A GENERALIZED INFORMATION SYSTEM FOR ARCHAEOLOGICAL USE

Alfred H. Kromholz.

Abstract

This paper presents a generalized data storage and retrieval system highly suitable for archaeological use. The system will operate on nearly any data-base format, contains provision for almost all desirable data retrieval operations, and is extremely easy to use. The program will work with single and multiple files and includes, among other features, the facility for multiple iterative scans of the data base. It is written entirely in FORTRAN IV and is extremely easy to expand or modify.

Introduction

In his paper given at this Conference last year, Cutbill presented a series of requirements for an "ideal" generalized data management system. Among his desiderata were that the program be independent of the data format, provide for manipulation of multiple files, include an information search and retrieval capability, have some form of security system to protect the files, and be easy to learn and use. Another highly desirable feature not included in Cutbill's list is that the system be able to operate from remote terminals, that is, at typewriter keyboards not necessarily located at the computer itself.

The system summarized below is one that fulfills all of the above requirements and that is designed for easy expansion to include or link up with other features such as sorting, report generation or statistical processing programs. The system was originally intended as a prototype data management program for use in social service applications in the United States and Britain. At the time the system was set up, neither the file formats not the commands for operating the program were finalized. As a result, the program had to be designed for maximum flexibility in these areas. In addition, the final system was intended to function simultaneously as a central data processing facility and as a multi-terminal remote-access service (i.e. operating with more than one keyboard in use at a time, all at different possible locations away from the central computer). Consequently, the program was designed to operate as part of a larger time-sharing system and includes provision for input and output of information "off-line" - that is, not at the user's keyboard. This prototype system was fully operational some two and a half years ago.

Before proceeding I should point out that the original purpose of this system - the processing of social welfare information - in no way decreases its applicability to the field of archaeology. As an archaeologist, I deliberately designed the program for eventual archaeological use, although I incorporated certain additional features and messages to the user that were specifically orientated for the social services. Since the original project was being carried out for the Heller School of Social Welfare at Brandeis University and was supported by funds from the Massachusetts Department of Social Welfare, this was clearly a necessity.

General System Description

In describing the system, the first thing I should mention is that the program normally operates in a conversational mode. When the user types in a one-word command at his keyboard, the system asks a series of questions in order to obtain the detailed specifications of the user's request. These questions take one of three forms: those that can be answered by a simple YES or NO, those that require the user to choose an item from a list by typing the number of the item, and those that expect some form of text, such as a date or a piece of data. For each question, the user types an answer and the system checks the appropriateness or correctness of the answer. After all of the details in the user's request have been specified, the system carries out the command. Should the user make an error in either the command itself or the subsequent detailed specifications, the system points out the error and asks for an immediate correction. The question-and-answer sequences are not particularly lengthy, but they can become slightly annoying when performed many times in succession. To alleviate this problem, a number of abbreviated methods of entering the data are included for the experienced user.

There are seven basic commands in the system: SIGNON, ENTER, ALTER, EXAMINE, LIST, SELECT and SIGNOFF. The SIGNON and SIGNOFF commands are essentially self-explanatory, in that they provide the means of access to and egress from the system. The ENTER command is used for inserting a new entry into one of the existing information files from the keyboard. In the current implementation, no provision is included for mass entry of data from a source other than the keyboard. Nevertheless, the system was specifically designed to allow the easy incorporation of this feature.

The ALTER command is used to change the information contained in an existing entry in one of the files; it is also used when deleting an entry. The EXAMINE command is used to look at individual entries in the files, as identified by the name of the entry. The LIST command allows the user to print out functional divisions of the file, either at the keyboard or, when the feature is installed, at an off-line printer separate from the keyboard. Neither the LIST command nor the EXAMINE command is intended for general use in the selective retrieval of data from the files. This function is accomplished by means of the SELECT command.

Before describing the SELECT operation, though, several other aspects of the system must be discussed. The program incorporates rather extensive security protection features. Every command has at least one, and, frequently, multiple authorization codes to inhibit unauthorized changes to the files and to prevent unauthorized access to restricted information. The program as a whole has a general password system, and individual information files also have internal authorization codes. When a user attempts to perform an operation for which he is not authorized, the system simply types out: SORRY. YOU ARE NOT AUTHORIZED FOR THAT REQUEST. The system also has the facility for recording the attempted violations of the security system; this is an outgrowth of the social service orientation of the prototype and can be included or omitted at the discretion of the individual installation.

