FC: A RELATIONAL QUERY LANGUAGE HAVING THE PROPERTY OF FUNCTIONAL COMPLETENESS

MARIA C. FERNANDEZ-BAIZAN
RAFAEL PORTAENCASA BAEZA

Facultad de Informatica, Universidad Politecnica de Madrid, E-28660 Boadilla del Monte, Madrid, Spain

ARTURO RIBAGORDA GARNACHO

Universidad "Carlos III," Madrid, Spain

Functional completeness of a relational language is the ability to express linear recursive queries. We present such a language that also has the property of relational completeness (relational algebra is, in fact, embedded in it). The transitive closure of a binary relation is carried out by means of the projection function and its inverse.

INTRODUCTION

As stated by Ioannidis and Wong (1987), relational algebra does not have functional completeness, because of the impossibility of expressing transitive closure by means of its operators.

Our aim is to show the foundations of a new query language that has relational completeness (meaning that relational algebra is, in fact, embedded in it) and also functional completeness (meaning that it allows for the expression of the transitive closure of a binary relation and thus the expression of linear recursive inferences).

PREVIOUS DEFINITIONS

Relations

Let $U = \{A_1, A_2, \ldots, A_n\}$ be an attribute set (universe).
Let dom(A_i) be the domain of A_i ($i = 1, 2, \ldots, n$)
We call a relation scheme any element of the power set $P(U)$.
Let T be the set of all universal tuples

$$T = \text{dom}(A_1) \times \ldots \times \text{dom}(A_n)$$

The set of all universal relations is the power set $P(T)$.

Projection Functions

We have adapted definitions given by Dugundji (1966), taking into account that, since relations are equivalent under column permutations, here "projection on i-component" means "projection on i-attribute."

Projection functions are the following:

$p_i: T \rightarrow \text{dom}(A_i)$, in such a way that $\forall\, t \in T$, $p_i(t)$ = value of attribute A_i in t.

$p_{ij}: T \rightarrow \text{dom}(A_i) \times \text{dom}(A_j)$, in such a way that $\forall\, t \in T$, $p_{ij}(t)$ = pair of values of A_i and A_j in t.

and so on (for three or more indexes).

$\pi_i: P(T) \rightarrow P(\text{dom}(A_i))$, in such a way that

$\forall\, r \in P(T)$, $\pi_i(r) = \{p_i(t) \mid t \in r\}$

$\pi_{ij}: P(T) \rightarrow P(\text{dom}(A_i) \times \text{dom}(A_j))$ in such a way that

$\forall\, r \in P(T)$, $\pi_i(r) = \{p_{ij}(t) \mid t \in r\}$

and so on (for three or more indexes).

Inverse functions are defined as follows:

$\pi_i^{-1}: P(\text{dom}(A_i)) \rightarrow P(T)$

$\forall\, s \in P(\text{dom}(A_i))$; $\pi_i^{-1}(s) = \{t \in T \mid pi(t) \in s\}$

$\pi_{ij}^{-1}: P(\text{dom}(A_i) \times \text{dom}(A_j)) \rightarrow P(T)$

$\forall\, u \in P(\text{dom}(A_i) \times \text{dom}(A_j))$, $\pi_{ij}^{-1}(u) = \{t \in T \mid p_{ij}(t) \in u\}$

and so on for three or more indexes.

PROPERTIES OF PROJECTION FUNCTIONS

The following properties may easily be stated:

a) $\forall\ r \in P(dom(A_i)),\ \pi_i^{-1}(r) = dom(A_1) \times dom(A_2) \times \ldots \times dom(A_{i-1}) \times r \times dom(A_{i+1}) \times \ldots \times dom(A_n)$

b) $\forall\ r \in P(dom(A_i) \times dom(A_j))$

$\pi_{ij}^{-1}(r) = r \times dom(A_1) \times \ldots \times dom(A_{i-1}) \times dom(A_{i+1}) \times \ldots \times dom(A_{j-1}) \times dom(A_{j+1}) \times \times \ldots \times dom(A_n)$

