DATAPLUS—A language for real time information retrieval from hierarchical data bases

by NORMAN R. SINOWITZ
Bell Telephone Laboratories
Holmdel, New Jersey

INTRODUCTION
To the average person, a computer user is synonymous to a programmer. In fact, to the average programmer, a user is synonymous to a programmer. Information retrieval is an area in which the computer user is not—or, at least, should not be—required to be a programmer. This paper describes a real time information retrieval system which is at once highly accessible, relatively inexpensive, and simple enough to be used by nonprogrammers.

The system was implemented early in 1967 on a GE-265 computer operating under the time-sharing monitor developed by the Missile and Space Division of the General Electric Company (Valley Forge, Pennsylvania). Access by users is thus from remote teletypewriters communicating over standard telephone lines.

By implementing a system on a commercially available time-shared computer service, the effort of developing the time-sharing software is eliminated, and the cost of the development is spread over the many users of the service. The retrieval system was field tested without any investment in expensive hardware. By providing the system with a powerful information processing ability, highly efficient use can be made of the potential information content of the data, with a consequent high performance/cost ratio of the system.

What makes the system easy to use is its query language. Called DATAPLUS, the name derives from the system's ability to access data, plus the ability to process data. The DATAPLUS "compiler," which can be more accurately described as an incremental interpreter, was implemented in FORTRAN, and was designed for real time information retrieval from a hierarchical data base.

At his remote terminal, the DATAPLUS user types his information request in the form of a message consisting of a continuous flow of statements resembling English. Erroneous or unintended statements can be altered by merely retyping them. Providing the user with these facilities has in most instances overcome the inertia people have against learning to "program." Furthermore, these facilities enable the user to spend more time thinking about his problem, because he is, to a large extent, thinking in the way he is accustomed.

Since a data item is often meaningful at a level of the hierarchical structure other than that at which it appears in the data base, flexibility is provided for addressing and manipulating the data at various levels. The language is also "open-ended" in the sense that the user is given the ability to create functions of items appearing in the data base, and to instruct the computer to operate on these functions in the same way that it operates on items already in the data base.

DATAPLUS was implemented in order to retrieve information useful to engineers at Bell Telephone Laboratories. So as not to burden the reader with telephone jargon, we shall exemplify our discussion by referring to a familiar (but hypothetical) data base. In addition to being intelligible to a wide audience, the hypothetical data base was chosen because it is structurally isomorphic to the data base for which the system was implemented.

The data base
Consider a department store chain which has branches in a number of the larger cities in the conterminous United States. The company operates throughout most of the country, with a corporate headquarters, and separate divisions in the various states. The state organizations are further broken down by cities. Figure 1 shows the hierarchical structure for this data base.

At the top is the corporate level. The second and third levels represent, respectively, the state and
city divisions. The focal level is the fourth level, that is, the STORE level. Each store has one or more departments, and each DEPARTMENT sells at least one ITEM.

A variable whose value is intrinsically numeric will be called a *range* variable; a variable whose value is intrinsically alphameric will be called a nonrange variable. Thus EARNINGS and IN STOCK are range variables, while STORE NAME and INVENTORY NUMBER are nonrange variables.

**General description of the syntax**

Like any language, natural or artificial, DATAPLUS has syntax, semantics, and vocabulary. Syntactically and semantically, DATAPLUS resembles English. The basic vocabulary consists of a few key words, such as “total,” “distribute,” “and,” “or,” “when,” and a set of nouns representing items appearing in the data base. This section gives an overall picture of the syntax. Vocabulary and semantics are considered only when necessary to the discussion.

We begin by citing three examples of messages written in the DATAPLUS language.

**Example 1**—Suppose we wish to find the total number of stores in Michigan and New Jersey which have any HI FI departments. The message could be ...

```
TOTAL STORES: IN MICHIGAN; NEW JERSEY: WHEN ANY DEPARTMENT NAME = HIFI: GO:
```

**Example 2**—Suppose we wish to obtain a frequency distribution of stores versus dollar sales per store in Miami and Tampa. The message could be ...

```
DISTRIBUTE STORES: BY DOLLAR SALES PER STORE: BETWEEN 0 AND 2000000 IN STEPS OF 20000: IN FLORIDA, MIAMI, TAMPA: GO:
```

**Example 3**—Suppose we want to extract the store name, the names of the departments, and the earnings/sales ratio for stores in Cook County, Illinois. The message could be ...

```
LET RATIO = EARNINGS/DOLLAR SALES PER STORE: EXTRACT STORE NAME, RATIO, DEPARTMENT NAME: FROM STORES: IN ILLINOIS: WHEN COUNTY = COOK: GO:
```

A message in DATAPLUS thus consists of a continuous flow of statements,* each beginning with a key word, such as "TOTAL," "WHEN," "IN," "LET," and ending with a colon. The final statement in the message must contain the single word "GO."

