REMOVING RESTRICTIONS IN THE RELATIONAL DATA BASE MODEL: AN APPLICATION OF PROBLEM SOLVING TECHNIQUES

Laurent Siklóssy
Dept. of Information Eng.
University of Illinois at Chicago
Chicago, IL 60680 (USA)

Jean-Louis Laurière
Institut de Programmation
Université Paris VI
4, Place Jusieu
75230 Paris Cedex 05 (FRANCE)

ABSTRACT

The principal restrictions previously placed on the relational data base model have been removed in the L^2 *model* presented here.

We extend the model to include *null* (i.e. unknown and non-relevant) values (even in keys), repetitions of tuples, functional dependencies, a very rich set of constraints and information originating from several sources.

The programmed problem-solver ALICE is utilized to manipulate the constraints and to simplify relations: to answer a query, ALICE selects both the tuples which *might* answer the query upon appropriate substitutions for *unknown* values.

I INTRODUCTION

We assume that the reader is familiar with the relational data base model [Codd, 1975; Date, 1981; Ullman, 1980]. In practice, some values may not be known in a data base; they will be called unknown values.

In other cases, to avoid a multiplicity of similar data bases, non-relevant values are introduced. For example, the value of the attribute "Name of Spouse" is non-relevant for a person who is not married. Unless we permit such non-relevant values, we would be obliged to divide a data base on persons into two similar data bases: one for married persons: a second one for unmarried persons. Unknown and non-relevant values are combined in the term null values.

Data bases with null values have been studied by a variety of authors [Codd, 1979; Grant, 1977; Imielinski et al, 1981; Lipski, 1979, 1981; Siklóssy, 1981; Vassiliou, 1979, 1980], among others. Generally these authors have placed a variety of restrictions on the use of null values. Some authors do not permit non-relevant values. Others do not permit null values in a key.

We shall show that it is possible, with the help of the problem-solving program ALICE [Laurière, 1976] to lift the restrictions previously advocated. In addition, our extended model will admit functional dependencies and a rich set of constraints, since ALICE can manipulate these dependencies and constraints.

If queried, ALICE can indeed return both the tuples which satisfy the query, and also those tuples which might satisfy the query upon appropriate specifications of unknown values.

We shall first discuss the restrictions previously imposed upon null values, and will argue that their justification is tenuous at best. Then, through a set of examples, we shall describe some extensions of data bases with null values that ALICE can process.

2. ARE RESTRICTIONS ON NULL VALUES NECESSARY?

Two principal restrictions on null values have been proposed: no null values are permitted in a key, and there should be no duplication of the same tuple. The first restriction is forcefully stated, for example by Codd (1979, p. 400):

Rule 1 (Entity Integrity): No primary key value of a base relation is allowed to be null or to have a null component.

Insisting upon a fully defined key for each tuple automatically prohibits the existence of two (or more) tuples having identical keys. Such tuples can be merged (or recognized as inconsistent).

The above restrictions appear motivated by access necessities: if a key (or part of a key) is unknown, how can one access tuples? In practice, though, such restrictions prohibit the representation of much information that occurs frequently, and leads to a model of restricted practical use. Indeed, let us consider a data base of individuals, with a variety of properties for that individual and let us assume that a person's name is the key to the individual (and her properties).

With the restrictions that we have just mentioned, it is not possible to represent the information provided by observer 1:
"I saw somebody (but I don't know her name) with only one arm, wearing green trousers",
since the key (the person's name) is unknown, and hence null. Nor is there a way to represent the information provided by observer 2: "I saw the same person as observer 1, and I noticed that she had hazel eyes. But I don't know her name either".

If we assume that a data base is grown piecemeal, as a result of the contribution of a variety of observers (or informants), then we must remove the above restrictions. A tuple can be viewed as the result of an observation by an observer. Observations are then often incomplete. The question becomes: Can one still compute? That is, is it possible to answer queries from such unrestricted data bases? We shall now show, through a series of examples, that indeed we can still compute.

3. THE L^2 MODEL: AN EXTENDED RELATIONAL DATA BASE MODEL

We shall now describe our extended relational model, which will be referred to as the L^2 model for short. Although the model can be described formally and precisely, we shall attempt a more intuitive and explanatory description here.

A relation in the L^2 model consists of two-dimensional tables. The columns are labelled as attributes; the rows will also be referred to as tuples. We can think initially of each row as representing a set of partial observations by an informer on some individual (or entity). The system ALICE will try to recognize that several rows refer to the same individual.

The following values of attributes in a tuple are presently accepted in the L^2 model, and can be processed by ALICE:

1°/-Scalar values, such as "blue", "69", "potato".