In addition to the basic commands already listed, there are several others that ought to be mentioned in passing. For the experienced user, there is a command (BRIEF) to shorten system messages from the lengthy, conversational, self-explanatory versions to a short, simple form; similarly, there is a command (FULL) to restore the messages to their normal length. If a record is kept of the security violations, there is a command (SECURITY) for printing out the list of violations and for clearing the record. There is a command (MESSAGE) for recording and typing out messages to other system users; this may have limited utility in a strictly archaeological application, but it can be retained if a particularly large installation has a need for it. There are commands for examining information contained in tree-structures and for generating those tree-structures. (See Figure 1.) This facility may also be of limited archaeological use except in a rather extensive data-bank application. Finally, there are several commands for use by those persons who are responsible for the maintenance of the program and its files. There is a command to add new authorizations to the system and to the information files (IDENTS), and a command to add and delete information files (CREATE). There are also commands to generate the system from scratch (the so-called "dead start" operation), expand the file capacity when more space is needed, locate the physical position of a specific entry in the file and write machine-code information directly into specific locations in the files.

There are four types of files in the system. The first is used for internal system management. In an archaeologically-orientated system, only one file of this type may be required: that containing the identification and access codes of those individuals authorized to use the system. Other files in this category, such as the security file and the message file, may or may not be included, depending on the requirements of the specific installation. The second type of file consists of those used for the tree-structures mentioned previously; these, too, are included in the system only when needed for a particular application.

The third type of file contains the actual archaeological information. When this information file is set up, the entries in each section of the file must be specified as taking one of two possible forms: a single line of data, or a series of lines under a major heading. (See Figure 2.) In order to avoid the restrictions imposed by short line lengths occasionally required by specific computers, the program has provisions for continuation lines as shown in the figure. Regardless of whether an information file section employs single-line entries or major-heading entries, the actual arrangement of the information in the entries is not fixed by the program; rather, the arrangement is specified in the fourth type of file, the "entry-description file".

In general, every individual application of a data storage and retrieval system will have its own format, or set of formats, for the information to be stored in its data bank. Rather than placing programmed-in restrictions on the file formats, the system being presented allows the use of essentially any arrangement of the data. An individual user can specify the manner in which his data are to be stored by specifying, for each item of his data, the name or label by which the item is to be called, the type of data that the item represents (text, date, integer number, or decimal number), the number of computer words that the item is to occupy, and the starting position of the item in the entry line (see Figure 3). A series of these four values for each item of data in the entry line constitutes a complete entry-description group. At the current time, only one group of entry descriptions is included in the system at any one time. However, provision for multiple groups with different formats can be included with only a slight change to the program.

