Rerum Naturalium Fragmenta

No. 350

Glasgow
1982

.

Rerum Naturalium Fragmenta
Tamas Jasko editor
34 Lomondside Ave, Clarkston, Glasgow G76 7UJ, Scotland

Salient Features of the Maxximum Database System

T. Jasko

INTRODUCTION

Looking for a commercially available package, B.N.O.C. acquired the MAXXIMUM database system to run the Exploration Database on its SEL computers.

The package was installed in November 1980. During the winter the package underwent extensive testing and a couple of experimental databases were created.

Software Development now concentrates on creating permanent and project oriented databases and providing interfaces to user program. The following short summary of the features of Maxximum reflects the actual status of interfaces. More detailed information is available in the MAXXIMUM manuals (8 volumes).

MAXXIMUM ™ is a complete database system for small to medium size computers. It was designed by California Software Products Inc. (CSPI) to operate in a real-time environment with efficient database organisation and fast access.

Retrieval of items from the database is by name so that the user is never required to be cognizant of the internal structure of the data. Up to 256 users can simultaneously access the database. A terminal interface driven by English-like commands can be

used for interactive retrieval and modification of data. Other interfaces can directly communicate with applications programs.

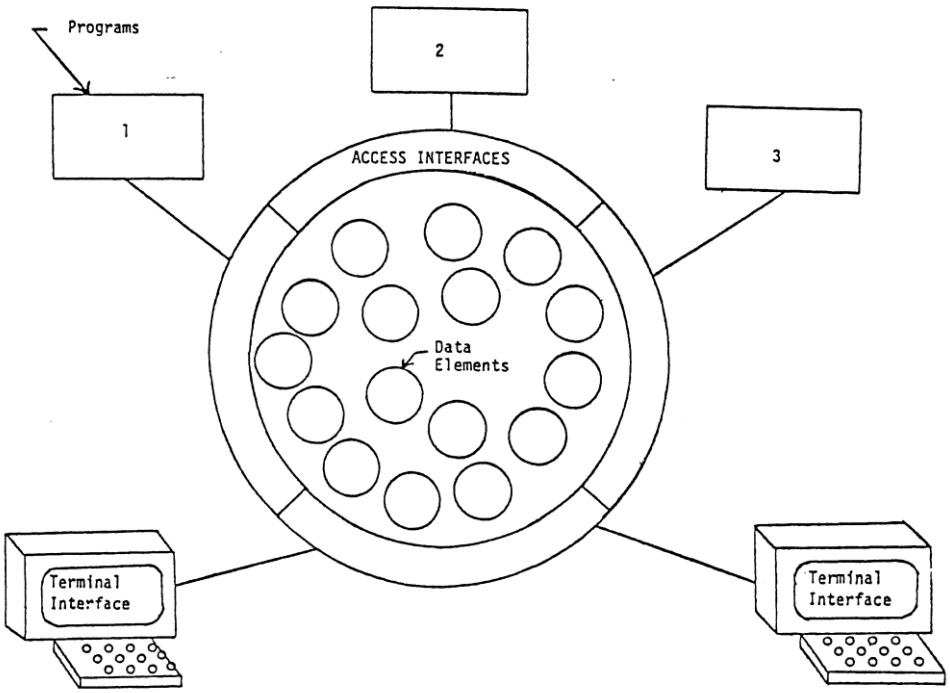

Figure 1
Data organisation in a DBMS

Database definition

The term database refers to a collection of data pertaining to a particular area of interest. In this sense, more than one database can exist in the system, e.g. one database containing information about wells, another on telephone extensions, still another data of ongoing projects.

The term item is used to refer the smallest identifiable piece of data in a database. For example, the extension number of Steve Smith is an item in the database.

The term repeating group is used to refer to a set of data items related to each other. For example, items relating to a particular well such as well number, latitude, longitude, spud date, total depth are collected in the repeating group called WELL.

Databases and the items contained in them are defined using the Data Definition Language. The definition statements set the storage size required for the data, the names of each data item, their type and default format etc. Available data types include character, integer, real, bit, data types, as well as arrays and blocks of these data.

Access operations

All access interfaces permit the following operations:

a) Selection of data sets by user defined profiles. Data sets are specified for selection by forming simple English-like expressions such as

WHERE TOTAL-DEPTH GT 9000

b) Retrieval of data sets with fast access since storage location of any set is computed by the system.

c) Updating of data sets is performed efficiently so that any item updated is immediately available for retrieval i.e. by other users. Whenever a keyed item is updated, inserted files are automatically updated.

d) Insertion of new data sets into the existing hierarchical structure. Insertion sets up actual storage location as compared to potential storage allocation through data definition. When a

data set is inserted, the user can specify all item values immediately or leave some of the items null until later updating.

e) Deletion of data sets, with automatic deletion of all descendants of a deleted data set.

