Advantages of Scientific Relational Data Bases

Thomas Cerwonka

Harbor Branch Foundation, Inc.
Ft. Pierce, Fl 33450

Harbor Branch Foundation, Inc., a not-for-profit research organization located in Ft. Pierce, Florida has been developing scientific relational data bases for several years. The relational architecture creates a flexible, open-ended data structure which can be easily modified and expanded as required. Rapid system prototyping and development are features of relational data base structures which have proven to be quite useful. INFO, a fourth generation language, has been used successfully to develop several major relational data base management systems at Harbor Branch. This paper will discuss relational data base concepts and design considerations involved in developing a scientific relational data base system. Examples of data structures, and coding techniques will also be presented.

INTRODUCTION

The Harbor Branch computing resource is comprised of a PRIME-750 super minicomputer with eight megabytes of main memory and 960 megabytes of online data storage. An Ungermann-Bass ethernet-based Local Area Network (LAN) with a fiber optic link provides communication between eight outlying buildings on the 450 acre Harbor Branch complex and the PRIME computer system. Over eighty terminals, seventeen printers and three plotters are attached to the PRIME-750 for use by Harbor Branch personnel.

Researchers at Harbor Branch have access to a wide array of software packages. One of these software packages is a fourth generation relational data base management language called INFO. Originally developed for business applications, INFO has been used successfully for scientific applications over the past several years. Its relational architecture allows for rapid system prototyping and development. The relational structure is open-ended and can be easily modified as research criteria change and expand.

Most fourth generation languages have similar features such as relational file manipulation, report generating, input/update screen formatting and programming capabilites; these features help to decrease applications software development time. Because of the flexible nature of the languages, program code can be reused for different applications. Fourth generation languages can be divided into two major categories:

1) Ad-hoc query languages
2) Code generating compilers

The compilers are oriented towards a professional programming audience. This type of language is more efficent and provides faster execution speed and processing, however to achieve this performance, compilers are more difficult to use. Query languages on the other hand, are end-user oriented and appeal to the non-programmer.

INFO, marketed by Henco Software Inc. of Waltham Massachusetts, is a programmable query language which is easy to learn and use. Prior to acquiring INFO, all programming applications were written in FORTRAN by the Computer Services department. Due to the nature of FORTRAN, the data bases which were developed used a hierarchical data structure. Hierarchical data sets are comprised of a rigid format with all data contained in one file. This type of data base is often hard to modify, historically creating a large backlog of change requests. Incorporating INFO into the computer system has relieved much of this backlog. Not only does Computer Services produce application software faster and more efficiently, but many of the users which were previously dependant upon the computer

CH2269-9/86/0000/0191$01.00 ©1986 IEEE

department are now developing and using their own data base management systems.

HIERARCHICAL VS THE RELATIONAL DATA STRUCTURE

The hierarchical data structure is one in which all data resides in a singular or primary file. Generally, a specific field within each data record is designated the record type indicator. By altering the value in the indicator field, a program is then able to differentiate between the type of the data contained in each record. In the example which follows, the indicating field is called the category number. Each type of data such as station, cruise or chemical contains a specific and unique category number.

The relational data structure on the other hand is one in which the data is broken into separate, yet related files. This relationship between files is established by means of a unqiue data item known as a key. This key value is identical in every data file in the data base, so that the separate files can be related together by this item. However, within each individual file, the key item value must appear only once.

Converting from hierarchical to a relational data structure

Because the relational database architecture is different from any data structure used previously, a new way of organizing data had to be developed. To illustrate the differences between the hierarchical and relational structures, an example of a converted data base follows. The example is a study of the chemical makeup of the Indian River Lagoon. The data was collected by Harbor Branch Foundation at 529 stations located along the Indian River which is located on the east coast of Florida. A total of 26 different chemical variables were monitored along with physical data such as temperature, depth and salinity.

Originally this data was structured as one large file, the data was keyed with a serial number in the first eight characters of each record. Chemical data records were assigned serial numbers on the basis of station number and date. Therefore, all parameters measured at a particular location on a given date had identical serial numbers. Serial numbers were assigned in numerical sequence as data was received by the data processing center. Other than providing unity to a set of data, the serial number has no significance.

Following the serial number is the category number which is used to indicate the type of data contained in that particular record. For example, station records contain category number 10, weather data records are assigned category number 12, and so on for every type of data record in the file. The next two characters of each data line represent the replicate number. This number is usually 01 but if a parameter was sampled more than once then the information would be recorded on two lines, having replicate numbers 01 and 02. Figure 1 contains an explanation of the data format for each record type and a sample of each type of record from the orginal hierarchical data file.

