Database Management Systems (DBMS)

Lec 7: Relational model of data (Cont.)

Ramesh K. Jallu

IIIT Raichur

Date: 12/2/21

Recap

- One example of the division operation (÷)
- Relational Calculus
  - TRC

Today's plan

- Relational Calculus
  - TRC
    - Rules to construct TRC expressions
  - DRC
    - Rules to construct DRC expressions

Formal definition

- TRC is based on specifying a number of *tuple variables*
  - Each tuple variable belongs to a relation
- A TRC expression is of the form: {*t* | *P*(*t*)}
- A tuple variable is ***bound*** if it is quantified; otherwise, it is ***free***

$$t \in \text{EMPLYOEE} \wedge \exists\, s \in \text{DEPARTMENT}(t[\text{Dno}] = s[\text{Dnumber}])$$

Atoms in logic

- An atomic formula (or simply an ***atom***) is an assertion, without any logical connectives, that must be *true* or *false*
- A propositional formula is constructed from atomic formula by using logical connectives
  - e.g., $(\boldsymbol{A} \wedge \boldsymbol{B}) \vee \boldsymbol{C} \Rightarrow \boldsymbol{B}$
- The truth value of a propositional formula can be calculated from the truth values of the individual atomic propositions it contains

Atomic formula in TRC

- In TRC, a formula is made up of atoms, which has one of the following forms
  - $t \in R$, where t is a tuple variable and R is a relation ($\notin$ operator is not allowed)
  - $t[A]\ \theta\ s[B]$, where t and s are tuple variables, A is an attribute on which t is defined, B is an attribute on which s is defined, and θ is a comparison operator
  - $t[A]\ \theta\ \boldsymbol{c}$, where t is a tuple variable, A is an attribute on which t is defined, θ is a comparison operator, and $\boldsymbol{c}$ is a constant in the domain of attribute A

Rules to build up formulae from atoms

- An atom is a formula
- If P_1 is a formula, then so is $\neg P_1$
- If P_1 and P_2 are formulae, then so are $P_1 \vee P_2$, $P_1 \wedge P_2$, and $P_1 \Rightarrow P_2$
- If $P_1(\boldsymbol{s})$ is a formula containing a free tuple variable $\boldsymbol{s}$, and $\boldsymbol{R}$ is a relation, then $\exists\ \boldsymbol{s} \in \boldsymbol{R}(\boldsymbol{P_1}(\boldsymbol{s}))$ and $\forall\ \boldsymbol{s} \in \boldsymbol{R}(\boldsymbol{P_1}(\boldsymbol{s}))$ are also formulae

Equivalent expressions

1. $P_1 \wedge P_2$ is equivalent to $\neg (\neg(P_1) \vee \neg(P_2))$
2. $\forall\, t \in R(P_1(t))$ is equivalent to $\neg\, \exists\, t \in R(\neg P_1(t))$
3. $P_1 \Rightarrow P_2$ is equivalent to $\neg(P_1) \vee P_2$

An issue

- In TRC, an expression may generate an infinite relation
  - e.g., {*t* |¬ (*t* ∈ EMPLOYEE)}
- ***Safe expression*** is guaranteed to yield a finite number of valid tuples
- **Domain of a TRC expression**: the set of all values that appear as constant values in the expression or exist in any tuple in the relations referenced in the expression

An example

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

dom({t | $t \in$ EMPLYOEE $\wedge$ t[salary] > 50000}) is the set containing 50000 as well as the set of all values appearing in any attribute of any tuple in the EMPLOYEE relation

dom({t |¬ ($t \in$ EMPLOYEE)})?

Testing

- An expression is said to be safe if all values in its ***result*** are from the domain of the expression

- $\{t \mid \neg (t \in \text{EMPLOYEE})\}$ is not safe

- $\{t \mid \exists u \in \text{EMPLYOEE}(t[\text{FName}] \neq u[\text{Fname}] \wedge \forall s \in \text{DEPARTMENT}(t[\text{Dno}] \neq s[\text{Dnumber}]))\}$ is not safe

- What does exactly the above expression retrieve?

Testing

- An expression is said to be safe if all values in its ***result*** are from the domain of the expression

- $\{t \mid \neg (t \in \text{EMPLOYEE})\}$ is not safe

- $\{t \mid \exists u \in \text{EMPLYOEE}(t[\text{FName}] \neq u[\text{Fname}] \wedge \forall s \in \text{DEPARTMENT}(t[\text{Dno}] \neq s[\text{Dnumber}]))\}$ is not safe

- What does exactly the above expression retrieve?
  - Tuples whose first name don't appear in EMPLOYEE table and also that their department number does not appear in any record of the DEPARTMENT table

The domain relational calculus

- SQL was developed based on TRC
- Query-by-example (QBE) was developed based on DRC
  - QBE allows users to retrieve information in tables by providing a simple user interface where the user will be able to input an example of the data that the user wants to access
  - An abstraction between the user and the real query that the database system will receive
  - In the background, the user's query is transformed into a database manipulation language form such as SQL, and the SQL statement will be executed in the background

The domain relational calculus

- A DRC expression is of the form: $\{<x_1, x_2, \ldots, x_n> \mid P(x_1, x_2, \ldots, x_n)\}$
  - where $x_1, x_2, \ldots,$ *and* x_n are domain variables
  - P represents a formula composed of atoms

- Q_1: Find the ID, name, dept number, and salary for instructors whose salary is greater than 90,000/-
  - $\{ <i, f, d, s> \mid <i, f, d, s> \in \text{Instructor} \wedge s > 90000\}$
  - o/p: {<22222, Einstein, Physics, 95000>, <83821, Brandt, Comp. Sci., 92000>}

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

- Q_2: $\{ < i > \mid \exists\, n, d, s\ (< i, n, d, s > \in \text{Instructor} \wedge s > 80000)\}$

Atomic formula in DRC

- In DRC, a formula is made up of atoms, which has one of the following forms

  - $<x_1, x_2, \ldots, x_n> \in R$, where R is a relation on n attributes and $x_1, x_2, \ldots, x_n$ are domain variables or domain constants

  - $x \Theta y$, where x and y are domain variables and Θ is a comparison operator ($<, \leq, =, \neq, >, \geq$)

  - $x \Theta \mathbf{c}$, where x is a domain variable, Θ is a comparison operator, and $\boldsymbol{c}$ is a constant in the domain of the attribute for which x is a domain variable

Rules to build up formulae from atoms

- An atom is a formula
- If P_1 is a formula, then so is $\neg P_1$
- If P_1 and P_2 are formulae, then so are $P_1 \vee P_2$, $P_1 \wedge P_2$, and $P_1 \Rightarrow P_2$
- If $P_1(x)$ is a formula in x, where x is a free domain variable, then $\exists\, x\, (P_1(x))$ and $\forall\, x\, (P_1(x))$ are also formulae

Example queries

Q_1: List the birth date and address of the employee whose name is ‘John B. Smith’.

$\{<u,v> \mid \exists q,r,s,t,w,x,y,z(< q,r,s,t,u,v,w,x,y,z> \in \text{EMPLYOEE} \wedge q = \text{'John'} \wedge r = \text{'B.'} \wedge s = \text{'Smith'})\}$

Thank you!