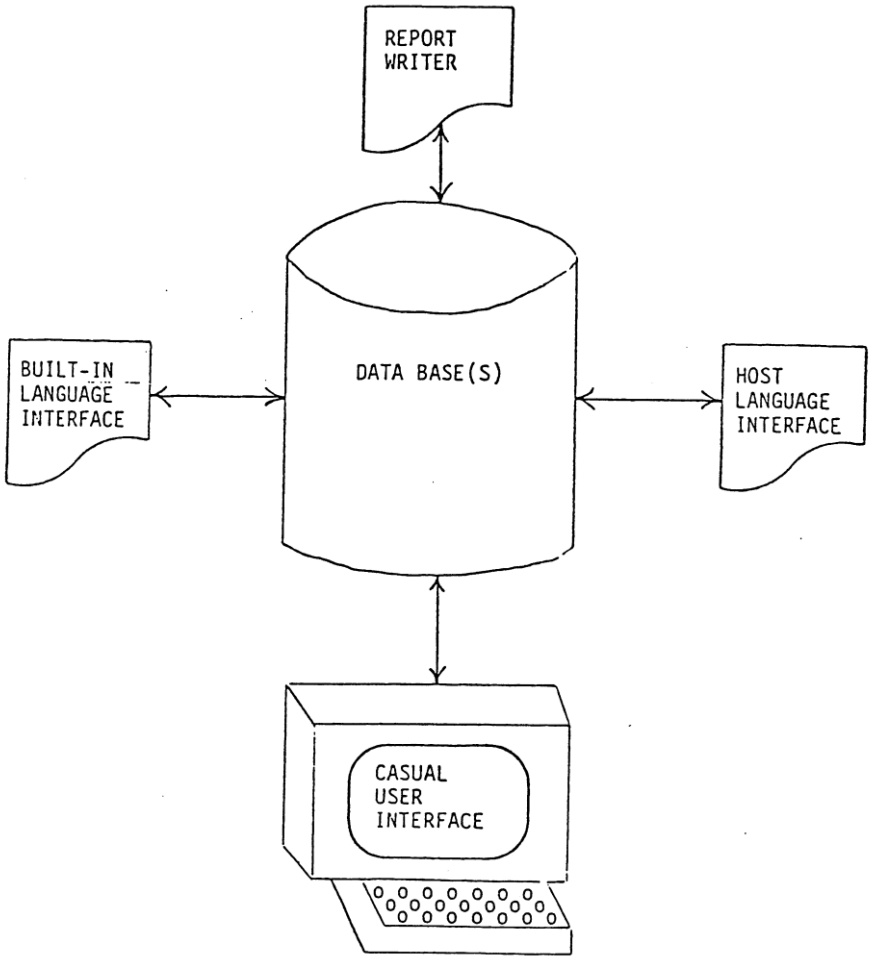

Figure 2
Maxximum database access interfaces,

f) Opening and closing of databases. A user can have up to 16 databases open simultaneously. Controlled opening and closing of databases protects against possible deadlock caused by

more than one user attempting to modify the same data. As databases can be opened at the repeating group level, the area locked by one user from updates by other users can be quite small and specific.

Access interfaces

Figure2 depicts the access interfaces provided by MAXXIMUM. There are 4 interfaces:

the Casual User Interface,
the Built-in Language Interface,
the Host Language Interface, and
the Report Writer.

Additionally, a Fortran Pre-compiler is available.

Casual User Interface

The Casual User Interface is an interactive, terminal oriented interface that accepts English-like commands to retrieve and/or update database items. This program is intended for one-off, interactively modified inquiries. Provided with valid pass-words to access the system the user can get any information in a database displayed on the terminal screen.

Alternatively, the output can go to a line printer producing a hard copy report.

The CUI responds to the following commands:
OPEN/CLOSE
DISPLAY/PRINT
CHANGE

INSERT
DELETE
DITTO
SAME
EQUATE
DESCRIBE
HELP

For convenience, any of the command verbs can be abbreviated as long as the stem typed remains unique e.g. C is OK for CHANGE, but DE is not for DESCRIBE (or DELETE?)