This data set was collected over a period of several years and when completed contained in excess of 300,000 records. Because of the single file hierarchical structure, it was an extremely slow process to extract a subset of the data. The data file was normally sorted by serial number and to sort by any other field, such as category number, was a very time consuming function. The reports, which were generated by a FORTRAN program, contained all possible fields and were difficult to interpret and alter. If new report formats were desired by the scientists, a programmer had to be assigned to produce it. In addition, the format of the file is quite rigid; to add a field to an existing data line meant that both the programs and the data file had to be modified. Once this data file was converted to INFO, these problems were, for the most part, eliminated.

Relational Data Base Structure

INFO is a template based system, as are most fourth generation languages; this template concept makes these languages extremely flexible. The template is the format or layout of a data record for the particular file. The length and characteristics of each data item in the record are defined in the template. Because the data format is defined by a template and is not imbedded in programs as source code, the format of a data set can be altered without having to change any programs. In addition each data item is given a name. Once named it can be referred to by that name rather than it's location within the file. This naming cability is the basis of the English like command syntax and interactive query ability of fourth generation lanuages.

An additional advantage to naming data items is that individual fields can be grouped together or redefined. For the Indian River Lagoon study it is neccessary to redefine the Serial number and the Category number fields into one field. A new data item called SERIAL was created which spanned columns 1 through 12. This combination was neccessary to provide the unqiue key for each type of file to be created in the new data base.

To convert the original file, it was transferred into INFO and broken down into seperate files according to category number. A general INFO template was defined which would accept all data record types. Then, by using a command called RESELECT, a subset of the data could be generated according to category number. The reselections were repeated for each category number in the original file. Once the process was completed, an INFO file existed for each data type station, weather, cruise, etc.

The following example of INFO templates and code is obviously language dependant, but while the specific syntax may differ, most relational languages possess similar capabilities. Comments

have been added along with the code and formats for clarity.

This record is an example of the original station data format:

```
00064000 01001 CP00302Z 240676
```

The following template format was used to bring the original data set into INFO

```
DATAFILE NAME: ORIGINAL

   7 ITEMS: STARTING IN POSITION    1
 COL  ITEM NAME          WDTH OPUT TYP N.DEC
   1  CR-DATE              5    5  I    -
   6  SP-DATA              3    3  I    -
   9  BLANK                1    1  C    -
  10  CAT-#                3    3  I    -
  13  REP#                 2    2  I    -
  15  BLANK-2              1    1  C    -
  16  OTHER-DATA          63   63  C    -
**  REDEFINED ITEMS  **
   1  SERIAL              12   12  C  -
```

Once the original data set is converted to INFO, it is accessed using the SELECT command. When a file is selected it is activated and all records in the file are made available for use. To create the station data file, the available records had to be narrowed to include just those records containing station data. To narrow an INFO data set, it must be RESELECTED for a specific data value. In this case, the data set will be reselected for all records with a category number of 10. The following INFO commands will accomplish the select and reselect processes.

```
>SELECT TEMPLATE            Access the master file
 300,000 RECORDS SELECTED   INFO displays records
                              available

>RESELECT FOR CAT-# = 10    Obtain only station data
 50,000 RECORDS SELECTED      records
```

Once the data set has been reselected for the proper range of records, a new file of just station data can be created. This new file is created by first defining a data template with the proper record format. The new template can be RELATED to the selected file by a common data item or key. Once the relationship between the selected and related files is accomplished INFO will do all the file handling neccessary to match the two files according to the key item value. Since new records are being created in the related file an option called INIT is used in conjunction with the relate command. The INIT command will instruct INFO to create one new record for each unique key value in the selected file. The following template matches the format of the station data record in the original file.

```
DATAFILE NAME: STATION-DATA

  10 ITEMS: STARTING IN POSITION    1
 COL  ITEM NAME          WDTH OPUT TYP N.DEC
ALTERNATE NAME
   1  SERIAL-#             8    8  C    -
   9  BLANK-1              1    1  C    -
  10  CAT-#                3    3  I    -
  13  REP-#                2    2  I    -
  15  BLANK-2              1    1  C    -
  16  STATION-#            8    8  C    -
  24  BLANK-3              1    1  C    -
  25  YEAR                 2    2  I    -
  27  MONTH                2    2  I    -
  29  DAY                  2    2  I    -
    **  REDEFINED ITEMS  **
   1  SERIAL              12   12  C    -
  25  DATE                 6    6  C    -
```

