CMPUT 291 - Mini Project 2 Design Document

1 Overview and User Guide

The following software package is comprised of 3 phases. Phase 1 takes email data from an XML file and outputs the data into four files: terms.txt, emails.txt, dates.txt, and recs.txt. Phase 2 sorts these files and builds four indexes. Phase 3 is responsible for data retrieval, and queries the data.

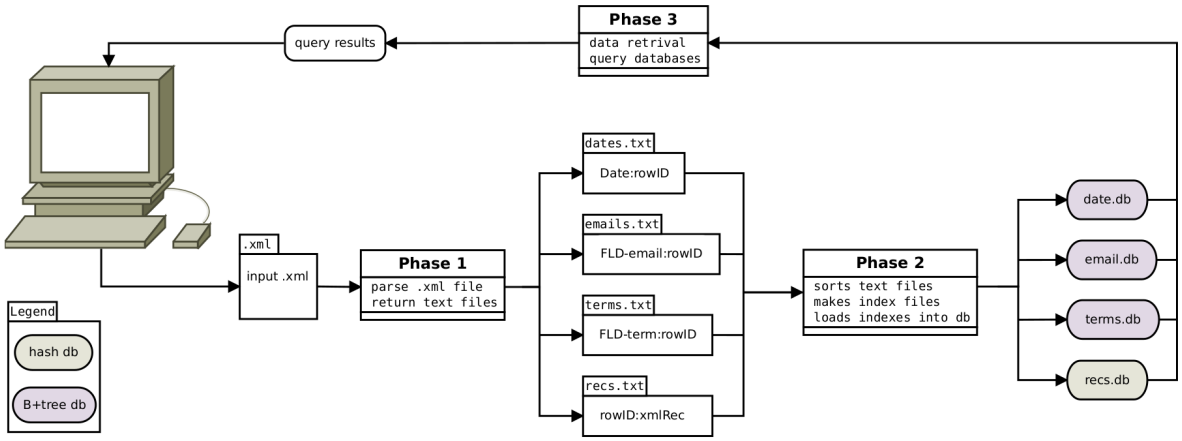

Figure 1: Flow diagram of files, data, and different phases of the software. For more implementation details see *Software Design*.

1.1 Phase 1: Preparing Data Files

Preparation of the data files is completed by the python program `phase1.py`. The software reads in an `.xml` file specified by the user line by line, parses the line, and writes the data into four output files: `terms.txt`, `emails.txt`, `dates.txt`, and `recs.txt`. The phase 1 software can be run by entering the following command in the Linux terminal:

```
$ python3 phase1.py
```

The program will prompt the user to enter an `.xml` file:

```
Enter .xml file: datafile.xml
```

If the provided file is not found, or an incorrect file extension is given, an error message will display, and the user can re-enter a filename. If no extension is specified, the program will assume it is `.xml`. If the output files already exist in the directory, the program will overwrite the data currently on the files.

1.2 Phase 2: Building Indexes

Phase 2 takes the output files from phase 1 and sorts them, and produces four index files from the sorted data: `te.idx`, `em.idx`, `da.idx`, `re.idx`. If the index files already exist from a previous run, the data on them will be overwritten.

The index files are then loaded into four databases: `terms.db`, `email.db`, `date.db`, `recs.db`. Once again, if these databases are already found in the directory, they will be overwritten by the new data.

No user input is required for phase 2. To build the indexes, simply execute the following command in the terminal:

```
$ python3 phase2.py
```

1.3 Phase 3: Data Retrieval

Phase 3 provides the user a simple interface to query the data prepared in phases 1 and 2. The query program uses *Berkeley DB* to process the queries entered by the user. The user can query dates, emails, or terms. Depending on the type of query, it will have different formats. In general, queries are of the form:

```
> prefix:query
```

A date query will return all emails with dates that satisfy the query.

```
> date(:|>|<|>=|<=)YYYY/MM/DD
```

An email query returns all emails sent to/from/cc/bcc the email address specified:

```
> (to|from|cc|bcc):email@address.com
```

A term query returns all records that have the term in their subject or body field (field specified by user)

```
> (subj|body):term
```

Terms can also be queried without specifying the field. For example,

```
> confidential%
```

will return all records that have a term with prefix *confidential* in their subject or body fields (confidential, confidentially, confidentiality, etc.).

The program will ignore all white spaces in the entered queries, so

```
> subj :   gas          body:earning
```

will successfully search the data for records that have gas in their subject and earning in their body fields. All queries are case-insensitive as well.

2 Software Design

3 Testing Strategy

4 Group Work