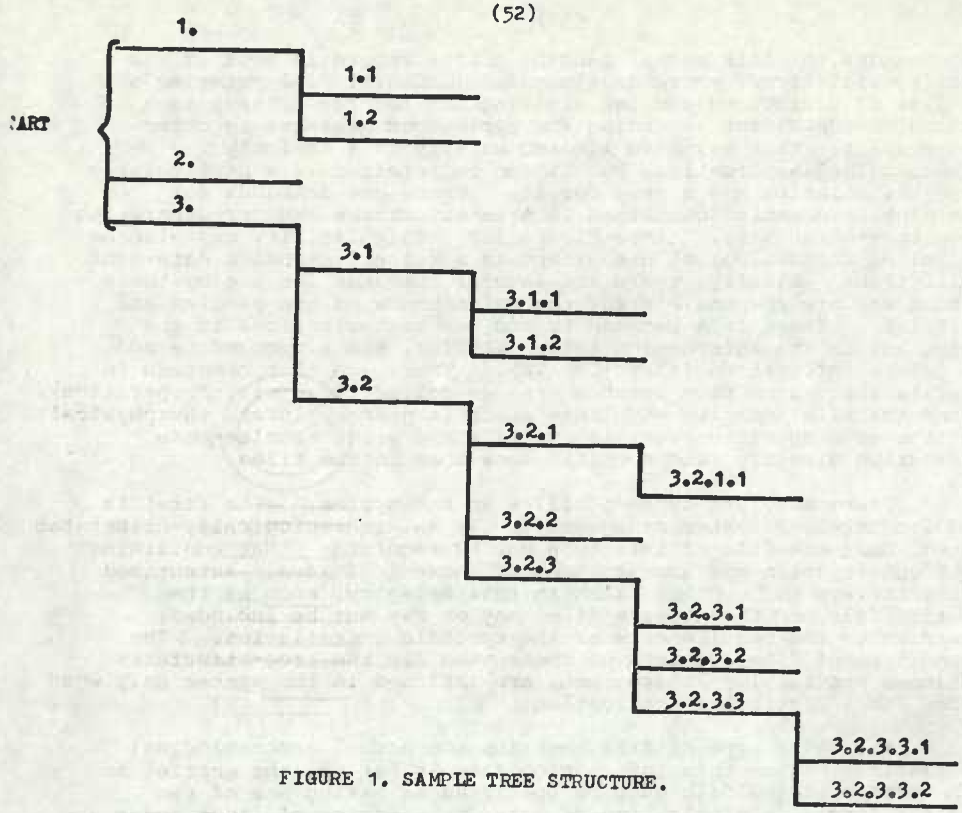
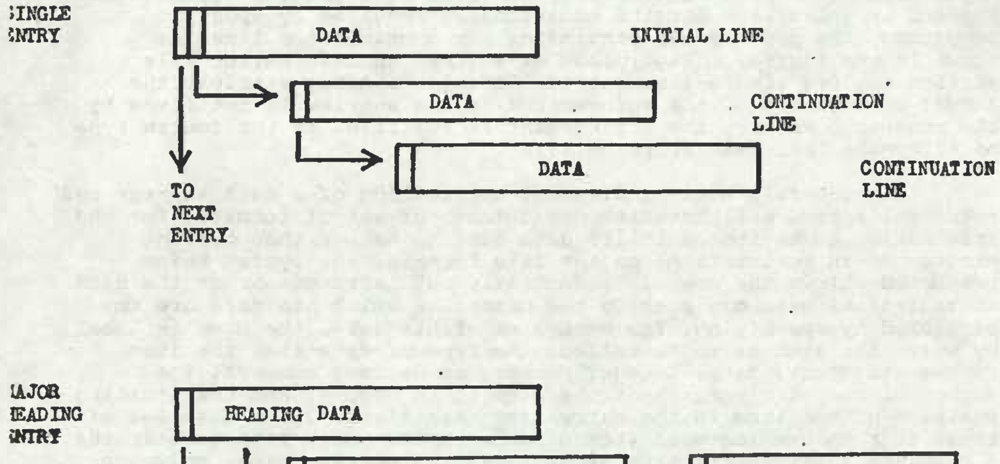


FIGURE 1. SAMPLE TREE STRUCTURE.



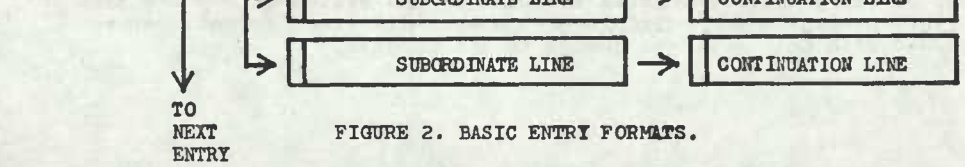


FIGURE 2. BASIC ENTRY FORMATS.

The SELECT Command

As was stated earlier, the SELECT command is the means by which related items or file entries can be retrieved from the data bank. When the user types in the word SELECT, the system asks the user to enter the criteria by which the desired entries are to be selected from the file. After checking the set of criteria for correctness of format, the system examines the entry-descriptions for the file and verifies that each selection criterion typed by the user contains both a legitimate label of an item of data and the appropriate type of comparison for that item. (For example, the criterion (MATERIAL<10) would not be acceptable. Although the word MATERIAL is included in the list of labels in Figure 3, it is identified in the list as an alphabetic item of data. Whereas a comparison for equality, such as (MATERIAL=BASALT), would be acceptable, a test for "less than" makes no sense and would be rejected by the system.) After all of the selection criteria have been checked in this manner, the system scans the entire information file and identifies the entries that meet the criteria. The system then tells the user how many entries met the criteria and gives him the option of having those entries printed out.

The foregoing description is roughly similar to that for almost any modern data retrieval system. What makes this system unique is the range and nature of the available options.- although some of these may be found individually in other programs, their combined features are not, as far as is known, available in any other published system. First, a single typed line may consist of multiple independent criteria in any logical arrangement. One can perform the usual simple tests, such as (MATERIAL=BASALT). One can also specify combinations of tests; e.g. "(MATERIAL=FLINT) and (QUANTITY>10) <u>or</u> (MATERIAL=OBSIDIAN) and (QUANTITY<2))".