Now create one record in the STATION-DATA file for each unique record in the master file.

```
>RELATE STATION-DATA BY SERIAL WITH INIT
```

Once the relate command has executed, the STATION-DATA file is built and contains 50,000 records. This same procedure was followed for each of the other types of files in the data base. Seven files were created in the conversion process:

STATION-DATA	Category number	10
WEATHER-DATA	Category number	12
CRUISE-DATA	Category number	20
POSITION-DATA	Category number	40
TIME-DATA	Category number	60
PHYSICAL-DATA	Category number	900
NUTRIENT-DATA	Category number	920

With this new structure, data is accessed only as required, that is when a particular type of data is needed. Only the file containing the required data type is selected for use. Other files can be related to it by means of common items or keys. It should be noted that for the relational data structure to be effective, each data file must contain a field with a key data item which makes that record unique. In these files the field SERIAL-# is unique; however, this is an artifical key which was created specifically for the purpose of record identification.

Once created, these files can be accessed using program logic if repetitive processes such as calculations or data entry are to be accomplished. In addition, because of the query capabilites of the language, the end user can interactively manipulate data and if desired create specialized reports. Since file handling is done automatically by the language, no programming skills are needed to access the data. The English like syntax of 4th generation languages helps to make the data more directly accessable to the end user.

An example of how the relational structure can be used to improve data reporting features can be illustrated with the previously defined CRUISE-DATA. This file contains a record for each sampling cruise done for this study. Contained within each data record is a 1 character code known

as SHIP-ID which indicates the research vessel used for that particular cruise. Originally scientists had to cross-reference an accompanying table to determine which ship was used for a particular cruise.

Using this 1 character field as a key, a new file was created with the same key plus a ship name field. Now when a report of cruise data is required the CRUISE-DATA file is selected and the SHIP-NAME file related to it by the item SHIP-ID. Once these two files were created, the reports generated were much easier to interpret. The report generation process can be coded into a program and run from a menu; or with a few English like commands, the report can be executed interactively. The following is an example of raw data contained in each file, the command syntax and a report format which can be generated using this technique.

CRUISE-DATA sample data

```
00001000 020 1 G0016  4
00002000 020 1 D0016  4
00003000 020 1 H0016  4
00004000 020 1 F0016  4
00005000 020 1 H0016  4
```

SHIP-NAME sample data

```
DSea Diver
FBlue Fox
GGosnold
HHouseboat
```

```
SELECT CRUISE-DATA
RELATE SHIP-NAME BY SHIP-ID
REPORT CRUISE-INFO
```

Serial Number	Category Number	Replicate Number	Ship Name	Chemical Number
1000	020	1	R/V Gosnold	4
2000	020	1	R/V Sea Diver	4
3000	020	1	R/V Houseboat	4
4000	020	1	R/V Blue Fox	4
5000	020	1	R/V Houseboat	4

Data Normalization and Disk Storage Optimization

This process of creating sub-files of data and using key fields to tie the separate files together is known as Data Normalization. Data is said to be normalized when fields of redundant data are removed from the primary files and placed in secondary or sub-files. Significant disk storage is conserved by using this normalization technique since the file with the greatest number of records only contains a small key field. For the above example, nineteen characters of disk storage were conserved for each data record, because the SHIP-ID field is defined to be 20 characters in length and was replaced by a 1 character key field. With approximately 500 records in the CRUISE-DATA file, a savings of about 9500 characters was affected. While Data Normalization is possible using any computer language, the ease of implementation using relational architecture is dramatic.

Natural vs Artifical Data Keys

The Indian River Lagoon study demonstrates how relational data base techniques can be used to improve data accessability. The primary limitation, however, is the use of the sequential serial number to key the data. This arbitrary number occupies the first eight characters in every data record in the data base. Primarily this key is finite in nature and should this study continue, eventually all possible key values will be used.

Having developed several scientific relational data bases, the Computer Services department at Harbor Branch Foundation has developed a data keying technique which is applicable to most studies. It is called a natural keying system since the actual data collected is used to key the records rather than inventing an arbitrary key field. Generally scientific data is collected over time from selected locations and this collection process is a repetitive one. The best way to key scientific data is by combining the physical and temporal collection information in the proper sequence.

