**COMP 4321 Project: Phase 1 Database Design**

Group 20:

WONG, Wang Lik

CASINO, Benedict Matthew Hilario

NG, Tsun Yam

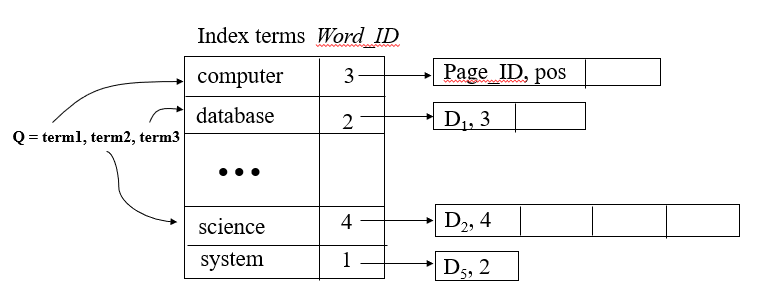
**Overview**

For phase 1 of our Search Engine Project, we incorporated the Java utility codes we learnt from lab sessions to construct our Database System. Namely, the RocksDB database from Lab Session 1, HTMLParser and Crawler from Lab 2, and the Porter Algorithm and StopStem from Lab 3. Apart from this, other libraries and sources were used to add onto the project.

This report will give an in-depth explanation of our project. Such as the Database Design, Inverted Indexes and Functions and Classes in the Phase1.java.

**Inverted Indexes**

Inverted Indexes are used to speed up the lookup process. In our current design, it is implemented in a posting list. Where keys are using the bytes[] converted from string of word and URL.



**Database Schema**

For our design we mainly have 2 tables, word\_db and URL\_db. Schema as follow:

**word\_db** (word\_ID,values)

values (doc\_ID, word\_position)

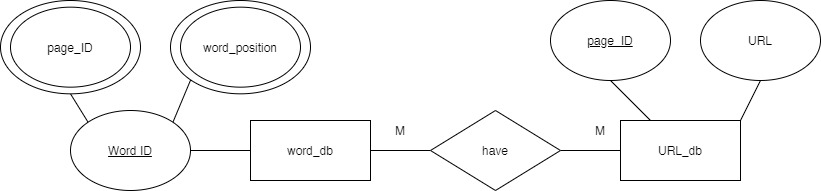
In word\_db , word\_ID is the key and values is a multi-valued attribute storing the information of pairs of doc\_ID and word\_positions. The word\_ID is generated using .getBytes() function. This database can perform word <=> word ID conversion by interchanging the byte[] to String.

**URL\_db** (page\_ID, URL)

In URL\_db , page\_ID is the key and URL storing the actual URL in string. Page\_ID is starts from zero, the new incoming URL will be assign a page\_ID equal to the current length of the list. If the URL exists in the database already, no new record will be added. This database can perform URL <=> page\_ID conversion by interchanging the byte[] to String.

