Közlemények 29/1983 21-30

A QUERY SUBSYSTEM FOR A RELATIONAL DATABASE SYSTEM

LUCIANO GARCIA
MIGUEL KATRIB
EDUARDO QUESADA

Universidad de la Habana
Centro Nacional de Investigaciones Cientificas
CUBA

INTRODUCTION

The present article describes INTERROG, the query subsystem of LORKA, a relational database system (dbs) designed and implemented by the authors [3,4,5]. As it is - well known, relational models [1] accomplish in the most optimal way one of the more important constructive requirements in the design and implementation of dbs: independence of the logical structure of requests from the storage organization of data. It is considered that this independence should be achieved in the sense freeing the nonprogrammer user of all the machine--dependent thecnicalities for accessing data unnecessary for the logical structuralization of requests. Besides that, relational models provide as no other dos model an adequate framework for developing natural language interface and enhancing the dbs with a deductive capacity to deal with implicit information.

In the relational model the information of a database is arranged into *relations*, each one containing a collection of *tuples*. A relation can be thought of as a table *Fig. 1*, with columns named after the different attributes which compose the relation and rows of attribute values representing the tuples.

DOCUMENTS

TITLE	AUTHOR	DESCRIPTORS	REF-NO
T1	A1	(D1 D3 D4 D6)	RN1
T2	A2	(D3 D5 D6)	RN2
T3	A3	(D1 D5)	RN3
T4	A4	(D3 D4 D6)	RN4

BORROWERS

NAME	ADDR	CARD-NO
N1	ADDR1	0215
N2	ADDR2	0236
N3	ADDR3	0725
N4	ADDR4	0630

LOANS

REF-NO	CARD-NO	DATA
RN1	0236	1 2 83
RN3	0725	4 1 83
RN2	0215	11 2 83
RN4	0630	20 4 82

Fig. 1.

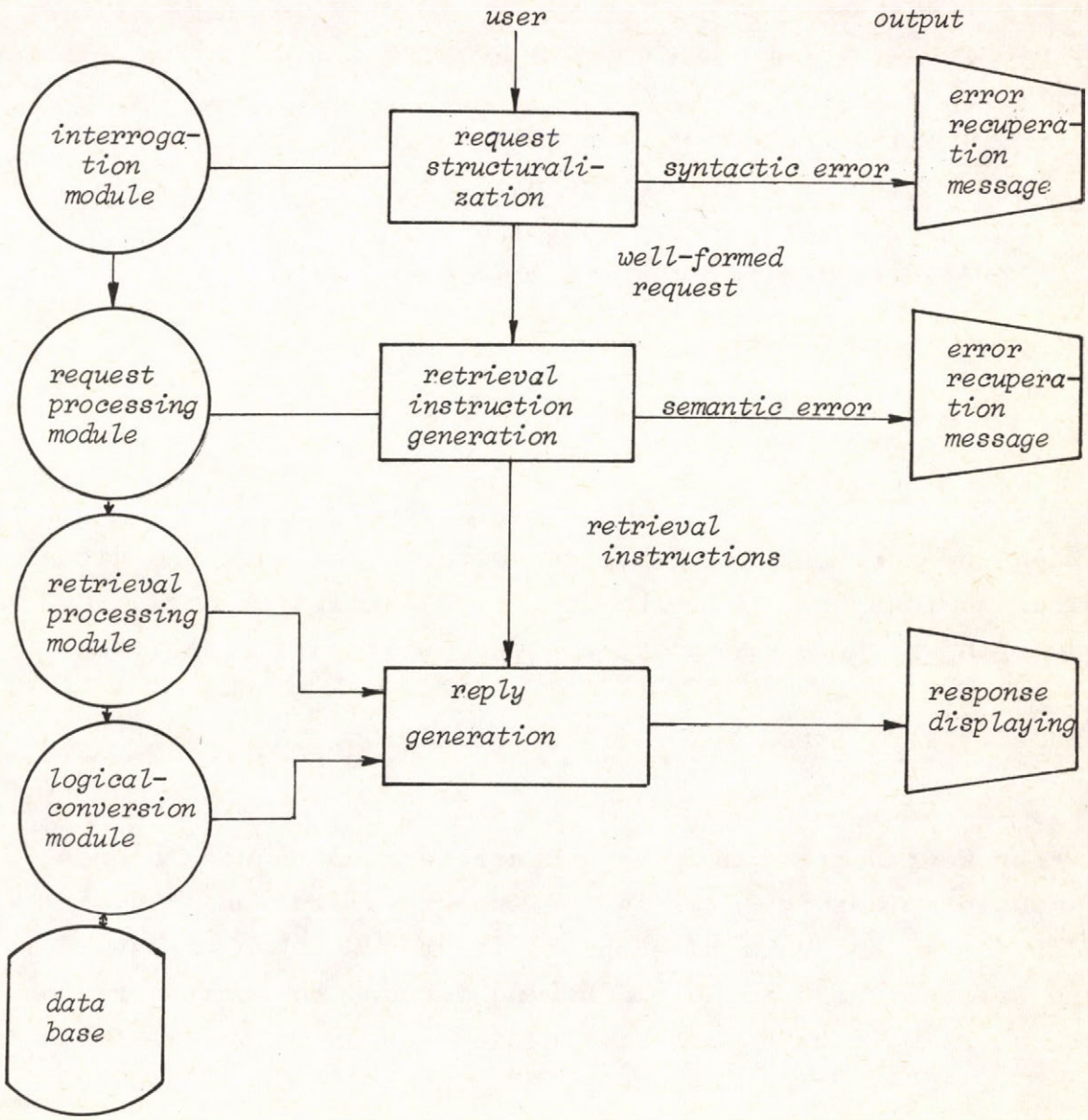
user
output
interroga-
tion
module
request
structurali-
zation
syntactic error
error
recupera-
tion
message
well-formed
request
request
processing
module
retrieval
instruction
generation
semantic error
error
recupera-
tion
message
retrieval
instructions
retrieval
processing
module
reply
generation
response
displaying
logical-
conversion
module
data
base