From this, it is easy to show that:

$r\ |\times|\ s = \pi_{ijk}(\pi_{ij}^{-1}(r) \cap \pi_{jk}^{-1}(s))$

If $r \in P(dom(A_i) \times dom(A_j))$ and $s \in P(dom(A_k) \times dom(A_1))$

$r \times s = \pi_{ijk1}(\pi_{ij}^{-1}(r) \cap \pi_{k1}^{-1}(s))$

c) $\forall\ r \in P(T),\ r \subseteq \pi_1^{-1}\ \pi_i(r)$

d) $\forall\ s \in P(dom(A_i))$

$\pi_i\ \pi_1^{-1}(s) = s$

e) π_1 is a monotonic operator:

$\forall\ r \in P(T),\quad \forall\ s \in P(T)$, if $r \subseteq s$, then $\pi_i(r) \subseteq \pi_i(s)$

f) π_i^{-1} is a monotonic operator:

$\forall\ r \in P(dom(A_i)),\quad \forall\ s \in P(dom(A_i))$ if $r \subseteq s$, then $\pi_1^{-1}(r) \subseteq \pi_1^{-1}(s)$

g) $\forall\ r_1 \in P(dom(A_i))$; $r_1 \times r_2 \times \ldots \times r_n = \pi_1^{-1}(r_1) \cap \ldots \cap \pi_n^{-1}(r_n)$

h) $\forall\ r \in P(dom(A_i)),\ \forall\ s \in P(dom(A_i))$

$\pi_1^{-1}(r \cup s) = \pi_i^{-1}(r) \cup \pi_1^{-1}(s)$

$\pi_1^{-1}(r-s) = \pi_i^{-1}(r) - \pi_1^{-1}(s)$

i) $\forall\ r \in P(T),\ \forall\ s \in P(T)$

$\pi_i(r \cup s) = \pi_i(r) \cup \pi_i(s)$

$\pi_i(r - s) \supseteq \pi_i(r) - \pi_i(s)$

A LANGUAGE THAT HAS FUNCTIONAL COMPLETENESS

Our query language is constituted by well-formed expressions based on the following operators:

Set union
Set intersection
Set difference
Projection (π)
Inverse projection (π^{-1})
Composition of functions ($\pi\pi^{-1}$)

Some well-formed expressions of the language:

$\pi_i \pi_j^{-1}$ (r), $r \in P(dom(A_j))$

$\pi_i \pi_j^{-1} \pi_j \pi_i^{-1}$(r), $r \in P(dom(A_i))$

π_i^{-1}(r) $\cap$ π_j^{-1} (s), $r \in P(dom(A_i))$, $s \in P(dom(A_j))$

RELATIONAL COMPLETENESS OF THE LANGUAGE

As stated by Codd (1972), a base for a complete relational algebra is constituted by set union, set difference, Cartesian product, selection, and projection. From foregoing sections on properties and language, for establishing the relational completeness of the language we need only demonstrate that selection may be expressed in it.

Let s be a (perhaps nonuniversal) relation defined on attributes $\{A, B, C\}$. The following may easily be shown:

$$\sigma_{A=a}(s) = s \cap \pi_{123}\, \pi_1^{-1}(\{a\})$$

$$\sigma_{A=a \wedge B=b}(s) = s \cap \pi_{123}\, \pi_{12}^{-1}(\{ab\})$$

$$\sigma_{A=a \vee B=b}(r) = (s \cap \pi_{123}\, \pi_1^{-1}(\{a\})) \cup (s \cap \pi_{123}\, \pi_2^{-1}(\{b\}))$$

$$\sigma_{A \neq a}(s) = s \cap \pi_{123}(s - \pi_1^{-1}(\{a\}))$$

$$\sigma_{c \geq 1}(s) = s \cap \pi_{123}\, \pi_3^{-1}(r)$$

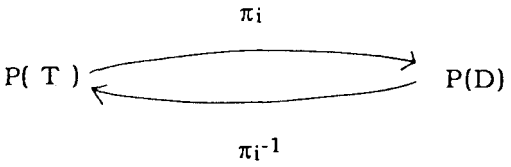
being $r = \{x \in dom(c) \quad x \geq 1\}$