Built-In Language

As the name implies Built-In Language refers to enhancements built in the Pascal compiler. This compiler is taking in addition to standard Pascal source statements, special commands to manipulate the database. This way database access operations form an intrinsic part of the Built-in (or enhanced Pascal) language. With programs written in a high level language, this interface is intended for the rapid development of new applications programs that can be quickly debugged and tested and easily implemented and maintained.

Host Language Interface

This is a collection of Fortran-callable subroutines that enable the user to perform database operations. HLI routines are particularly useful for existing application programs that can be modified to access a database through HLI calls.

Report Writer

The Report Writer is mainly used to generate formatted reports from database items. Although the CUI can produce reports from the database, the user has little control over the actual output format. The Report Writer allows for the specification of page headings, column heading, column footings. Items printed in multi-column tables can be variously high-lighted.

Information can be centred on a page and triggers can control conditional page formatting.

Fortran Pre-Compiler

In addition to the 4 standard interfaces, Maxximum also includes a Fortran pre-compiler. The pre-compiler accepts a mixed input of Fortran and Built-in database commands to produce a Fortran source with HLI calls incorporated.

Database Administrator Utilities

DBAU are a collection of routines providing back-up and recovery from unintentional updates or hardware/software failures outside the control of Maxximum. Archive copies of existing databases can be created on a regular basis such as weekly or monthly.

Additionally, DBAU control the security of the databases. Passwords can be assigned to entire databases or individual repeating groups, specifying also the range of operations permitted i.e. retrieval only, update, delete or 'master'.

The utilities (which can be regarded as a 5th interface) can be invoked by commands entered from the operator console or a terminal.

(BNOC, Glasgow, 1981, 9p)

Lajos Telegdi Roth: in memoriam

Sándor Jaskó

Lajos TELEGDI ROTH (1841-1928) who died 50 years ago was chief geologist, chief mining counsellor, President and Honorary Member of the Hungarian Geological Society, bearer of several state decorations and medals.

He has drawn geological maps of several parts of Hungary. To be mentioned are his surveys in the Krassó-Szörény Ore Mountains and in the Transylvanian Ore Mountains.

He was editor of Földtani Közlöny, Bulletin of the Hungarian Geological Society. He participated in the compilation of the geological map of Hungary published in 1896. Travelled abroad many times and represented Hungary at several scientific congresses.

Lajos TELEGDI ROTH was an outstanding personality of Hungarian geology. His modesty, great diligence, unselfish love of nature and harmonic family life are for us examples to follow.

(Földt. Tudománytörténeti Évkönyv; 1979, p.105, 1981)

MEGAB: macro to list total size of a set of files

by T. Jasko

This macro reads a log file (in FILEMGR log format) and outputs the total number of files, and total size of files in blocks and megabytes.

Prepare a log file by running FILEMGR in batch or online (see example). When log file is ready, type

```
MEGAB logfilename
```

The program used, TJ.MEGAB will echo the list of files if Option 7 is specified. Input cards other than FILEMGR "LOG" cards are ignored.

Example: Size of files under username XDB with filenames starting 'D$' (The file XDBLOG is used as log file):

```
A1 SLO=XDBLOG
FILEMGR
LOG FILE=(XDB)D$??????
EXIT
MEGAB XDBLOG
```

(Software Update No.2; 11 August 1982)

The DARALO (Database Rapid Loader) system

by T. Jasko

1. PURPOSE

This program system is for the accelerated loading of card image data files to a Maxximum format database. Using a data description file which is an extension of the DDL, parameters are passed on to a background job (queued) which, upon completion, prints a summary of activities.

2. HOW TO RUN

Create a data description file using the editor, then store the file.
In TSM, type

```
QUMAC filename,user,key,project
```

(each parameter up to 8 characters)

The user may redefine the range of lines to be loaded from the data file and has to enter the password for updating the database.

3. JOB FLOW

DARALO Starts the load run as a queued background job. On successful completion a synoptic printout Is generated and returned to the user.

4. DATA DEFINITION

This is an extension of the DDL used to generate the database. The file gives general run parameters, describes input data fields and receiving database items. Actions of the System are controlled by verbs e.g. !FIL, !DBN etc. Verbs should be in the first columns of the data definition cards.

4.1 The !FIL verb

This verb gives the name of the input data file, the range of lines to be processed, the number of different pictures (formats) to be recognised, tag position and format tag characters.