Second, in multiple-item criteria such the above, the results of the file scan are accumulated not only for the overall selection criterion but also for each of the individual criteria and for the logical operations that join them. Thus, for the previous example, the system counts the number of entries having (MATERIAL=FLINT), those having (QUANTITY>10), and those having both (i.e. the "and" joining them); the number of entries having (MATERIAL=OBSIDIAN), those having (QUANTITY<2), and those having both (i.e. the second "and"); and the final count, which is the result of the "or" joining the two groups. The system prints out the final count automatically; if, in addition, the user asks to see the intermediate counts, the system will print out all of these values.

A third feature of this system is that comparisons may be performed between items of data within a single file entry. Such comparisons may be expressed in two different ways: as direct comparisons or as calculated comparisons. To illustrate: assume that two quantities of material are specified in a single entry, the quantity of flint being denoted by the label FL_QTY and the quantity of obsidian by OB_QTY. One can select those entries for which the quantity of flint is less than the quantity of obsidian by performing the direct comparison (FL_QTY<:OB_QTY). One can select those entries in which the ratio of flint to obsidian is less than ten to one by performing the calculated comparison (FL_QTY/OB_QTY<10).

The system also includes the capcability for performing "iterations" on previous selections from the information file. After the system executes a SELECT command by scanning the information file for entries that met some particular selection criterion, the list of selected entries is temporarily saved. At the end of the

SELECT operation, the system allows the user to perform a further SELECT scan on the results of the previous one. For example, assume that the user had selected all of the bowls in the file by means of the criterion (OBJECT=BOWL). He could then select all entries having bowls made of andesite by performing an "iteration" on the results of the first selection using the criterion (MATERIAL=ANDESITE).

The results of this new iteration are stored by the system along with the results of the original selection, and the user can perform further iterations based on either of the previous selections. Thus one could select all entries having more than ten andesite bowls by scanning the results of the previous iteration using the criterion (QUANTITY>10). Similarly, the user can go back and select all entries having more than five basalt bowls by scanning the initial set of results using the criterion: (MATERIAL=BASALT) and (QUANTITY>5).

This process of iteration - using the results of a previous selection as the basis of further selections - is a very powerful tool in data retrieval operations. The system allows up to eighty such iterations in a single SELECT operation. Note that results of these iterations are not preserved indefinitely within the system. When the user informs the system that he has finished with the current set of SELECT operations, the stored lists of all prior iterations are erased.

Conclusion:

The system just described was written entirely in FORTRAN IV and was in successful operation on both an IBM-360/67 and an IBM-370/165. A version is being prepared for the ICL-4000, although that conversion has been temporarily suspended. Plans are also underway to set up the system on a Digital Equipment Corporation PDP-10.

The program as currently available is still basically in the form originally used for the social services; up to this point there has been no pressing need to remove the special features that were incorporated for those purposes. Aside from the preparation of new documentation, no more than two to three weeks would be required to set up the system for strictly archaeological use. When this is done, a number of small improvements will also be included.

First, a file creation subroutine will be incorporated to allow information to be read from cards and other input media and to be converted automatically to the more efficient internal format of the information files. Second, the facility for off-line listing of information will be added in accordance with the requirements of the particular installation. Third, multiple sets of entry-descriptions for the information file will be permitted. Fourth, an m-out-of-n option will be added to the allowable operations of the SELECT command. Finally, the entry-description file will be modified to allow variable-length items in the formats.

One of the major design goals of the original system concept was that the actual operation of the program would be easy enough not to discourage the non-computer-orientated user. This was achieved completely successfully; people were literally fascinated by operating the prototype during field trials! This encouraged me with the adaptation of the program to archaeology, and those archaeologists who have played with the system have thoroughly enjoyed it.

The first archaeological project to employ the system is the preparation of a Near East site index and bibliography now being carried out at Brandeis University and on Cyprus. Although recent events in Cyprus have somewhat impeded our progress we hope to have a first version of the index essentially complete about now.

Any further projects for using the system would be most welcome.

CATEGORY		MATERIAL		OBJECT		AREA	QTY	unused			
1	2	3	4	5	6	7	8	9	10	11	12

	label	type	starting position	length
entry-description	CATEGORY	1 (alphabetic)	1	2
	MATERIAL	1	3	2
	OBJECT	1	5	2
	AREA	1	7	1
	QUANTITY	2 (integer)	8	1

sample entries	GR_STONE	BASALT	QUERN	H2	3
	CH_STONE	OBSIDIAN	BLADE	XY	73
	POTTERY	RED_POL	JUGLET	T5	154

FIGURE 3. SAMPLE ENTRY-DESCRIPTION FOR THE SINGLE-ENTRY FORMAT.
(based on up to four characters per computer word)