Every message must have an "IN statement" i.e., a statement beginning with "IN." This informs the program in which states and cities the user is interested.

<table>
<thead>
<tr>
<th><strong>Variable Name</strong></th>
<th><strong>Level</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>STORES</td>
<td>Store</td>
</tr>
<tr>
<td>STORE NAME</td>
<td>Store</td>
</tr>
<tr>
<td>ADDRESS</td>
<td>Store</td>
</tr>
<tr>
<td>COUNTY</td>
<td>Store</td>
</tr>
<tr>
<td>EARNINGS</td>
<td>Store</td>
</tr>
<tr>
<td>DEPRECIATION</td>
<td>Store</td>
</tr>
<tr>
<td>DEPARTMENTS</td>
<td>Dept</td>
</tr>
<tr>
<td>DEPARTMENT NAME</td>
<td>Dept</td>
</tr>
<tr>
<td>SALES FORCE</td>
<td>Dept</td>
</tr>
<tr>
<td>DOLLAR SALES</td>
<td>Dept</td>
</tr>
<tr>
<td>ITEMS</td>
<td>Item</td>
</tr>
<tr>
<td>ITEM NAME</td>
<td>Item</td>
</tr>
<tr>
<td>INVENTORY NUMBER</td>
<td>Item</td>
</tr>
<tr>
<td>IN STOCK</td>
<td>Item</td>
</tr>
<tr>
<td>ON ORDER</td>
<td>Item</td>
</tr>
<tr>
<td>PURCHASE PRICE</td>
<td>Item</td>
</tr>
<tr>
<td>SELLING PRICE</td>
<td>Item</td>
</tr>
</tbody>
</table>

*A statement in DATAPLUS is either a complete sentence, a prepositional phrase, or an adverbial clause.
There are three basic operations in DATAPLUS as implemented to date: TOTAL, DISTRIBUTE, and EXTRACT. Every message must contain a statement that begins with one of the basic operation words TOTAL, DISTRIBUTE, or EXTRACT.

Any range variable may be summed by using the TOTAL statement. DATAPLUS automatically supplies an average along with the total. Thus, "TOTAL EARNINGS:" will cause the system to furnish the average earnings/store.

If there is a DISTRIBUTE statement in the message, there must also be a BY statement and a BETWEEN statement. If we visualize a typical histogram, the DISTRIBUTE statement contains information pertaining to the Y axis variable; the BY statement contains information pertaining to the X axis variable; the BETWEEN statement contains information pertaining to the limits of the X axis and the class interval size.

A WHEN statement is optional in every message. Its purpose is to specify the search. In Example 2, the search is implicitly specified as being over every store in Miami and Tampa; in Example 3, the search is explicitly specified for stores in Cook County.

The LET statement can be used to define a created function of variables in the data base. A LET statement may appear anywhere in the message, provided that the variable that it creates has not been referred to earlier in the message.

Blanks are allowed as they are in English: A blank may not be placed in the middle of a word, but as many blanks as the user wishes may be placed between words. Any special symbol, such as

;+-*'/,:();

may be surrounded by as many blanks as desired.

The order of the statements—other than the LET and GO statements—is arbitrary. For example, the user may type the IN statement before the TOTAL statement, or vice versa.

Any statement may be overridden by merely retyping it. Thus, the user can change his mind about the content of any portion of the message, provided he has not typed GO.

If he has typed GO and there are errors in his message, appropriate diagnostic comments are printed. He need then only retype the erroneous statements in the message, followed by the GO statement.

Running the program

Once the program is loaded, it communicates its sole signal for user information requests by typing "?:=" after which the user starts typing. The user must be careful not to end a line in the middle of a word or number. Unlike English, DATAPLUS does not allow hyphenation. (However, we have allowed the typesetters to hyphenate DATAPLUS messages in the printing of this paper.)

If a message has syntactic or semantic errors, or if not enough user information is supplied, appropriate diagnostics are typed, and the system again returns with its perennial ?:= symbol.