EXPRESSION OF RECURSIVE QUERIES IN THE LANGUAGE

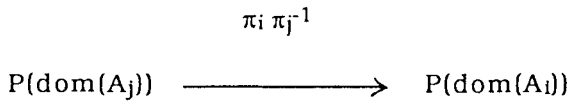
When posing recursive queries we suppose that all domains of the attributes involved in the query are the same:

$$dom(A_1) = \ldots = dom\ (A_n) = D$$

For any i, we have:



From the fact that, for any i, j, we have:



The following result may be stated for recursive inferences:

and $(\pi_i \pi_j^{-1})^n$ is a well-formed expression of the language [that it is not the case if $dom(A_i) \neq dom(A_j)$].

Let us have the following relation:

A	B	C
b	c	d
b	d	c
c	e	b
c	e	d
d	a	b
e	a	d

dom(A) = dom(B) = dom(C)
(we suppose the attributes ordered from left to right)

Let $s = \{b, c\}$. If we want to compute all the 21 ancestors of s, we have the following sequence:

$\pi_2\,\pi_1^{-1}$ ({b,c}) = {c,d,e}

$(\pi_2\,\pi_1^{-1})^2$ ({b,c}) = $\pi_2\,\pi_1^{-1}$ ({c,d,e}) = {e,a}

$(\pi_2\,\pi_1^{-1})^3$ ({b,c}) = $\pi_2\,\pi_1^{-1}$ ({e,a}) = {a}

$(\pi_2\,\pi_1^{-1})^4$ ({b,c}) = $\pi_2\,\pi_1^{-1}$ ({a}) = $\varnothing$

with $I_{21}(\{b, c\}) = \{c, d, e\} \cup \{e, a\} \cup \{a\} = \{a, c, d, e\}$.

In general, $A_{ij}^k(s)$ *is the set of* k-order ij ancestors of s; that is,

$A^k{}_{ij}$ (s) = $(\pi_i\,\pi_j^{-1})^k$ (s), we have:

$I^n{}_{ij}$ (s) = $A^1{}_{ij}$ (s) $\cup$ $A^2{}_{ij}$ (s) $\cup$ ... $\cup$ $A^n{}_{ij}$ (s)

After some simple algebraic transformations, we may state that $I_{ij}(s)$ is the least fix point of the equation: $I_{ij}^n(s) = A_{ij}^1(s) \cup A_{ij}^1(I_{ij}^{n-1}(s))$, whose solution may be posed as the iterative problem:

$I^n{}_{ij}$ (s) = $A^1{}_{ij}$ (s) $\cup$ $A^1{}_{ij}$ ($I^{n-1}{}_{ij}$ (s))
$I^1{}_{ij}$ (s) = $A^1{}_{ij}$ (s)

Taking into account the results of Ioannidis and Wong (1987), we may conclude that this problem pertains to the class of recursivity limited by N, N depending, in general, on the extension of the initial relation r.

COMPUTING $I_{ij}(s)$

Coming from:

$I^1{}_{ij}$ (s) = $A^1{}_{ij}$ (s)

$I^n{}_{ij}$ (s) = $A^1{}_{ij}$ (s) $\cup$ $A^1{}_{ij}$ ($I^{n-1}{}_{ij}$ (s))

and taking into account that $I_{ij}(s) = I_{ij}^n(s)$ if $I_{ij}^n(s) = I^{n-1}(s)$, we propose the following (nonoptimistic) algorithm for computing $I_{ij}(s)$:

X: = $\pi_i\pi^{-1}{}_j$ (s)
Y: = X
DO UNTIL Z = Y
Z: = Y
Y: = X $\cup$ $\pi_i\pi^{-1}{}_j$ (Y)

The transformation in a relational algebra program is straightforward, because the only operator that does not belong to relational algebra is π^{-1}, but it can easily be shown that it may be calculated as the union of several sections.

