Design Questions

1. According to the 1976 census, Canada had 22,000 named towns and a population of 22,000,000 (approximately). The world has about 100 times the population of Canada. The following data is to be recorded for each town in the world: `year`, `town id`, `country name`, `status` (a four-character code), `population`, `latitude` and `longitude`. The year is to record the date of the census from which the data is taken, and permits a history to be kept for each town.

   The data is to be used for statistical purposes and for fast retrieval of information on any specific town. Statistics could group towns by year, by country or by geographical location. The data is stored on a disk with 2 microsecond (μs.) per byte transfer time and 20 millisecond (ms.) seek time, and retrievals on individual towns should take no more than 1/5 second.

   Using the categories on which the course is based, and making appropriate calculations approximately, discuss the design of the file or files to satisfy the above criteria.

2. A hospital intensive care ward has 12 beds, each with monitors for pulse rate, breathing rate and E.E.G. data. For each bed every two seconds data is collected in the form of one real value for each of the three monitors. The time and bed number are also recorded.

   The resulting file is accumulated for several days and used to generate further files: every minute, an average value for each of the three quantities for each bed is calculated and stored in a minute-average file; every hour, a similar average is taken for the hour-average file. The minute-average file is used primarily to look for trends during the day for each bed. The hour-average file is used primarily to inspect all the patients for a given hour.

   The data is stored on a disk with 2 microsecond (μs.) per byte transfer time and 20 millisecond (ms.) seek time. There are 86400 seconds in a day. How many bytes do you estimate will be added per day to each file? Show how you estimate record sizes.

   Using the categories on which the course is based, and making appropriate calculations approximately, discuss the design of each file to serve its purpose best. Estimate times required to perform the main operations on your files.

3. A personnel file in an organization of 2000 employees is used for preparing twice-monthly paycheques, is updated infrequently by the deletion or addition of employee records, and is very occasionally used to compute correlations between employee salaries and their savings or charitable donations.

   Each employee has a record with the following fields. *Name* and *Address* occupy about 80 bytes together. The remaining fields are integers or real numbers: *Salary, Tax Rate, UIC, Pension, Health Plan, Dental Plan, Insurance, Union, Donations*, and *Savings*.

   The file is stored on a device with a seek time of 20 milliseconds (ms.) and a transfer rate of 2 microseconds (μs.) per byte. Updates of single records must be done in a half second or less.

   The correlation between two fields x and y may be calculated from

$$\frac{\sum_i (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_i (x_i - \overline{x})^2 \sum_i (y_i - \overline{y})^2}}$$

   where $\overline{x}$ is the average value of x and $\overline{y}$ is the average value of y.

Using the categories on which the course is based, and making appropriate calculations approximately, discuss the design of the file or files to best serve these purposes. Estimate the cost of computing the correlations of *Salary* and *Donations* (e.g., to Centraide, by payroll deduction) and of *Salary* and *Savings* (e.g., buying a Savings Bond), using the file(s) you have designed.

4. A city maintains data on properties, their evaluations, and their selling histories in two files. The first file contains *street, house number, owner name, construction code* (1 digit), *land area, land evaluation, building evaluation* (the last three are integers). The second file contains *street, house number, sale code* (1 letter), *sale registration number* (7 digits), *date of sale, sale price* (integer).

   Estimate record sizes, and, for 500,000 properties, each of which is sold *on the average* once every ten years, estimate the file sizes for a twenty year period.

   Using the categories on which the course is based, and making appropriate calculations approximately, discuss the design of files to support the following transactions.

5. A music researcher is building a multimedia database and wants to store musical excerpts. She also needs to record information on *composer, date, title* and *type*.

   A typical excerpt plays for about ten seconds, and occupies about 200 Kbytes on disk. The excerpt length is highly variable.

   A query might request a display of all symphonies (*type*) by Sibelius (*composer*), prior to asking to hear excerpts from some. Or, it might ask for the names of composers active in the years 1783 to 1791 *date*. Or it might simply request playing the excerpt from The Marriage of Figaro (*title*).