Once a valid message is given, the system searches the data, types out the processed results, again types ?:= and the cycle begins anew. Thus, a request for a distribution may be followed with a request for extracting data—without reloading the program.

When the cycle begins anew, the program assumes that a new message will follow. This assumption can be suppressed by typing "EDIT": followed by those portions of the message that the user wishes to edit. Thus, after the computer had printed its results for the message in Example 3, the user could continue with...

EDIT: WHEN COUNTY=DU PAGE: GO:
and have the computer supply the same results for Du Page County.

The ability to shift from one operation to another combines with the EDIT feature to provide an extremely useful capability of DATAPLUS—the capability for browsing and "hunch pursuit." Upon examining the computer's reply to one message, the user is often stimulated to formulate another message. Experience with the system has demonstrated that a user frequently finds himself as part of this "feedback loop"—coming away from the system with information much more valuable to him than the information he originally intended to seek.

The vocabulary

The basic vocabulary has two portions—the key words, and the nouns. We have already been introduced to most of the key words. This section is concerned with nouns, which include the state and city names and the names of variables such as those given in Table I.

Although a variable appears on a given level in the data base, it can be used on a higher level in a message. An example of this "raising the level" of a variable is the noun DOLLAR SALES which appears as a DEPARTMENT level variable in Table 1, and was used as a STORE level variable in Example 2. In that example the variable of interest was the DOLLAR SALES PER STORE, viz, the dollar sales summed over all departments in the store.
In DATAPLUS, the level of any range variable may be raised. This is accomplished by following the variable name (as it appears in Table I) by the word PER and then by the word DEPARTMENT or STORE.

The variable DEPARTMENTS is a range variable which has the value unity for every department, and the variable ITEMS is a unity-valued range variable for every item in the data base. Since these are both range variables, their levels may be raised. Thus, to obtain a picture of the number of different types of items sold in stores in Dallas, one might say . . .

DISTRIBUTE STORES: BY ITEMS PER STORE: BETWEEN 0 and 700 IN STEPS OF 7: IN TEXAS, DALLAS: GO:

This will produce a distribution with stores as the Y axis variable, items per store on the X axis, with 100 class intervals each of size 7 items. In this example, the level was raised from item level to store level—i.e., two levels.

If the user tries to “lower the level” of a variable, he will get a diagnostic message. Clearly, the level cannot be lowered since this is a piece of information unobtainable from the data base. It is perfectly valid to follow the word PER by the name of the level on which the variable was collected. In fact, if the user is unsure about the level, he can always play safe by following the variable name with PER and then the level.

The vocabulary can be obtained from the system at run time. If the user types an invalid state name, the computer will type out a list of valid state names. If the user types a valid state name but an invalid city name, the computer will type a list of valid city names for that state. A listing of valid variable names may be obtained by putting the statement “CATALOG”: anywhere in the message. Furthermore, the user can obtain definitions of any of the variables by using the DEFINE statement. For example, the statement

DEFINE IN STOCK, ON ORDER:

may be placed anywhere in the message, and the computer will furnish concise definitions of the variables IN STOCK and ON ORDER.

Creating variables

DATAPLUS has provision for extending its vocabulary, that is, for adding to its nouns. This is accomplished by the LET statement, which we mentioned briefly in Section III.

A created variable in DATAPLUS is always a range variable. In the previous section we saw how to create certain variables that are not in the data base. Those “raised level” variables (since they had to be summed) were range variables. In Part A of the present section we describe how to create algebraic functions of variables. Since we are performing algebraic operations, the operands, as well as the function, must be range variables. Part B of the present section shows how to create special variables, called qualified variables.

A. Algebraic functions

An algebraic function is formed by algebraic manipulations on range variables appearing in the data base, on raised level variables, or on variables created in earlier LET statements.

Some examples are . . .

(a) LET PERCENT = (SELLING PRICE-PURCHASE PRICE)/SELLING PRICE:

(b) LET RATIO = EARNINGS/DOLLAR SALES PER STORE:

(c) LET CASHFLOW = EARNINGS + DEPRECIATION:

The algebraic operators are +, −, *, /. Exponentiation is not allowed. The hierarchy of operations is the same as in FORTRAN. Redundant parentheses are ignored. (Unary operations are not allowed; thus it is not valid to say LET X = EARNINGS:)

The level of all variables appearing on the right hand side of the equal sign must be the same. (The construction would be semantic nonsense otherwise.) The created variable is assigned this level. Once a variable is defined in a LET statement, it may be used in any subsequent statement where a range variable of that level is permitted, including another LET statement.