2°/-Ranges of scalar values, such as blue or green or red}, meaning the value is blue, green or red; or such as {22..35}, meaning the value is between the values of 22 and 35, with ends included.

3°/-Non-relevant values, denoted NR. An attribute may have several non-relevant values. For example, the attribute "Name of Spouse" may have the value "NR1" to mean "never married", or NR2" meaning "Spouse is Dead". ALICE must be given the specific semantics of non-relevant values by means of formal definitions.

4°/-Unknown values. The attribute for a tuple has a value which is unknown in the observation represented by the tuple. Unknown values are represented by: ?. If two unknown values are the same, as for example in "I saw the same person as you did", then the unknown is indexed, for example as: ?5 .
The semantics are clear: ?i = ?i; while it is not generally known whether ?i = ?j, if $i \neq j$; or whether ?i = ?.

5°/-Functional dependencies, from one set of attributes to another set of attributes in a relation.

6°/-Keys, a special case of functional dependencies.

7°/-A variety of constraints.

ALICE was designed to manipulate constraints, and has proved very successful. Examples of constraints on the relation R, with attributes A and B, could be:
-"for each value of A, there may be at most two values of B".
(Application: a society where a person may have at most two spouses!)
-"the values of A must be at least twice larger than the values of B".

Since ALICE has been described extensively elsewhere, we shall not describe it here. The examples will illustrate her capabilities. (The original ALICE required no modifications for the present applications).

4. RELATED WORK

Few works in the area of extensions of relational data bases (to null values or ranges, for example) include running programs. An exception is Lipski (1979). Lipski describes a theory and a program to manipulate data bases where values are within a range. We note that the L^2 model includes such ranges and, in addition, null values, functional dependencies and a rich set of constraints.

ALICE has been run on the two problems mentioned by Lipski, which we shall repeat here:

Problem 1: The Database is:

OBJECT	AGE	DEPT #	HIREYEAR	SALARY
	(0,+∞)	{1,2,3,4,5}	70..80	(0,+∞)
x_1	60..70	{1,2,3,4,5}	70..75	(10000)
x_2	52..56	{2}	72..76	(0,20000)
x_3	30	{3}	70..71	(0,+∞)
x_4	(0,+∞)	{2,3}	70..74	(22000)
x_5	32	{4}	75	(0,+∞)

The query is:

[Dept # in (2,3)]∧
[[Salary < 10000)∧ Hireyear 72]∨
[Age > 50]∧[Salary < 150000]].

ALICE determines (in 1.5 seconds on a 370/168 computer), that x3 and x4 satisfy the query: that x2 might satisty the query; and that xl and x5 do not satisfy the query. These results agree with Lipski's.

Problem 2: The Database is:

NAME	SALARY	STATUS	# CHILDREN	DEPT #
"?"	(15000,18000)	{Married}	{3,4,5}	{2,3,4}
Brown	(18000)	{Married}	{2}	{2}

The query is:
[(Dept # = 3)∧ (Name = Jones)∨
(Name ≠ Smith)∧ (Dept # = 2)]∧
[(Salary < 15000)∨ (# Children > 2)∧
(Salary in (10000, 20000))∧ (Status = Married)].

ALICE determines (in 1 second) that "?" might answer the query, while Brown does not. Again, these results agree with Lipski's. See the exact formulation of these problems in the ALICE language in the annex.

5. EXAMPLES OF MANIPULATIONS IN THE L^2 MODEL AND SOLUTIONS BY ALICE

We shall briefly describe a problem illustrating the representational capabilities of the L^2 model and the data base reduction capabilities of ALICE *. When given a data base, ALICE first reduces the

* The example is somewhat artificial, so that it remains simple to describe.

database by eliminating redundant tuples, and finding values or ranges for unknowns.

The database is:

Tuple	A	B	C	D
1	(0∨4)	?1	c3	?2
2	(0∨4)	?3	c4	?4
3	(0∨6)	?5≠?1	c3	?6
4	?7	?8	c5	5
5	2	?9	c4	?10
6	?11	b1	c1	?12
7	1	b1	c1	?12
8	?19	?14	?15	?20
9	3	b1	?16	(1,2,3)
10	?17	b1	c2	?18

The constraints are:

(I) Keys: A & (B,C)
(II) F.D.: C → D
(III) Not more than 2 C's for 1 B
(IV) At least two different tuples with C = c1
(V) $0 \leqslant A \leqslant 10$; A integer.
(VI) $0 \leqslant D \leqslant 5$; D integer.
(VII) $A \geqslant 2D$.

Notice null values in the key A (in tuples 4,6, 8 and 10) and again in the B component of the key (B,C) (in tuples 1 to 5 and 8).
Tuple 8 is entirely unknown.