   The researcher should be able to add new data and excerpts at any time.

   Estimate record sizes, and retrieval times for a maximum database size of 10,000 entries, on a disk of 2 μsec. (microseconds) transfer time per byte and 20 msec. (milliseconds) average seek time.

   Using the categories on which the course is based, and making appropriate calculations approximately, discuss the design of file(s) to support the application.

6. Every January, a certain university collects data from each of its 8000 students who plan to graduate that year in order to help them find jobs. The data include *student id, name, age, address, expected degree, major, date available, work preferences* (a text field of some 100 bytes) and *thesis topic* (also a text field of some 100 bytes). These are stored in a file. There are 2 different degrees, each in some 200 subjects.

   A second file is derived from student records, and contains for all students *student id, course/skill* and *grade*. Each student will have had some 30 courses.

   Between January and June, companies submit requests which include *company name, address, degree required, major of interest*, which is stored in another file (which may contain only one or a few companies), and, stored separately, *company name, course/skill* and *importance*. Companies typically specify about 10 *course/skill*s, which fortunately coincide with the *course/skill*s offered by the university.

   The matching process is carried out as soon as requirements are received from a company. It takes place in two stages. First the two files with *degree* and *major* are compared to select students who may be of interest. The *degree* and *major* must match the requirements, and the *date available* must be no later than the *start date*. These candidates are then filtered by comparing the *course/skill*s offered to those required, and calculating a weight for each student based on his/her *grade* in each required *course/skill* and on its *importance*.

Using the categories on which the course is based, and making appropriate calculations approximately, design the four files to support the above matching. Calculate file sizes, and estimate the time to do the match. Assume a disk with 2 μsec. (microseconds) transfer time per byte and 20 msec. (milliseconds) average seek time.

7. A world-wide database for travellers records information on 500,000 restaurants, 100 restaurant chains, and 1000 restaurant districts.

   Some, but certainly not all, of the restaurants belong to chains. Some, but certainly not all, of the restaurants are in restaurant districts.

   The fields stored should include *RestName, Address, TC* (whether the restaurant is in town or in the country), *Cuisine* (e.g., Afghanistani, Greek, Hamburger), *PriceRange* (low/medium/high), *ChainName* and *HeadQuarters* of the chain, *District, City* of the district, and *South, West, North* and *East* streets bounding the district.

   The database should be able to answer queries such as "Find all restaurants in Montréal [*Address*] serving medium-priced Lebanese food", or "Find the district with the most low-priced restaurants", or "Find all the districts which have a McDonald's". Other queries should also be possible: the files should be designed for flexibility.

   Furthermore, the agency maintaining the database should be able to add new restaurants, districts and chains.

   Using the categories on which the course is based, and making appropriate calculations approximately, design files to support the above matching. Calculate file sizes, and estimate the time to do the match. Assume a disk with 2 μsec. (microseconds) transfer time per byte and 20 msec. (milliseconds) average seek time.

8. In an air traffic control system, aircraft file flight plans, which specify a sequence of rectangular cells they will fly through, from a rectilinear grid of cells all the same dimensions, covering the country. They also file departure and arrival times and airports. This makes it possible to predict roughly when each aircraft will be in each cell. Aircraft are identified by flight numbers.

   Thus, flight plans involve fields *FlightNo, DepPort, DepTime, ArrPort, ArrTime*, and a sequence of *Cell*s.

   While an aircraft is in flight, it is monitored, and every ten seconds its *Longitude, Latitude*, and *Bearing* are recorded. These must be checked against the predictions of the flight plan, and a warning issued if the aircraft is in the wrong cell at the wrong time.

   If this happens, the aircraft must be checked against any other aircraft near it for position and bearing. If two aircraft will come within a preset distance from each other on their present bearings, the time when this will happen must be calculated. This process must be complete before the next position and bearing data are recorded.

