Create a reasonable struct that can contain any single datum from a record in the data file, and associated ctor and dtor , initializers, comparison functions, etc.

Create a record class that contains a vector for the datum class, as well as functions to sort, select, add, delete, categorize, etc. , datums.

Create a record collection class to accumulate records, search for records, categorize records, etc.

Create an index class functor to create an index for each record based upon a datum or combination of datums to select sets of records from a collection.

Write some functions to print out a single record according to some selected format. Each format can have its own print function, or you could make formatting a separate class for a single print function to use.

Write a function to print out a selected batch of records from a collection, with a selected format.

The index file:

hash each record (MD5, sha1, other).

For each field in the record, make a section for that field in the index file.

Copy the field to the index file if it is not already there. add the hash after the datum. Do this for all fields, all records.

Go through the data file and do this for each record.

You will end up with a dense data index wherein each unique datum is on its own line with the hash of the record in which it occurs, if the datum occurs in multiple records the other hashes will be there after that.

So then when you search for “last name Johnson” you can jump to the lastname section of the index , find Johnson, read all the hashes, then get them from the data file.