THE INFLUENCE OF DATA-BASE ON CONSULTING ACTUARIES

M S. Spry, (Australia)

1. INTRODUCTION TO DATA-BASE

1.1 Why is Data-Base Important?

The term data-base has recently risen to a high position in the list of computer jargon. To a non-computer person, it means little or nothing, and many computer-people do not really understand data-base concepts.

This paper attempts to throw some light on the subject for non-computer people who have an understanding of life insurance and superannuation. I have tried to remove the jargon from the explanation, and to the extent that I have failed I offer apologies in advance. To the extent that I have succeeded I offer apologies to readers who class themselves as computer-people.

Data-base technology has a particular relevence to consulting actuaries for two major reasons. Firstly, many consulting firms have experienced some involvement in computer systems work and regard computer knowledge as an extension of actuarial expertise into the wider field. Secondly, consulting actuaries are continually making assumptions about long term expense rates, and long term expense rates are bound to depend on the influence that computer technology can have in the future. An understanding of the likely impact of the computer in future would at present be incomplete without an understanding of data-base.

1.2 What is Data-Base?

There has been much literature written recently attempting to answer the question "what is Data-Base?" I make no references to particular papers, since in this paper I attempt to give my own views using a life insurance application as an example. The very fact that it appears so difficult to explain what data-base is reflects the complexity of the ideas involved in data-base and therefore the difficulty which laymen have in grappling with it.

There are many ways in which the term data-base is used, but in my mind there are both broad and narrow concepts involved.

The narrow concept is that of the physical data-base, or the layout of records on disk. The broad concept is of a set of data-base techniques, being a whole approach to computer systems. This approach includes –

(i) the principles of modular programming and structured design (to make programmers' and analysts' lives easier and to reduce the maintenance burden);

(ii) the principles of data-independence or of changing the layout of data without affecting the programs using the data;

(iii) the use of proper documentation of systems and the idea that the documentation of a system is a part of the data handled in the data-base;

(iv) the use of high-level languages to reduce the amount of computer expertise needed;

(v) a stronger emphasis on the importance of back-up and recovery of data files; and

(vi) the most important principles of data structure which are incorporated in the narrow definition.

A data-base approach to computer systems involves placing a great deal of importance on the data of the company. Data is a resource of the company, in the same way as money is a resource. We have complicated social structures and complicated computer structures to handle money – yet in the past we have never had a data-accountant or a data-auditor. Data-base technology emphasises the importance of data.

Each of the abovementioned components of a data-base approach is deserving of a whole paper, and each has been the recipient of a great deal of attention from various parts of the computer industry. A full understanding of the wider concept of data-base requires an understanding of each of the components. However, much understanding can be gained by considering the narrower concept.

The narrower concept of data-base is that of a "data-base file organisation", or a data-base approach to structuring data. This is a method of laying out data within the computer (usually on a magnetic disk) so that it is most conveniently accessible. Non-data-base systems typically have data on a master file arranged in "records" which have some kind of key (e.g. policy number for a policy master file). To access any piece of information held in that record it is necessary to know the policy number or to look at every record. Knowing the name of the life assured, or the post-code of his address would not be sufficient to find, say, his date of birth. The "entry point" to the data is only the one record key.

Early data-base systems attempted to provide multiple data entry points by constructing separate files which were index files, so that looking up the index gives the record key. More recently, however, a large number of different ways of providing multiple entry points for the same piece of data have been developed. The second section of this paper is devoted to an example of the use of multiple data entry points.

Many firms, including the hardware manufacturers, independent software suppliers and even the users themselves, have worked out different techniques for storing and accessing data. The data

facilities are usually made available through a set of programs or a programming language which supports the technique. This is called a Data-Base Management System, or DBMS.

There are approximately 400 DBMS's available in the market ranging from very basic file handling programs, through straightforward data mangement systems, to complicated systems providing a vast range of alternatives and requiring a great deal of expertise to use. Each DBMS has its own name, and the names of some are:–

IMS
DL/1
TOTAL
ADABAS
MARK IV
SYSTEM 2000

The investigation of the systems available and the choice of one of them is a time-consuming initial task for those contemplating beginning a data-base system.

A DBMS generally deals with only the narrow concept of data-base. However, it will encourage the user to use the broader concepts by providing some of the facilities necessary or simply by creating an awareness of the broader problems.

1.3 Why Data-Base?

Obviously the purchase of a DBMS, the provision of a machine which can handle the obviously more sophisticated functions, the education of programming staff and others in all the broad concepts, and the risks involved in making the many choices required, are all going to be quite expensive. Why do it? What is so wrong with the old approach of sitting down a programmer with a bright idea and a coding sheet and saying, "write out a program to do such-and-such?" A real danger exists for many companies that the only answer given is "because it is the latest and so must be good". Data-base has such a large impact on the cost of an EDP development project that it should be properly investigated before it is used, and this investigation obviously takes time.