Fig. 2

The general design of INTERROG is depicted in *Fig. 2.* It is composed of four fundamental modules:

The request expressing module.
The request processing module.
The retrieval processing module.
The logical-conversion module.

which are going now to be explained in some length.

The request expressing module

When the user enters the system after a wellcome the visual terminal depicts the following (only the sublist of query commands in which we are going to concentrate are - depicted)

P(ROJECTION) S(ELECTION) B(OTH PROJECTION SELECTION) (1)

After keying the appropriated character the request expressing module guides the user in the structuralization of his request and at the same time checks it step by step for syntactic errors. For example; if having decided to enter a request which implies to perform a projection after a selection over two relations the user keys "B". The following lines are depicted one after another (user reponses to each one are include for purposes of illustration and are associated with the examples given in *Fig. 1.*):

Enter the atribute list

*(ADDR NAME? (2)

Enter the selection formula

```
*(= 1-2-83 DATE?                                    (3)
```

Enter the relation list

```
*(BORROWERS LOANS?                                  (4)
```

As can be appreciated the user constructs are LISP-like lists. The projection-list (2) is a list composed of an arbitrary numbers of atoms which denote atributes of the relations involved. The order in which these names appear in the projection list is arbitrarily determined by the users and reflects the order in which he wants the associated columns of values of the corresponding tuples to be displayed.

The formula-list (3) represents a formula of an applied first-order predicate calculus built with the propositional operators NOT, OR and AND, quantifier-free, with variables (attribute names) and constants (attribute values) as terms.

The elemental formulae are of the form (R T1 T2) where R is either an arithmetical relational operator (<,>,=,<>, <=,>=), either a set relational operator (MEMBER, SUBSET, DISJOINT) and T1 and T2 are terms. The formula is built in prefix notation and the operators AND and OR may have an arbitrary numbers of operands.

A relation-list (4) is composed of one or more atoms which denote the relations involved in the request. When there are more than one relation names in the list, INTERROG performs automatically the natural join operation whenever possible.

The character "?" at the end of any list is for terminating the list, and it is left in this way to the request expressing module to supply all the end parentheses needed.

If a syntactical error is committed through (2) - (4) the user receives an appropriate message and he has the option of recuperating it or of making a fresh start from (1).

The request processing module

When the request is well-formed it is delivered to the request processing module which first checks it for semantic errors. A semantic error is committed, for example, when a projection-list or a selection-formula contains an atom which is not an attribute name or an attribute value. The request processing module detects all kind of semantic - errors and sends appropriate messages to the user who is obliged to make a fresh start.

After passing successfully the semantical test, the request is finally accepted as input data by the request - processing module which proceeds to generate from it the retrieval instructions. This means that there are not *ad hoc built-in* retrieval instructions. As we have already explained the system has to cope with any projection-list or formula and the latter (see more examples further) can have a very complex structure. The request processing - module is provided with procedures which transform projection lists and formulae in retrieval instruction *lists* which in turn are to be applied as functions to the relations. The same thing occurs if the request implies the performance of a natural join of two relations: this can have an arbitrary number of attributes in common and the module generates the appropriate retrieval selection formula and projection list for the execution of the join.