Suppose we wish to obtain a distribution, for stores throughout the United States, of cash flow versus earnings. The message could be . . .

LET CASHFLOW = EARNINGS + DEPRECIATION: DISTRIBUTE CASHFLOW: BY EARNINGS: BETWEEN 0 AND 500000 IN STEPS OF 50000: IN COMPANY: GO:

B. Qualified variables

Qualification is accomplished in DATAPLUS by enclosure in parentheses. (This construction is a familiar one in English.) Some examples are:

(a) LET X = IN STOCK(INVENTORY NUMBER = A0578):

(b) LET Y = DOLLAR SALES (DEPARTMENT NAME = HABERDASHERY):

(c) LET Z = ON ORDER (INVENTORY NUMBER = B6724):

LET T = Z PER STORE:
The variable X is an item level variable. Its value is the number of units in stock that have an inventory number of A0578. The variable Y is a department level variable. Its value is the dollar sales for the department if the department is a haberdashery department and zero otherwise. The variable Z is an item level variable. The variable T is a store level variable, and represents the number of units on order in the store which have inventory number B6724.

In general, the value of the created variable is the value of the qualified variable if the equality enclosed in parentheses is true, and zero otherwise. The level of the created variable is assigned the level of the qualified variable. Once a variable is created by qualification, it may be used in any subsequent statement where a variable of that level is permitted, including another LET statement.

Suppose we wish to determine, for Chicago, all stores which have more than 30 TV sets in stock. The message could be...

LET TVSUPPLY = IN STOCK(ITEM NAME = TV SET): EXTRACT STORE NAME, ADDRESS: FROM STORES: IN ILLINOIS, CHICAGO: WHEN TVSUPPLY PER STORE=30 to 9999: GO:

**Boolean operations**

Most information retrieval systems allow for Boolean operations on index terms. DATAPLUS allows for Boolean conjunction and (inclusive) disjunction by use of the words "AND" and "OR" in the WHEN statement.

For example, suppose we wish to find the total number of furniture departments in stores in Denver that have a sales force of at most 14 people and dollar sales of more than $100,000. The message

TOTAL DEPARTMENTS: IN COLORADO, DENVER: WHEN DEPARTMENT NAME = FURNITURE, AND SALES FORCE=1 TO 14, AND DOLLAR SALES=100000 TO 9999999: GO:

If we are interested in finding the number of departments whose name is either "furniture" or "childrens furniture," the message could be...

TOTAL DEPARTMENTS: IN COLORADO, DENVER: WHEN DEPARTMENT NAME = FURNITURE, OR DEPARTMENT NAME = CHILDRENS FURNITURE: GO:

Intersection and union may be used in the same WHEN statement, with the AND taking precedence over the OR. Thus the message...

TOTAL DEPARTMENTS: IN COLORADO, DENVER: WHEN DEPARTMENT NAME = FURNITURE AND SALES FORCE = 1 TO 14, OR DEPARTMENT NAME = CHILDRENS FURNITURE, AND SALES FORCE = 1 TO 14: GO:

will total those departments whose name is furniture and which have a sales force of at most 14 people, or whose name is childrens furniture and which have a sales force of at most 14 people.

It is also possible to use the qualified variable as a means for specifying conjunction. Thus, the messages...

TOTAL STORES: IN TENNESSEE: WHEN COUNTY=KNOX, AND EARNINGS=0 TO 70000: GO:

and

LET X=STORES (COUNTY=KNOX): TOTAL X: IN TENNESSEE: WHEN EARNINGS=0 TO 70000: GO:

are equivalent. Both will determine the number of stores in Knox county, Tennessee, which earn less than $70,000.

Two operations that prove extremely useful in retrieving from hierarchical data bases are the ANY and ALL operations. These can be used for variables mentioned in the WHEN statement which are on a lower level than the level specified by the TOTAL, DISTRIBUTE, or EXTRACT functions. (The level specified in the EXTRACT function is given in the FROM statement.) We have already seen a use of the ANY operation in the example...

TOTAL STORES: IN MICHIGAN; NEW JERSEY: WHEN ANY DEPARTMENT NAME=HI FI: GO:

where the variable DEPARTMENT NAME is on a lower level than the variable STORES specified by the TOTAL statement. As an example of the ALL operation, suppose we wish to find those departments in New Jersey, all of whose items sell for $8.00 or more. The message could be...