The computer industry has come to realise what is wrong with the old apporach. While spectacular benefits can often be gained from modest efforts, the amount of effort expended in gaining further benefits tends to grow exponentially and becomes impracticable. Endless conversions from outmoded systems to only slightly better ones typically gobble up programming and machine resources. The basic problem of only one entry point to the data still remains. The problem is represented graphically in figure (i). The area under the curve is the total cost of the system to date.

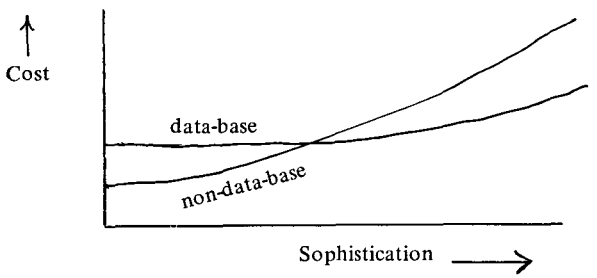


Figure (i)
Two Approaches to Computer Systems

Perhaps it is the predominance of this feature which has caused computers generally to fail to realise their full commercial potential in a reasonable time scale. Once an installation sets out on the old path, it is not easy to move to the new.

If an installation were to commence a system using data-base, the high initial cost can be expected to pay off dramatically as sophistication of the system increases, as shown on the graph.

For conversion of an existing non-data-base system, the cost curve would represent a complicated combination of the two simple curves, but could still be expected to be flatter at higher levels of sophistication.

In summary, the decision to "go data-base" must be made in the light of some preconceived idea as to the required eventual level of sophistication of the system and the relative areas under the curves at that level of sophistication.

1.4 Data-Base vs Real-Time

In another dimension in the computer industry there is the move from "batched processing" to "real-time" systems. Briefly, with batched processing all the transaction being processed go through the first procedure, then all through the next procedure, and so on. This is a typical manual approach extended into the computer. With real-time, however, a transaction is entered into the computer via a terminal and is dealt with in its entirety before the next transaction is entered from that terminal. The former is procedure-oriented, the latter is transaction-oriented.

The most effective use of data-base is with a transaction-oriented system, and therefore "data-base" tends to be linked with "real-time". It is not unusual, however, to have data-base systems which are procedure-oriented, and it is not unusual to have transaction-oriented systems which are not real-time.

If "data-base" conjures up thoughts of a huge machine and many real-time terminals, it shouldn't. Use of data-base is independent of the batch-processing vs real-time decision. However, it is most effective in a transaction-oriented system.

2. DATA-BASE AND LIFE INSURANCE

2.1 Need for Data-Base in Life Insurance

The long-term nature of a life insurance contract and its inherent complexity are the two major factors influencing the design of a life insurance computer system. The systems designer finds himself confronted with a bewildering array of old policy types as well as the ones currently in use, and in addition there is a wide range of transactions to perform the basic record-keeping functions. Most of these problems must be tackled before the information requirements of the office can be met.

2 2 Complexity of Transactions

Many of the transactions mentioned above, although appearing to be straightforward, are in fact quite complicated. Apart from the obviously complicated actuarial programs, transactions such as premium billing can be complicated. To work out how much premium is to be paid on a policy requires consideration of the premium due on each benefit of the policy, whether or not that benefit is premium paying at the time, the possibility of increasing or decreasing premiums, the possibility of premiums paid in advance or arrear and any interest thereon, etc.

2.3 Data Accessing Requirements

Most of the transactions are related to policies and require accessing only the records for a specified policy. Non-data-base systems are able to handle these quite easily using a sequential file of policy records. Some transactions, however, require accessing policies in groups. For example, those policies paying premiums by deduction from pay must be accessed according to employer; preparation of agents' commission statements requires policies by agent; administration of superannuation schemes requires grouping by scheme, and a reinsurance system requires grouping by reinsurance company.

Invariably, the addition of these functions to the typical sequential file systems causes major difficulties within the system. A data-base approach, providing multiple data entry points, is most useful for these transactions.

2.4 Information Requirements

For many of the reasons outlined above, the energy of life office systems designers has of necessity been concentrated on the record-keeping aspects of the system. Even these have caused problems. The pressing needs of the office for thorough actuarial management of all aspects through a comprehensive information system have been largely neglected.