   (We assume that all aircraft fly at the same altitude.)

   Using the categories on which the course is based, and making appropriate calculations approximately, design file(s) to support the above application. Calculate file sizes, and estimate the times to do the operations. Assume a disk with 2 μsec. (microseconds) transfer time per byte and 20 msec. (milliseconds) average seek time.

9. A retail chain records analytical data for products sold by stores at given times. For each of 30,000 products, *ProductId, Description, Type, Size, Weight*, and *Cost* are recorded. For each of 100 stores, *StoreId, Address, Manager* and *Capacity* are recorded. For each of 730 days, *Date, DayOfWeek*, and *IsPromotionDay?* are stored. Finally, there is data on *Quantity* and *Price* for each product sold by each store for each day.

Typical analyses would seek to find the total quantities and revenues for each product for all days and all stores, or for each store and day for all products, or for each store and product for the second year compared with the same for the first year.

Using the categories on which the course is based, and making appropriate calculations approximately, design file(s) to support the above application. Calculate file sizes, and estimate the times to do the operations. Assume a disk with 20 nsec. (nanoseconds) transfer time per byte and 5 msec. (milliseconds) average seek time.

10. The *hyperglobe* is a proposal for organizing knowledge according to two geographical coordinates (longitude and latitude) and one temporal coordinate (date), indicating the contents or the origin of the topic. The "knowledge" is represented as large chunks of unstructured text or other unformatted data.

    For instance, at the location of Hastings, England, at 1066 AD, there might be a short book (100 Kbytes) on the Norman invasion; at the location of Montreal for 1987 AD there might be a score (10 Kbytes) and an audio record (40 Mbytes) of the premiere of Patriquin's *Earthpiece*; at the location of Moscow for 1991 AD, there might be a short CNN television report on the Soviet coup (1 Gbyte).

    To resolve longitude and latitude to about 0.1 seconds of arc (about 3 metres at the equator) requires 3-byte fields. To resolve dates to quarter-days also requires a 3-byte field.

    There may not be room on secondary storage for the chunks of unstructured data, and they may even be stored in some other format, such as on optical disk, or as documents in some remote online library. So they will be represented in the main file by an identifier of, let us say, 4 bytes.

    The documents contain cross references to each other by this identifier, by which we must be able to locate them rapidly in the system. The geographical and date coordinates should be rapidly retrievable given the identifier. A user should be able to examine documents for all dates at a given location or for all locations at a given date.

    Secondary storage should hold a description of the name and of the address of any data that is stored elsewhere.

    New documents are frequently added, but once there do not often change.

    Suppose there are 10,000 different geographical references, 10,000 different dates, and 10,000,000 different documents.

    Using the categories on which the course is based, and making appropriate calculations approximately, design file(s) to support the above application. Calculate file sizes, and estimate the times to do the operations. Assume a disk with 20 nsec. (nanoseconds) transfer time per byte and 5 msec. (milliseconds) average seek time.

11. There are potentially about 18 million postal codes in Canada, each defining a region on the map which we can suppose to be a polygon of about six sides. Thus, there are about 53 million edges in the map of all postal codes (almost every edge is shared by the two regions it separates).

    This question is concerned with a secondary storage representation of this postal code map, based on the edges. Each region is a cycle of edges, and each vertex (a corner where edges meet) is also a cycle, of the edges meeting there. So the edges, vertices, and regions are described by a structure in which each edge is stored four times, about 211 million records in all. Each edge appears in two vertex cycles (the vertex at the start of the edge and the vertex at the end) and in two region cycles (the region to

the left of the edge and the region to the right). Each edge has an identifier and a direction (so "start", "end", "left", and "right" are established).

[The *above* two paragraphs motivate the following. You need not have read them completely.]

A cycle is stored as a set of edge pairs: an edge, and the edge following it in the cycle. The file to represent the cycles will have four fields, *edgeID*, *edgeDirection*, *successorID*, and *successorDirection*. It will have 211 million records.