EXTRACT STORE NAME, ADDRESS, DEPARTMENT NAME: FROM DEPARTMENTS: IN NEW JERSEY: WHEN ALL SELLING PRICE PER ITEM=8.00 TO 9999999: GO:

In this case, the variable SELLING PRICE is on a lower level than the variable DEPARTMENTS.
Although DATAPLUS does not presently support the Boolean negation operation—due primarily to core size limitations—it is often possible to perform negations by appropriate use of qualified variables. Suppose we wish to find the number of stores in Illinois that are not in Cook County, we could say...

LET X = STORES (COUNTY = COOK): LET Y = STORES (X = 0): TOTAL Y: IN ILLINOIS: GO:

The variable X will have the value 1 if the store is in Cook County, and 0 otherwise. The variable Y will have the value 1 if and only if X=0, that is, if the store is not in Cook County.

Application to other data bases

Although DATAPLUS was designed specifically to process data files useful to telephone engineers, the language is capable of wider application. The data file described in this paper can use a simple cognate of the language. Specifically, this means that only the nouns need to be changed and this can be accomplished by a trivial modification of a few program tables.

Furthermore, the hierarchical structure of the department store chain is representative of a tree structure common to many data bases. Suitably extended, the techniques employed in DATAPLUS can be applied to other tree-structured data bases, with the complexity of the extension depending on the number of levels in the hierarchy.

Although this paper has been primarily concerned with the DATAPLUS language, it should be clearly understood that the language is the “front end” of an information processing system, the “back end” of which performs such actions as fetching the data from disk to core, calculating distributions, and printing results. Therefore, if one were to use DATAPLUS as the interrogation language for an information system built upon another data file, it would be profitable to borrow as much of the back end of the present system as possible. A large measure of borrowing can in fact be accomplished because of the system’s modular design.

There is one subroutine (the Read routine) whose sole function is to fetch data from disk. The other programs are independent of the actual physical layout of data on the disk. Hence, application to other data bases could be performed by appropriately changing the Read routine with minor revisions of the other subroutines.

The fact that the data layout is not an inherent part of the total system suggests three immediate advantages: The data in any application can be structured on disk to take advantage of any idiosyncrasies of the particular data base. The data can be structured to take advantage of the particular computer system’s executive program and scheduling policy. The relative efficiencies of different layouts can be experimentally tested by inserting different Read routines.

CONCLUSION

In the literature on query languages comparatively little consideration has been given to the manipulation of data at the various levels of the multilevel file.1-8 The necessity of such hierarchical operations as the ANY, ALL, and PER operations in DATAPLUS has been recognized in a recent paper9 describing a proposed general purpose data management system. Such operations, combined with the DATAPLUS capabilities of totalling, distributing, and creating algebraic functions, make it possible to more fully utilize the information potential of the data file, and consequently increase the performance/cost ratio of the information system.

DATAPLUS was implemented on the GE-265 computer operating under the GE time-sharing monitor, which allows only 5000 words of user core available for compiled FORTRAN code. (A number of program overlays were obviously required.) The system described in this paper has thus demonstrated the feasibility of implementing—on a small machine—a highly accessible, relatively inexpensive, and easy to use information retrieval system with substantial processing capability.

REFERENCES

1 C W BACHMAN S B WILLIAMS
   The integrated data store—A general purpose programming system for random access memories
   AFIPS Conf Proc 1964 pp 411-422
2 Introduction to integrated data store
   General Electric Computer Department CPB 1048 April 1965
3 J H BRYANT P SEMPLE JR
   GIS and file management
   Proc of ACM September 1966 pp 97-107
4 Generalized information system
   IBM Manual E20-0179 Reference No A 01 8773
5 W D CLIMENSON
   Annual review of information science and technology
   John Wiley & Sons
   New York 1966 Volume I Chapter 5
   File organization and search techniques
6 J MINKER J SABLE
   Annual review of information science and technology
   John Wiley & Sons
   New York 1967 Volume II Chapter 5
   File organization, maintenance and search of machine files
7 C T MEADOW
*The analysis of information systems*
John Wiley & Sons
New York 1967 Chapter 2
*The languages of information retrieval*

8 N S PRYWES

9 R E BLEIER
*Man–computer problem solving with multilist*
*Proc of IEEE Volume 54 No 12 December 1966*
*Treating hierarchical data structures in the SDC time-shared data management system (TDMS)*
*Proc of ACM August 1967 pp 41-49*