COMP4321 Phase 1 – Database Design Report

# Low-level data storage and serialization

Our custom database uses JSON as the file format for storing the data on the disk. For the data serialization, it is achieved by the json library, which is included in Python as default.

# Database storage

The read/write of the custom database to disk is achieved by src/egod\_search/crawl/database.py Database class. The class implements read and write methods. Reading is achieved by database\_obj = await database.read(). Writing is achieved by await database.write(database\_obj).

# Database scheme

Fundamentally, the custom database is a dictionary, where key lookups occur in constant time in the average case. This attribute was bear in mind while we design the database scheme, so as to avoid any reverse lookup that would take linear time instead.

Key url\_ids is a dictionary whose keys are the URLs and values are the corresponding page-ID

🡪This enables high-efficiency lookup from URL to page-ID

Key word\_ids is a dictionary whose keys are the keywords and values are the corresponding word-ID

🡪This enables high-efficiency lookup from word to word-ID

Key pages is a dictionary whose keys are page-ID and values are the corresponding dictionary storing the information for that page with a scheme as follows:

* Key title: the value is the title of the page
* Key text: the value contains the body text of the page
* Key link: the value is a list of URLs in the page
* Key mod\_time: the value is the last-modified time of the page

🡪This enables high-efficiency lookup for fetching general information for generating spider\_result.txt

Key inverted\_index is a dictionary whose keys are word-IDs and values are the corresponding dictionary storing at what page the word has appeared in, with a scheme as follows:

* Each key is the page-ID, and the value is the corresponding list which stores at what word position the keyword in question has appeared in the page.

🡪This enables high-efficiency lookup for the search engine part in the final submission, after phase 1

Key forward\_index is a dictionary whose keys are page-IDs and values are the corresponding dictionary storing at what word the page in question contained, and at what frequency. The scheme is as follows:

* Each key is the word-ID, and the value is the number of times the word has appeared in the page in question.

🡪This enables high-efficiency lookup for fetching keyword frequency for generating spider\_result.txt