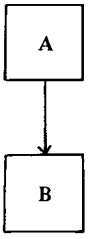
Up-to-date information on new policies being submitted, on policies lapsing and surrendering, on budgeting and "profit and loss" statements is rarely produced by life systems.

2.5 Data Structure Principles

A data structure design involves grouping pieces of information together logically (making records) and defining the logical relationships which exist. Relationships can exist between records in different ways. They are best illustrated using diagrams and record types A, B etc.

(i) Simple relationship

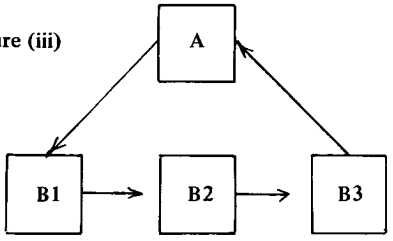
Figure (ii)



Given record A, record B can be found.

(ii) Ring Structure, Single Record-Type Rings

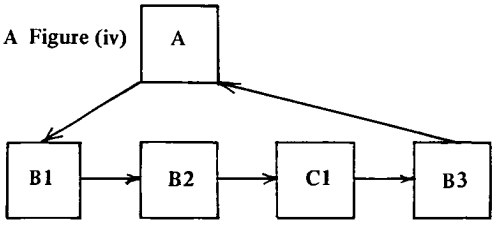
Figure (iii)



Given record A each record of type B "belonging to" A can be found in turn. Record B2 cannot be found without finding B1, etc.

(iii) Ring Structure, Multiple Record Type Rings

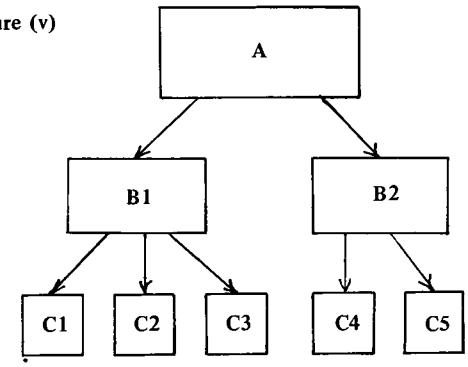
A Figure (iv)



Given record A, each record of types B or C "belonging to" A can be found in turn.

(iv) **Hierarchical Structure**

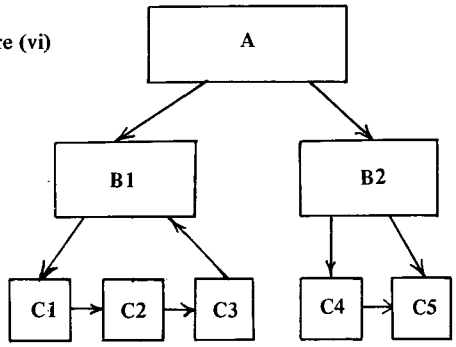
Figure (v)



Given A, each record of type B can be found.
Given B, each record of type C can be found.

(v) **Hierarchical and Ring Combination**

Figure (vi)



(vi) **Inverted Heirarcical Structure**

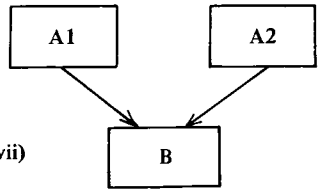


Figure (vii)

Given A1 or A2, B can be found.

It can be seen that there are a number of different ways of presenting relationships between records, of which the above are but examples. The remainder of this Section uses the hierarchical structure for illustration.

2.6 **Life Insurance Data Structure**

The logical groupings of data which can be used to represent life insurance business of a typical company are:–

Basic Policy Information (BP) – being the status (in force or out of force), policy number and so on – the very basic data. One per policy.

Billing and Accounting Information (BA) – the premium instalment, premium balance, due date of next premium, method of payment, etc. One per policy.

Policy Benefit Information (PB) – the sum assured, premium, date risk commenced, date of premium and benefit cessation, type of assurance, details of regular variations to premium or sum assured, etc. One for each benefit payable under the policy.

Reassurance Information (RE) – sum assured reassured, premium, reassurance basis etc. One for each reassurance of the policy.

Bank Deduction Authority Information (BD) – the amount of the authority, bank, account number, account name etc. One for each deduction regardless of how many policies are paid through it.

Employee Deduction Information (ED) – the amount deducted from an employee's pay, etc. One for each employee paying to the company.

Employer Information (ER) – the total remitted to the company by each employer with a deduction scheme, the name and address of the employer and so on. One for each employer.

Client Information (CL) – the name, date of birth, sex, underwriting information and so on of each client of the company.

Salesman Information (SL) – name and other details of each salesman who receives the commission for a policy.