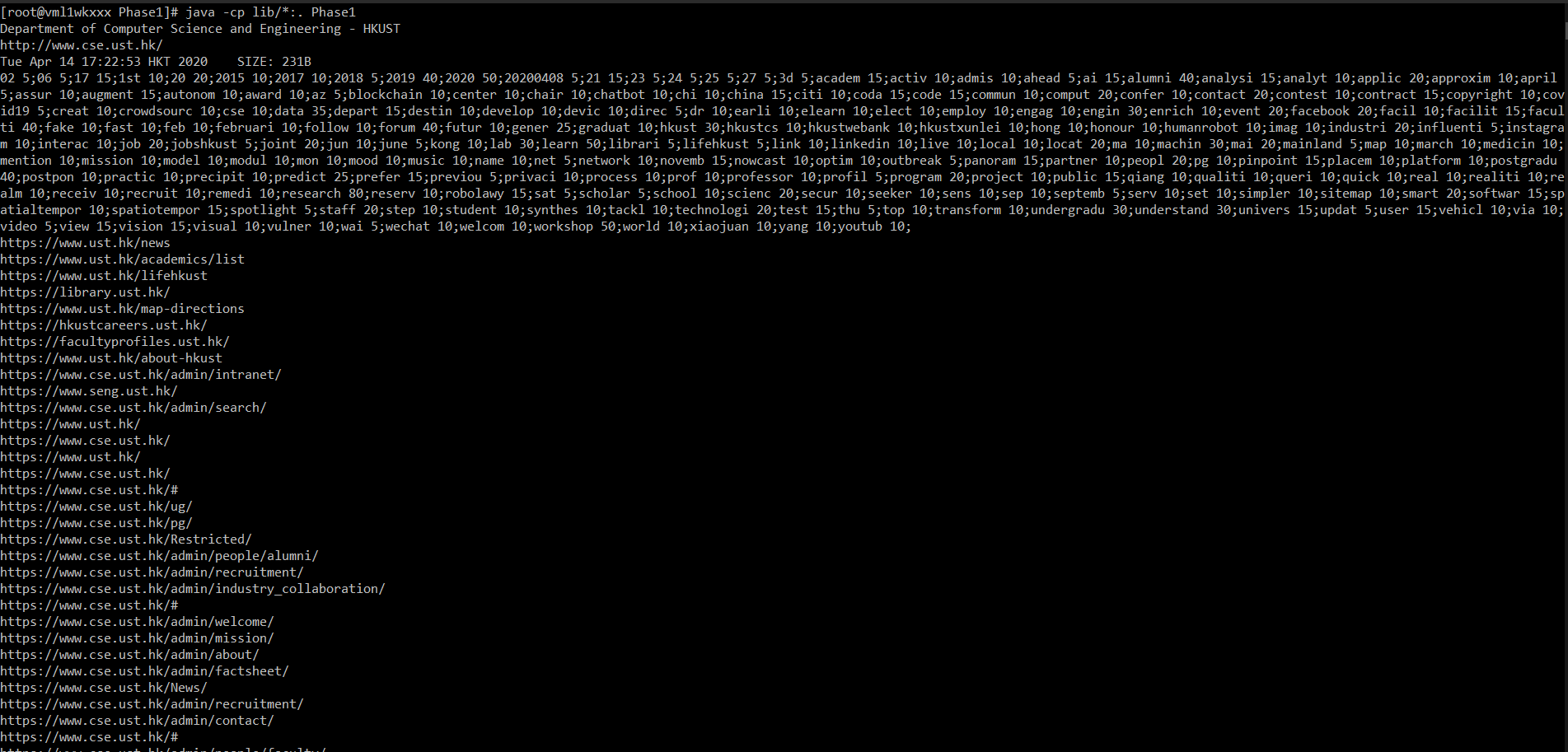
Database relations

Our current database design can be visualized as follow:



Word\_db and URL\_db are in Many to Many relationship, by joining word\_id.page\_ID and URL\_db will able to link 2 database together

Sample Output:



**Functions and Classes**

For our Java code, we implemented 2 classes: StopStem and Phase1.

**StopStem**

StopStem refers to the IRUtilities package for Porter.class and NewString.class. StopStem has the following functions:

public boolean isStopWord(String str)

Stopword Removal with reference to ‘stopwords.txt’

Returns true if the inputted word is a stopword, vice versa.

public String stem(String str)

Stemming with Porter Algorithm and Porter.class

Returns stemmed word as String

**Phase1**

Phase1 is the main body of the program, and consists of several key functions:

public Vector<String> extractWords() throws ParserException

Extracts Words from URL

Returns an vector array of Extracted Words

public Vector<String> extractLinks() throws ParserException

Extracts Links from URL

Returns an vector array of Extracted Links

public void addEntry(String word, int x, int y) throws RocksDBException

Adds Word Entry to words DB by converting entry to byte[] format

Added in the format of docX + Y, where X is the Doc Number and Y is Position

public int addUrl(String url , RocksDB url\_db) throws RocksDBException

Adds URL Entry to URL DB by converting entry to byte[] format

Returns index of URL in the DB

public static int countOccurences(String str, String word)

Counts the number of instances of Substring word within String str.

Returns count

public void printAll(PrintWriter printWriter,int i) throws Exception

Prints out Word Entries and Occurences to Terminal and spider\_result.txt

public static void writing (PrintWriter printWriter,Phase1 master,Phase1 child,int doc\_id,int size,int size\_format) throws Exception

Prints out Title, URL, Date, Size, Word Entries and Occurences, and Child Links to Terminal and spider\_result.txt.

If the Date and Size are not mentioned in the URL Header, they will be set to the current date and character count respectively