their positions (after stemming) contained in the page is returned.   Explanation/Notes:  Store the words appearing in a page along with their positions so that searching for phrases will be easier. |

For convenient management of database designed above, we use a enum class called DbTypeEnum to manage the database because it is an extendible design for adding more database file in future. We can easily get the instance of database by using the static method in DbTypeEnum.