Salesmans Production History (SP) – details of production by salesman for analysis purposes.

Policy Commission Information (CO) – details of the commission due to an agent in respect of a policy.

Reassurance Treaty Information (RT) – details of each treaty under which reassurances are effected.

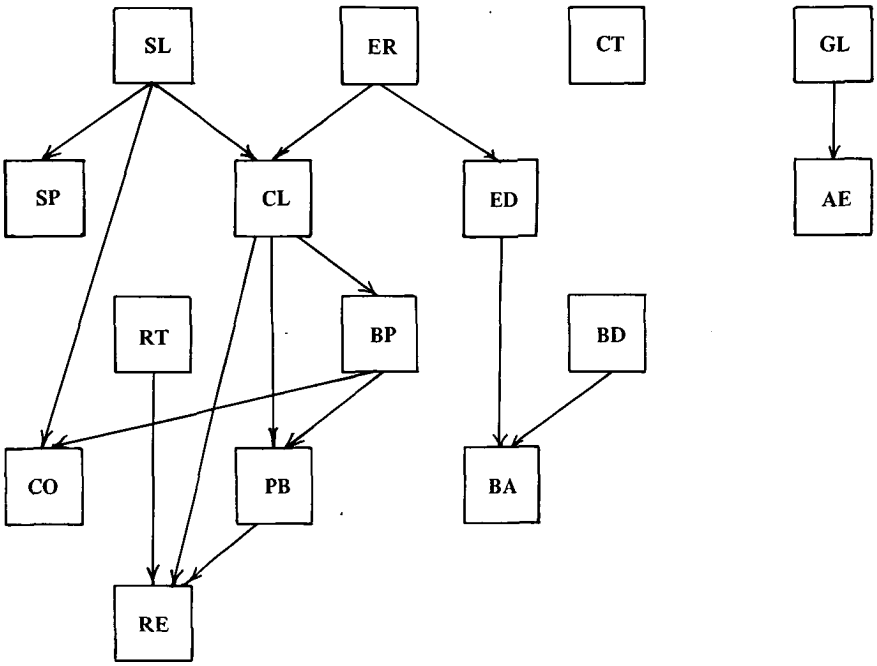
General Ledger Account Information (GL) – name of each general ledger account of the company and the balance from time to time.

Accounting Entry Information (AE) – details of each accounting entry posted to the general ledger.

Control Information (CI) – details of new business conditions in force from time to time, bonus rates, valuation bases, surrender bases and so on to avoid such items being programmed directly into programs.

Other possibilities include details of investments, of company employees and so on. They are omitted here for simplicity. Figure (viii) presents the above records in a hierarchical structure.

Figure (viii)



It can be seen by examining the data structure in Figure (viii) that given a policy number, all the information about the policy can be found (relationships BP→ BA, BP → PB, BP→ CO) Further, given a particular client, all his policies can be found (CL → BP), as can all benefits on his life (CL → PB) and all reassurances on his life (CL →RE). Given a salesman, his clients (SL →CL), his production history (SL → SP) and his commission earnings (SL → CO) can be found. All reassurances under a treaty (RT → RE), all payments in a bank deduction authority (BD → BA) or by an employee (ED → BA) working for an employer (ER→ ED) can likewise be found.

A structure of this form obviously provides a high degree of utilisation of data without duplication. It is also a reasonably complicated structure and many DBMS's would have some difficulty in supporting it.

Before applying such a structure in practice, it would be necessary to devote considerable time to detailed thought about the contents of each record, their relevance for the particular application, and the likely performance of the chosen DBMS to support that structure. However, the structure does serve as an example of the narrower concept of data-base.

3. CONCLUSION

Data-base technology is not entirely new. Many aspects have been around for some time. The change in philosophy which emphasises the importance of data is, however, quite new. With data-base the computer industry has developed a powerful new approach to its old problems.

But what are the disadvantages? I have already mentioned the high initial cost. Some others are:–

(i) Maintaining a large number of relationships can be very time-consuming in running since an alteration to one record may involve altering many relationships and therefore many other records.

(ii) Information which is logically related (e.g. two records on the same ring) may be physically far apart and so require substantial disk activity.

(iii) Occasionally, for one of many reasons, a data-base disk file must be unloaded (all records logically unlinked and written to tape) and reloaded (all records relinked and the data-base recreated). This procedure can take incredibly large machine times.

(iv) The use of a data-base is more difficult for the programmer than sequential files. This complication effect is worse in a real-time environment. Therefore, some specialist programmers must have key roles and some generalisation is necessary for applications programmers.

Data-base is a vast new challenging area for most computer installations. This paper has aimed to begin to build an understanding.