In addition, you need to store information for each edge naming its two vertexes and its two regions, and coordinates for each vertex.

There are two kinds of operations that should be supported by your file design. The first is drawing the map of all the postal code regions, or any part of the map. The second is changing the map by means of *splice* operations: *splice*(*edgeID1, edgeDirection1, edgeID2, edgeDirection2*) finds in the file each of the two edges, *edge1* and *edge2*, and replaces it by the other edge. With enough *splice* operations, any rearrangement of the map is possible.

Using the categories on which the course is based, and making appropriate calculations approximately, design file(s) to support the above application. Calculate file sizes, and estimate the times to do the operations. Assume a disk with 20 nsec. (nanoseconds) transfer time per byte and 5 msec. (milliseconds) average seek time.

12. A retail chain is interested in what items customers are likely to buy together, so it can place them on the shelves so as to maximize the time the customer spends in the stores looking for them. So it collects data at the checkout points, on fields *item* and *cart*. (For instance, *cart* 1 contains `bread, butter,` and `coffee`, *cart* 2 contains `bread, butter,` and `milk`, and so on.)

    In a month, a million carts pass the checkout points, with an average of twenty items each.

    When the month is up, another version of this data is created, associating sets of items with sets of carts. Thus, in the above, the set {1,2} of *cart*s is associated with the set {`bread, butter`} of *item*s, while the set {1} of *cart*s associates with the *item* set {`coffee`}. A set is represented by a set identifier in one file, and by pairs in another file, each consisting of this identifier and an element of the set. The resulting system of files must allow programs to count the numbers of elements of each of these sets, and select certain of the pairs of sets depending on the sizes of these counts. (A full analysis is called "association data mining".)

    Using the categories on which the course is based, and making appropriate calculations approximately, design file(s) to support the above application. Calculate file sizes, and estimate the times to do the operations. Assume a disk with 20 nsec. (nanoseconds) transfer time per byte and 5 msec. (milliseconds) average seek time.

13. A genome sequencing project has collected data on the DNA sequences of many species. The data is in terms of base elements which can each have one of four values (called, because of the names of the related molecules, `a, c, g` and `t`). Depending on the species, there may be from tens of thousands to billions of base elements in the genome.

    Three base elements represent one of the twenty amino acids fundamental to the proteins made by biological cells. These amino acids are represented by twenty of the capital letters of the alphabet. Base triplets are called codons, and the relationship between them and the amino acid symbols is called the genetic code. (Note that the genetic code is necessarily redundant: why?)

A gene is a sequence of letters representing these amino acids, as coded by a subsequence of the genome base elements, starting at a specific codon (either `atg (M)` or `gtg (V)` or `ttg (L)`). Genes vary in length from tens to thousands of codons.

Here is a gene of 54 codons from *Mycobacterium tuberculosis*, starting at base element 729 of the genome. The upper line of each pair shows the base triplets, and the lower line shows the translation into the amino acid sequence.

```
atgtcatatgtgatcgcggcgccggaggcgctggtggcggcggccacggatttg
M  S  Y  V  I  A  A  P  E  A  L  V  A  A  A  T  D  L

gctactctcggctcgacgatcggcgccgccaacgcggccgctgcgggctcgaca
A  T  L  G  S  T  I  G  A  A  N  A  A  A  A  G  S  T

acggcgttgctgaccgccggcgccgacgaagtgtcggcggcgatagcggcctat
T  A  L  L  T  A  G  A  D  E  V  S  A  A  I  A  A  Y
```

Design file(s) to represent the sequence of base elements (`a, c, g, t`) for the genomes of these species. You should represent this data compactly. Your system should also represent the genetic code and be able to translate the codons into amino acid symbols given the location of the starting codon. (This location is not necessarily 0 mod 3.) Finally, your system should be able to find genes shared by two different species (even though the genes may start at different positions modulo 3 in different species).