A relational data base which demonstrates this data structure is the taxonomic collection system developed at Harbor Branch Foundation to record species information for organisims collected at sea. This data base system makes use of an IBM-PC version of INFO, which is used on board ship for data entry of collected organism information. The INFO programs on both the PRIME 750 and the IBM-PC are identical, giving the scientists the advantage of using the same software on both systems.

Each organism collected is assigned an organism number, this number is actually a combination of three distinct fields; the date of collection, the number of the dive on which it was collected and a sequential sample number. In the template definition, the organism number is called DATE-DIVE-SAMPLE, and is located at the beginning of every record in the data base. The combination of these three fields creates a unique key which identifies that particular organism for all further studies conducted on it. Use of this type of keying technique provides an inexhaustible list of values to draw upon for identifing organisms.

Once an organism has been collected by Harbor Branch scientists it is entered into the Cruise datafile using the onboard data entry system. Upon returning to Harbor Branch Foundation, the cruise data is transferred to the PRIME-750 super mini-computer and merged with cruise data in the existing data base. All supplemental information or testing done to an organism is identified by the organism number; for example, the freezer location of where the actual organism is stored is contained in the data base. The freezer location file is also keyed by the organism number, DATE-DIVE-SAMPLE.

The relational data base structure has been very effective in organizing data of this type. Since these organisms are examined by different teams of chemists and biologists, each team is searching for particular traits or activities. As new tests are conducted and as research progresses, new data files can easily be added to the existing data base. Each research team enters data directly into the computer using INFO data entry routines.

The results of each test is stored in its own file in the data base system. Once again all data files begin with the unqiue DATE-DIVE-SAMPLE key value.

This natural keying technique has proven most useful when scientists are interested in examining the results of a particular organism. By entering the desired organism number, a comprehensive report of all tests to date can be generated in a matter of a minutes.

The template of the cruise data taxonomy file format and a few lines of actual data follow to demonstate the natural keying technique described above.

```
 DATAFILE NAME: RAW.DAT
   43 ITEMS: STARTING IN POSITION    1
 COL  ITEM NAME          WDTH OPUT TYP N.DEC
    1  DATE                8    8  C    -
    9  DIVE                1    1  C    -
   10  SAMPLE              3    3  C    -
   11  TAXONOMY           70   70  C    -
   **  REDEFINED ITEMS  **
    1  DATE-DIVE-SAMPLE   12   12  C    -

DATE-DIVE-SAMPLE      TAXONOMY

1985 810 1 1 White plate sponge.
1985 810 1 2 White round sponge.  Geodia?
1985 810 1 3 sponge, amorphous, white
1985 810 1 4 Small round black sponge.
1985 810 2 1 Yellow green tube worm.
1985 810 2 2 White cap sponge.
1985 810 2 3 Diaphanous white stalk. Farrea sultion
1985 810 2 4 Gorgonian (sea whip?).
1985 810 2 5 Yellow green tube worm.  MISSING.
```

CONCLUSION

Overall, the introduction of a relational DBMS language has been positive for Harbor Branch Foundation. INFO has given certain researchers the autonomy they required to develop their own specialized data base applications. The shorter development time has also benefited Computer Services, in that the department is better able to fill the scientists requests in a timely manner. In addition, these new languages are often integrated with other types of software, such as word processing, electronic spreadsheets, statistical analysis systems and graphics packages. This integration helps to simplify the process of data reduction and publication.

This paper is Harbor Branch Foundation Contribution Number 481.

REFERENCES

George A. Kerr, 1976. INDIAN RIVER COASTAL ZONE STUDY INVENTORY, Harbor Branch Consortium, Ft. Pierce, Fl.

INFO PRIME Reference Manual, 1985, Henco Software Inc., Waltham, Mass.

Figure 1.

Hierarchical data file format

Data type

```
Station data  Serial No., Category No., Replicate No., Station code, date
Weather data  Serial, Cat., Rep., Wind Speed&Direction, Weather, tide, rain
Cruise data   Serial, Cat., Rep., Shipo code, Cruise code, Chemical Number
Lat-Long data Serial, Cat., Rep., Latitude, Longitude
Time data     Serial, Cat., Rep., Collection Time
Physical data Serial, Cat., Rep., Temp, Depth, Diss. 02, Ph, Conductivity
Nutrient data Serial, Cat., Rep., Silicate, Nitrate, Phosphate, Ammonia
```

Sample data record

```
00064000   010   01  CP00302Z 240676
           012   01  04 SWE UN 05 X
           020   01  H0016 004
           040   01  2839.02 8049.40
           060   01  1408
           900   01  27.5 01.0 06.60 8.10 4.00E01 245
```