The retrieval and conversion modules

The retrieval instruction lists are passed on the retrieval module where they are going to be applied as LISP functions to the appropriate relations. The retrieval process takes place in the computer working memory in a LISP-like manner. To accomplish that, the relations are brought to the working memory via the conversion module whose task is to transform the physical data structure of the relations in the suitable for processing LISP--like list structure in the working memory. In this way both retrieval instructions and relations are LISP-like listsand the process of aplying the firsts to the seconds is carried out through the LISP_EVAL function whose output is the response to the request.

It is in this way that the system succeeds in extablishing a complete independence of the logical reception and processing of requests from the physical organization of the dbs.

After being retrieved, the results are visually displayed with printing as an option. The displaying takes place inmediately to the evaluation of each tuples of the relations invclved. In our example if a tuple results from the natural join of the relations and the retrieval instruction formula evaluates it to true, it is immediately displayed and in case there is a projection list, only the row of values of the corresponding attributes is displayed. The answer is displayed with additional counting and average.

Additional features of INTERROG

The query language of INTERROG has the expressive power of the relationsl calculus [2]. Nevertheless the structuralization of a request by the user following the "menu" does not required the mathematics of the relational calculus. - The user has only to think in terms of attributes associated to logical formulae and relations. Thus, the request structuralization is non-proce-

dural and it is completely left to INTERROG to generate an *optimal* retrieval procedure: it is not the user but INTERROG the one which optimizes a Join taking advantage of an appropriate selection formula or of ordered-key relations.

The relation attributes have ranges which can take the following classes of types: scalar types like *nominal* (string), *literal* (finetely enumerative values), *integer*, *longinteger*, *real* and structured types like *date* (record of the form day--month-year) and *list* (dynamics sets of scalar and date types). It is possible to perform operation over date and list values. From a date we can single out the day, the month or the year:

```
(AND(=AUTHOR A3)(OR(=(YEAR DATE)83) (=(YEAR DATE)82?
```

with list values we can perform the already mentioned relational set-theoretical operations:

```
(AND(=AUTHOR A1)(SUBSET(D3 D4) DESCRIPTORS?

(OR(AND(=(YEAR DATE)82)(DISJOINT(D2 D1)DESCRIPTORS))

(AND(=(YEAR DATE)83)(SUBSET(D1 D5)DESCRIPTORS?
```

Others functions are included:

```
F(ILE
```

to create a PASCAL file from the request,

```
T(EMPORARY
```

to create a temporary relation from the request, and

```
M(AX        M(IN
```

to obtain the maximum minimum value in an attribute field of the tuples satisfaying the request.

CONCLUSIONS

INTERROG is the query subsystem of the first prototype of LORKA which was implemented in UCSD-PASCAL for cuban mini and microcomputers. This system are being used in Biomedicine in the Cuban National Center of Scientific Researchs. Further versions of LORKA already under development will enhance this prototype with natural language interface and deductive capacity.

REFERENCES

[1] Codd, E.G. "A relational model of data for large shared data banks". CACM V 13, N 6 1970 p. 377-387

[2] Codd, E.F. "Relational completeness of data base sublanguages". Data Base Systems Rustin R. editor. Englewood Cliffs, N.J. 1972 p. 65-98.

[3] Garcia, L. "Un sistema de interrogación de una base de datos fundado en la estructura de lista". 3er Evento Cientifico de la Universidad de la Habana, 1981.

[4] Garcia, L., Katrib, M., Quesada, E. "Realización del Sistema de interrogación del sistema de base de datos relacional LORKA". 1er Congreso Nacional de Matematica, Academia de Ciencias, Ciudad Habana, Cuba.

[5] Katrib, M. "Realización del lenguaje LISP en la configuración CID-300-10". 3er Evento Cientifico de la Universidad de la Habana, 1981.

A relációs adatbázis-rendszernek egy lekérdezési alrendszere

L. Garcia, M. Katrib, E. Quesada

Összefoglalás

A cikk ismerteti a szerzők által kifejlesztett LORKA elnevezésü relációs adatbázis-rendszernek lekérdezési alrendszerét.

ПОДСИСТЕМА ЗАПРОСА РЕЛЯЦИОННОЙ СИСТЕМЫ БАЗЫ ДАННЫХ

Л. Гарция, М. Катриб, Е. Квесада

Резюме

В статье описывается подсистема запроса реляционной системы базы данных LORKA разработанной авторами.