Indexed unknowns are used to indicate equality of unknowns, as in 6.D = 7.D, or explicity to indicate inequality. The constraint on line 3:"?5≠?1" means that: 3.B is equal to ?5 but must be different from ?1.

Solution by ALICE (4 seconds):

.By constraint (I): 3.A = 6. Tuples 1 and 2 give indeed that 1.A and 2.A form together the set {0,4}. Since A is a key, if 3.A = 0 then tuple 3 must be either tuple 1 or tuple 2. But both cases are impossible due to: ?5 ≠ ?1 and c3 ≠ c4.
.?4 = ?10 = (0∨1) by constraints (II) and (VII).
.?2 = ?6 by (II).
.?7 = 10 by (V) and (VII); ?6 ⩽ 3 by (VII).
.Tuple 6 = tuple 7 by (I),
therefore: ?11 = 1 and ?12 = 0 by (VII).
.?16 = c2 by (I) and (III) with 7.A ≠ 9.A together with 7.C = c1 and 10.C = c2.
.?15 = c1 by (IV) and ?14 ≠ b1 by (II).
.Tuple 9 = tuple 10 by (I).
therefore: ?17 = 3; ?18 = 1 by (VI) and (VII).
.b1 ≠ ?1, ?3, ?5, ?8, ?9 by (III).
.?19 ≠ 0,1,2,3,4,5, 10 by (I).
.?20 = 0 by (II).

The reduced database becomes:

Tuple #	A	B	C	D	
1	(0∨4)	?1≠b1	c3	?2=(0,1,2)	
2	(0∨4)	?3≠b1	c4	?4=(0,1)	
3	[6]	?5≠?1	c3	?2=(0,1,2)	
4	[10]	?8≠b1	c5	[5]	
5	[2]	?9≠b1	c4	?4=(0,1)	
7	[1]	b1	c1	[0]	← tuple 6 removed
8	?19	?14≠b1	c1	[0]	
9	[3]	b1	c2	[1]	← tuple 10 removed

The following questions are answered by ALICE in a total of 3 seconds:

Question	Answer
Q1: Tuples with D=5?	A1: Surely:4; Maybe: 8.
Q2: Tuples with B=b1?	A2: Surely: 7,9; Maybe: none.
Q3: Tuples with B=b2?	A3: Surely: none; Maybe: 1,2,3,4,5,8.
Q4: Tuples with B=b2 and D=3?	A4: Surely: none; Maybe 8.
Q5: Tuples with D=0?	A5: Surely: 7; Maybe: 1,2,3,5,8.
Q6: Tuples with A=0?	A6: Surely: none; Maybe:1,2.

(Actually A6 is more precise; Surely: either 1 or 2 but not both. Tuple 8 may not answer Query 6 by one of the last conclusions of the previous reduction process).

The reader can verify, by solving the problem, that some of the deductions, if not very difficult, are not trivial either.

NOTE: For each problem, ALICE receives the formal definitions of the keys, constraints and queries. She does not know the application a priori but interprets these formal data on the data base. An optimised "frozen" version of the system for this kind of application would probably reduce the running times significantly.

6. CONCLUSIONS

The L^2 data base model is a natural extension of the relational data base model. It allows null values in any field, including keys; the repetition of tuples, and therefore information from different and possibly coupled sources: and a great variety of constraints. Therefore, we can now model situations that were forbidden in previously proposed extensions.

The problem-solver ALICE reduces a given database, finds contradictions in it, and answers queries. The performance of ALICE in other domains indicates that constraints that are significantly more powerful than those previously allowed, or even those shown in this article, can be successfully accepted in a relational database.

We are pursuing our study of the L^2 model, in particular in the areas of non-relevant nulls (which were not given in our examples here); secondary storage handling; and the incorporation of significantly more challenging constraints.

Acknowledgments

Some of this work was pursued while one of the authors (LS) was Visiting Professor at the University of Paris IX Dauphine. The support of Paris IX is gratefully acknowledged.

ANNEX

The statements in ALICE of two of the problems, together with the run of one, follow:

Problem 1: Give the constraints:

range of age:	ia = 0;	sa = 100.
range of department:	id = 1;	sd = 5.
range of hireyear:	ih = 70;	sh = 80.
range of salary:	is = 0;	ss = 90.