CONCLUDING REMARKS

We have shown that the language FC provides for the expression of linear recursive queries. We plan to continue this work by doing the following:

Optimize the algorithm shown in the last section
Implement the language (perhaps on SQL)
Extend the language for expressing recursive queries of higher order.

REFERENCES

Agrawal, R. 1987. Alpha: An Extension of Relational Algebra to Express a Class of Recursive Queries. *Proc. IEEE Third International Conference on Data Engineering*, pp. 80–90.

Agrawal, R., and P. Devanbu. 1988. Moving Selections into Linear Least Fixpoint Queries. IEEE, pp. 452–461.

Agrawal, R., and H. V. Jagadish. 1987. Direct Algorithms for Computing the Transitive Closure of Database Relations. *Proc. 13th VLDB*, Brighton.

Bacilhon, F. 1978. On the Completeness of Data Base Sublanguages. In *Data Base Systems*, ed. R. Rustin. Englewood Cliffs, NJ: Prentice-Hall.

Bancilhon, F., and R. Ramakrishnan. 1986. An Amateur's Introduction to Recursive Query Processing Strategies, *Proc. ACM Sigmod*, pp. 16–52.

Codd, 1972. Relational Completeness of Data Base Sublanguages. In *Data Base Systems*, ed. R. Rustin. Englewood Cliffs, NJ: Prentice-Hall.

Dugundji, J. 1966. *Topology.* Boston: Allyn & Bacon.

Eder, J. 1989. Extending SQL with General Transitive Closure and Extreme Value Selections. *IEEE Trans. Knowl. Data Eng.* 2(4):381–390.

Güntzer, U., W. Kiessling, and R. Bayer. 1987. On the Evaluation of Recursion in (Deductive) Database Systems by Efficient Differential Fixpoint Iteration. IEEE, pp. 120–129.

Golovko. 1985. Recursive Axioms in Deductive Data Bases. *Tejnícheskaya Kibernetika* 5:230–233.

Helm, A. R. 1988. Detecting and Eliminating Redundant Derivations in Deductive Database Systems. IBM Research Report RC 14244.

Ioannidis, Y. E. 1985. A Time Bound on the Materialization of Some Recursively Defined Views. *Proc. 11th VLDB,* Stockholm.

Ioannidis, Y. E., and E. Wong. 1987. An Algebraic Approach to Recursive Inference. In *Expert Database Systems,* ed. L. Kerschberg, pp. 295–309. Menlo Park: Benjamin/Cummings.

De Maindreville, Ch. 1985. Evaluation of Recursive Predicats in Deductive Databases.

Minker, J., and J. M. Nicolas. 1983. On Recursive Axioms in Deductive Databases. *Inform. Syst.* 8(1):1–13.

Nejdl, W. 1987. Recursive Strategies for Answering Recursive Queries—The RQA/FQI Strategy. *Proc. 13th VLDB,* Brighton, pp. 43–50.

Rohmer, J., R. Lescoeur, and J. M. Kerisit. 1986. The Alexander Method: A Technique for the Processing of Recursive Axioms in Deductive Databases. *New Generation Comput.* 4:273–285.

Rogers, H. 1967. *Some Problems of Definability in Recursive Function Theory.* Amsterdam: North Holland.

Sernadas, A. 1980. Logical Procedure Definition for Information System Specification. Ph.D. thesis, University of London.

Ullman, J. 1988. *Principles of Data Base and Knowledge Base Systems,* Vol. I. Rockville, MD: Computer Science Press.

Vieille, L. 1986. Recursive Axioms in Deductive Databases: The Query/Subquery Approach, Expert Database Systems, Kerschberg, pp. 253–268.

Wong, W. Ch., and L. Bic. 1987. A Tagging Scheme to Prevent Infinite Recursion in First-Order Databases. *IEEE Second International Conference on Computers and Applications,* pp. 480–481.

Requests for reprints should be sent to Maria C. Fernandez-Baizan.