Using the categories on which the course is based, and making appropriate calculations approximately, design file(s) to support the above application. Calculate file sizes, and estimate the times to do the operations. Assume a disk with 20 nsec. (nanoseconds) transfer time per byte and 5 msec. (milliseconds) average seek time.

14. An HTML file uses the HyperText Markup Language to mark up text for display by a Web browser. Markup can also be used to extract meaning as well as to specify display format. Here is an example of a minimally marked up text which illustrates both.

```
<title>tagtest</title>
This year is <year>2003</year>.
Last year was <year>2002</year>.
Next year will be <year>2004</year>.
```

Here, the `<title>` tag causes a browser to display "tagtest" on the title bar of the window. All sorts of other display tags are supported by HTML browsers, such as `<b>` for bold, `<p>` for new paragraph, and so on. The `<year>` tag, on the other hand, is ignored by browsers, but can be used by other processors to extract information from the text. For instance, the above example could be used to generate the file

```
title  tagtest
 year    2003
 year    2002
 year    2004
```

as well as the unmarked text

```
tagtest
This year is 2003.
Last year was 2002.
Next year will be 2004.
```

Imagine a collection of linked web pages with a total of 2Gb (gigabytes) of data (both text and tags). They are to be processed into one file containing nothing but the unmarked text from all pages, and another containing all the tags and their values. The first file may then be used to look up substrings in the text, such as `year will be`. The second file must be available for analyses such as counting the number of different tags (for instance, there are only two different tags in the above example, `title` and `year`), or seeing if a particular value (e.g., `2003`) is present.

Using the categories on which the course is based, and making appropriate calculations approximately, design file(s) to support the above application. Calculate file sizes, and estimate the times to do the operations. Assume a disk with 20 nsec. (nanoseconds) transfer time per byte and 5 msec. (milliseconds) average seek time.

15. A conference submission system is designed to manage both the papers and the refereeing process for a conference. Up to 1000 papers may be submitted, in PDF format, in a four-week period up to a deadline. The average size of a PDF file is about 150Kbytes, but some files could reach 1Mbyte or more. Each author (who may submit more than one paper but is unlikely to submit more than three) is given a password and can replace the paper by a new version any time up to the deadline. Authors may subsequently track the status of their papers through the review process: how many reviews have been completed and, when all are completed, the decision of the program committee and the scores given the paper by the referees.

    Three referees are assigned to each paper, and each referee may review up to a dozen papers. Each referee has a password and is able to submit, for each paper, a score (from 0 to 10), a self-rating of the referee's own competence in the subject addressed by the paper (from 1 to 3), comments to the author, and comments to the program committee. These last two can each run to several paragraphs of comments. The referees may alter these reviews up to a second deadline, when the program committee uses them to make its decisions.

    The system must calculate the average score for each paper, from the three scores given it by the referees. It must be able to merge all the referee comments on each paper for presentation to the program committee.

    The program committee must select fifty papers from the submissions. When the program committee has decided whether to accept or reject the paper, the system must be able to post this decision, the three scores and referee self-ratings, and the comments, so the author can look up this response. It also must send an email to the author, with the decision to accept or reject. Finally, it must extract the selected papers for printing in the conference proceedings.

    Using the categories on which the course is based, and making appropriate calculations approximately, design file(s) to support the above application. Calculate file sizes, and estimate the times to do the operations. Assume a disk with 20 nsec. (nanoseconds) transfer time per byte and 5 msec. (milliseconds) average seek time.

    For this question, please do a full secondary-storage analysis irrespective of the size of the file.

Copyright ©2006 Timothy Howard Merrett
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation in a prominent place. Copyright for components of this work owned by others than T. H. Merrett must be honoured. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or fee. Request permission to republish from: T. H. Merrett, School of Computer Science, McGill University, fax 514 398 3883.

The author gratefully acknowledges support from the taxpayers of Québec and of Canada who have paid his salary and research grants while this work was developed at McGill University, and from his students (who built the implementations and investigated the data structures and algorithms) and their funding agencies.