CONDITIONAL TABLES

Gösta Grahne
Department of Comptuter Science
Concordia University, Montreal, Canada
grahne@cs.concordia.ca

SYNONYMS

C-Tables; Extended Relations

DEFINITION

A conditional table [3] generalizes relations in two ways. First, in the entries in the columns, variables, representing unknown values, are allowed in addition to the usual constants. The second generalization is that each tuple is associated with a condition, which is a Boolean combination of atoms of the form $x = y, x = a, a = b$, for x, y null values (variables), and a, b constants. A conditional table essentially represents an existentially quantified function free first order theory.

Formally, let *con* be a countably infinite set of constants, and *var* be a countably infinite set of variables, disjoint from *con*. Let U be a finite set of attributes, and $R \subseteq U$ a relational schema. A tuple in a c-table over R is a mapping from R, and a special attribute, denoted φ, to $con \cup var \cup \beta$, where β is the set of all Boolean combinations of equality atoms, as above. Every attribute in R maps to a variable or a constant, and φ maps to β. In a multirelational database schema, there are multi-tables, meaning in effect that variables can be shared between tables (just as constants are). An example of a 2-multitable is shown below. The conditions φ are all *true*, and it so happens that the two tables do not share any variables.

T_1

A	B	C	φ
a	b	x	*true*
e	f	g	*true*

T_2

B	C	D	φ
y	c	d	*true*

MAIN TEXT

It is now possible to extend the complete set of regular relational operators $\{\pi, \sigma, \bowtie, \cup, -, \rho\}$ to work on c-tables. To distinguish the operators that apply to tables from the regular ones, the extended operators are accented by a dot. For instance, c-table join is denoted $\dot{\bowtie}$. The extended operators work as follows: projection $\dot{\pi}$ is the same as relational projection, except that the condition column φ can never be projected out. Selection $\dot{\sigma}_{A=a}(T)$ retains all tuples t in table T, and conjugates the condition $t(A) = a$ to $t(\varphi)$. A join $T \dot{\bowtie} T'$ is obtained by composing each tuple $t \in T$ with each tuple $t' \in T'$. The new tuple $t \cdot t'$ has condition $t(\varphi) \wedge t'(\varphi) \wedge \delta(t, t')$, where condition $\delta(t, t')$ states that the two tuples agree on the values of the join attributes. The example below serves as an illustration of this definition. The union $\dot{\cup}$ is the same as relational union, and so is renaming $\dot{\rho}$, except that the φ-column cannot be renamed. Finally, the set difference, say $T \dot{-} T'$ is obtained by retaining all tuples $t \in T$ and conjugating to them the condition stating that the tuple t differs from each tuple t' in T'. A tuple t differs from a tuple t', if it differs from t' in at least one column.

Let T_1 and T_2 be as in the figure above. The three c-tables in the figure below, from left to right, illustrate the result of evaluating $\dot{\sigma}_{C=g}(T_1)$, $T_1 \dot{\bowtie} T_2$, and $\dot{\pi}_{BC}(T_2) \dot{-} \dot{\pi}_{BC}(T_1)$, respectively.

A	B	C	φ
a	b	x	$x = g$
e	f	g	$g = g$

A	B	C	D	φ
a	y	x	d	$x = c \wedge y = b$
e	y	g	d	$y = f \wedge g = c$

C	D	φ
y	c	$(y \neq b \vee c \neq x) \wedge (y \neq f \vee c \neq g)$

Note that the second tuple in the leftmost c-table has a tautological condition. Likewise, since any two constants differ, the condition of the second tuple in the middle c-table is contradictory.
So far, nothing has been said about what the c-tables mean. In the *possible worlds* interpretation, an incomplete database is a usually infinite *set* of ordinary databases, one of which corresponds to the actual (unknown) database. Considering c-tables, they serve as finite representations of sets of possible databases. One (arbitrary) such database is obtained by instantiating the variables in the c-table to constants. Each occurrence of a particular variable is instantiated to the same constant. Formally, the instantiation is a valuation $v : con \cup var \longrightarrow con$, that is identity on the constants. Valuations are extended to tuples and conditions tables in the obvious way, with the caveat that given a particular valuation v, only those tuples t for which $v(t(\varphi)) \equiv true$, are retained in $v(T)$. Consider for instance the leftmost table above. For those valuations v, for which $v(x) = g$, there will be two tuples in $v(T)$, namely (a, b, g) and (c, f, g). For valuations v', for which $v(x) \neq g$, there will only be the tuple (c, f, g) in $v'(T)$.
The remarkable property of c-tables is that for all c-tables T and relational expressions E, it holds that $v(\dot{E}(T)) = E(v(T))$ for all valuations v. In other words, the extended algebra is a Codd sound and complete inference mechanism for c-tables. Furthermore, c-tables are closed under relational algebra, meaning that the result of applying any relational expression on any (schema-wise appropriate) c-table can be represented as another c-table. The extended algebra actually computes this representation, as was seen in the example above.
Needless to say, all of this comes with a price. Testing whether a c-table is satisfiable, that is, whether there exists at least one valuation T, such that $v(T) \neq \emptyset$ is an NP-complete problem [2]. Furthermore, even if one starts with a simple c-table where all variables are distinct, and all conditions are *true*, applying even a monotone relational expression to such a c-table can result in quite a complex table, so here again [2] satisfiability of the resulting table is NP-complete [2]. To make matters even worse, testing containment of c-tables is Π_2^p-complete. A c-table T is contained in a c-table T', if every for every valuation v of T, there exists a valuation v' of T', such that $v(T) = v(T')$. Nonetheless, c-tables possess a natural robustness. For instance, it has been shown [1, 4] that the set of of possible databases defined by a set of materialized views, can be represented as a c-table, as long the view defining queries do not use difference. This holds both for the open and closed world assumptions.

CROSS REFERENCE

Incomplete Information, Naive tables, Certain answer, Maybe answer.

RECOMMENDED READING

[1] Serge Abiteboul, Oliver M. Duschka: Complexity of Answering Queries Using Materialized Views. *PODS* 1998: 254-263
[2] Serge Abiteboul, Paris C. Kanellakis, Gösta Grahne: On the Representation and Querying of Sets of Possible Worlds. *Theor. Comput. Sci.* 78(1): 158-187 (1991)
[3] Tomasz Imielinski, Witold Lipski Jr.: Incomplete Information in Relational Databases. *J. ACM* 31(4): 761-791 (1984)
[4] Gösta Grahne, Alberto O. Mendelzon: Tableau Techniques for Querying Information Sources through Global Schemas. *ICDT* 1999: 332-347