Format:

columns	content
1-4	!FIL
6-13	filename (must be a stored(blocked) file)
15-18	Start line number
19-23	end line number
25-28	number of card types (pictures/formats)
30-33	column position of tag indicating type of card
35-	tags (1 character each, separated by a blank)

4.2 Picture mask cards

The !FIL verb card is always followed by one or more picture mask cards (one per each picture/format defined).
Format:

```
AAA BBBBB CG ... ZZ .. iii .. zz
```

i.e. the position of each field is indicated by a letter in the corresponding columns.

If e.g. columns 11..16 are masked by the letter P, it means that 'P' field is occupying columns 11..16 in the input file. The letters correspond to the definition of database items (See below).

4.3 The !DBN verb

This verb defines the name of the database.

Format:

```
!DBN database-name
```

The database name should be separated from the verb and subsequent fields (these are, in any case, ignored) by one or more blanks. This card should be the first parameter card.

4.4 Item definition cards: general format

The common format is

```
!NNN<alK> item-name picture
```

where
!NNN is the verb (KEY, INS, LET - see below);
al is the data field reference, meaning field 'a' of card type 1;
K is the key reference pointing to another field if the action is controlled by a where clause. This field should be defined by another card.
item-name is a database item name as in DDL.
picture ~ format (picture) as specified for DDL.

Apart from integer, real, character and date, two additional picture variants are recognised:

QCHARACTER is a Special case of character format, where strings are to be compressed by removing internal blanks.

LDATE incomplete date (as e.g. in Lexis files) of month and year only. Day is added as '01'.

4.5 The !KEY verb

For format, see 4.3. Use upper case letters only as field designator. Defines a field/item as a 'key only' item that is used in where clauses but generates no load action on its own.

N.B. Items/fields defined by the LET verb can also be used as locating keys.

4.6 The !INS verb

This verb defines a repeating group to be inserted every time the Card indicated by the card reference digit is processed. The first letter in the card reference determines the order of processing (HLI mode only), the second letter guides keyed/unkeyed insert.

4.61 Unqualified Insert

Key reference should be blank.

4.62 Current insert

The key should refer to fields in the same card designated by a letter in the range A,...,I as insert is issued after processing these fields and before processing the range J,...,Z. In HLI

mode, the permitted letter range for the key field is A cref, where cref is the first letter in the card card reference.

4.63 Delayed insert

Any keyed insert other than current insert will use the stored value of the last occurrence of the key field.

4.7 The !LET verb

This verb controls the assignment of values to database items inside a DO FOR EACH/DO INSERT loop. E.g.

```
!LET<B2 > WELL"NUMBER INTEGER ...
```

will result in assigning the value of field B (card type 2) to database item WELL"NUMBER in the currently loaded set and data format is INTEGER.

4.8 The !USERNAME verb

Indicates that the data file named in the !FIL parameter belongs to another user (not the one running the Job).

Format:

```
!USERNAME username,key
```

If present, this parameter card should precede the !FIL card, i.e. should be the second card.

4.9 The !END verb

This signals the end of parameter cards to be validated. Subsequent cards of the parameter file will be passed on without validation until the end of file or a card starting II is found. Can be used to input comments.

5. QUEUEING

Jobs are placed Into the LOADER queue. All are named DALnnn, where nnn = job number (automatically incremented). Each job is waiting for completion of the previous one. If none running, the queue can by started by operator action.

6. OPTIONS FOR THE BATCH LOAD RUN

DARALO background jobs can run in two modes, using either CUI or HLI logic.

6.1 CUI logic mode

Runs consist of preprocessing (TJ.POLOS), loading (BADRIVER) and post-processing. The full range of facilities is available but for multiple insert i.e. inserting the same dataset in more than one parent dataset.

The volume of data handled by a run is limited by buffer space to about 3000 data fields. Input files exceeding this limit should be loaded by more than one run - using the facility of line range redefinition.

E.g. if input file contains 1000 fields in 3900 card images, 4 runs can be queued loading cards 1-1000, 1001-2000, 2001-3000, 3001-4000, respectively. Line ranges are inclusive, and should be set not to cut logical records. End of file, if reached within specified range, terminates processing without error.

6.2 HLI logic mode

Runs use TJ.HALOM. This mode is much faster than CUI logic due to simplified buffering, validation and look-up procedures, but functions are limited.
Presently limited to level 0 repeating groups (INS, LET, KEY).

6.3 Setting run mode

Using QUMAC sets run mode to CUI logic (default), for HLI mode use the macro QUMAC2.

6.4 Compile only option

To validate parameter flies, use QUMACl.

This will read the parameter file, check all parameters, and produce job files for both HLI/CUI logic. No job will be queued, the job sequence number, however, will be incremented.
(DARALO : Software Update No.5: 22 December 1981)