Given the sets: X = [1, 1]
DA = [ia, sa] DD = [id, sd]
DH = [ih, sh] DS = [is, ss]

Find the functions:
AGE from X to DA DEP from X to DD
HYE from X to DH SAL from X to DS

With the constraint (question):
Dep(X) ⩾ 2 and Dep(X) ⩽ 3 and
(Sal(X) > 10 or Hye(X) ⩽ 12) and
(Age(X) ⩽ 50 or Sal(X) > 15)

All these formulas are written, for the system ALICE, in polish notation. The full problem specifications follow:

```
S OI CST   IA   SA   ID   SD   IH   SH   IS   SS
ENS X = INT I I          DA = INT  IA  SA
   DD = INT  ID  SD      DH = INT  IH  SH
   DS = INT  IS  SS
TRO FON AGE >-> X DA    DEP >-> X DD
        HYE >-> X DH    SAL >-> X DS
AVE ET  >=  DEP 1    2 <= DEP 1  3
    OU  <=  HYE 1   12 >  SAL 1  10
    OU  <=  AGE 1   50 >  SAL 1  15
FIN    0,100,1,5,70,80,0,90
```

The other problems are given in the same manner: objects and relations are formally defined; key or (FD) correspond to partial or total onto functions.

Problem 2 gives for instance:

```
SOI  ENS  X    =    INT  1   1
          DD   =    INT  2   2
          DN   =    OBJ
          DS   =    INT  18  18
          DM   =    OBJ
          DC   =    INT  2   2
TRO  FON  DEP  >->  X    DD
          NAM  >->  X    DN
          STA  >->  X    DM
          SAL  >->  X    DS
          CHI  >->  X    DC
AVE  OU   ET    =    DEP  1   3  =  NAM 1  JON
          ET   ≠    NAM  1   SMI =  DEP 1  2
     OU   <    SAL  1    15  >  CHI 1  2
      =   NAM  1    BRO
     OU   >    SAL  1    10  <  SAL 1  20
      =   STA  1    MAR
FIN    BRO    JON    SMI    MAR    FIN
30,30,3,3,70,71,0,100
```

Comments: the set DN of the possible names is read as data: {Brown, Jones, Smith}, the set DM is {married}. The form of the constraints is similar. The run is:

```
call   .II(m)
PHIO***

        internal        FINDIC          NIFI=     0
        translation
NNT=       5        NSG=      5     NSD=       7
NK=      108        PE=      13     PF=        1
PCG=       1        PLG=    121     PSG=       1

8  0  12  410  1400  400  120  120 1500 7000 200 1
CTTE 1 OU  ET = 0 + MNS 3 DEP 1 = 0 + MNS 2
NAM 1 ET ≠ 0 + MNS 3 NAM 1 = 0 + MNS 2 DEP 1
                  ←impossibility of the constraint:
CTTE   2   NON   sal(x)< 15 or children(x)>2
NSOL=      0;
****      AUCUNE SOLUTION AU PROBLEME  PHIO***
IERR=      0;←no solution
NIFI=     14          PL=       1020
READY   ←number of unifications
time
TIME-03:55:39 PM.  CPU-00:00:01  SERVICE-5060
SESSION-00:05:29 NOVEMBER 20
```

7. REFERENCES

[1] Codd, E.F. "Understanding Relations." FDT Bulletin of ACM-SIGMOD 7 (3-4), 1975, pp. 23-28.

[2] Codd, E.F., "Extending the Data Base Relational Model to Capture More Meaning." ACM Trans. Data Base Syst., 4,4 1979, pp 397-434.

[3] Date, J.C.: An Introduction to Database Systems (3rd ed.), Addison Wesley, 1981.

[4] Gallaire, H. and J. Minker, eds: Logic and Data Bases. Plenum Press (1978).

[5] Grant, J. "Null Values in a Relational Data Base." Inf. Proc. Letters, 6,5, 1977, pp. 156-157.

[6] Imielinski, T. and W. Lipski: On representing incomplete information in a relational database. Proc. VLDB , 1981, pp. 388-397.

[7] Laurière, J.L.: A language and a program for stating and solving combinatorial problems. Artificial Intelligence 10 (1978) pp. 29-127.

[8] Lipski, W.: On semantic issues connected with incomplete information databases. ACM Trans. Database Syst. 4, 1979, pp. 262-296.

[9] Siklóssy, L: Efficient query evaluation in relational databases with missing values. Inf. Proc. Letters, 13,4/5, 1981, pp. 160-163.

[10] Ullman, J.D.: Principles of Database Systems, Computer Science Press, 1980.

[11] Vassiliou, Y.: Null values in database management: A denotational semantics approach, Proc. ACM SIGMOD 1979 Int. Conf. on Manag. of data, 1979, pp. 162-169.

[12] Vassiliou, Y.: Functional dependencies and incomplete information, Proc. VLDB, 1980, pp. 260-269.

[13] Wang, H: Toward mechanical mathematics, IBM, J. Res. Dev., 4,1, 1960.