SEMINAR

Sandra Anna Joshy

S2MSc computer science

WELCOME

Tuple Relation Calculus and Domain Relation Calculus

Contents

RELATIONAL CALCULUS

- Relational calculus is a non-procedural query language,that tells what to do but never explains how to do.

- Relational calculus is of two types:

    1.Tuple relational calculus

    2.Domain relational calculus

1.TUPLE RELATION CALCULUS

Tuple Relational Calculus

- A nonprocedural query language.
- Query: $\{t \mid P(t)\}$.
- Free variable unless it is quantified by a $\exists$ or $\forall$.
- $t \in instructor \wedge \exists s \in department(t[dept\_name] = s[dept\_name])$.
- A TRC formula is built up out of atoms.
  - $\rightarrow s \in r$
  - $\rightarrow s[x]\ (<, \leq, =, \neq, >, \geq)\ u[y]$
  - $\rightarrow s[x]\ (<, \leq, =, \neq, >, \geq)\ c$

Tuple Relational Calculus

- ★ An atom is a formula.
- ★ If P_1 is a formula, then so are $\neg P_1$ and (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 $P_1(s)$ is a formula containing a free tuple variable s, and r is a relation, then $\exists s \in r\ (P_1(s))$ and $\forall s \in r\ (P_1(s))$.
- ★ $P_1 \wedge P_2$ is equivalent to $\neg(\neg(P_1) \vee \neg(P_2))$.
- ★ $\forall t \in r\ (P_1(t))$ is equivalent to $\neg \exists t \in r\ (\neg P_1(t))$.
- ★ $P_1 \Rightarrow P_2$ is equivalent to $\neg(P_1) \vee P_2$.

Predicate Relational Calculus

- Set of attributes and constants.
- Set of comparison operators: ($<$, $\leq$, $=$, $\neq$, $>$, $\geq$).
- Set of connectives: $\wedge$, $\vee$, $\neg$.
- Implication ($\Rightarrow$) Example: $P_1 \Rightarrow P_2$.
- Set of Quantifiers: $\exists$ and $\forall$.

Safety of Expressions

- ★ $\{t \mid \neg(t \in instructor)\}$.
- ★ Unsafe expressions.
- ★ Domain of a tuple relational formula.
- ★ dom(P).
- ★ $dom(t \in instructor \land t[salary] > 80000)$.
- ★ $\{t \mid P(t)\}$ is safe if all values in the result are from dom(P).

Example Queries

Example 1: Find the ID, name, dept name, salary for instructors whose salary is greater than $80,000:

$\{t \mid t \in instructor \wedge t[salary] > 80000\}$

Example 2: Find the instructor ID for each instructor with a salary greater than $80,000:

$\exists t \in r\ (Q(t))$

$\{t \mid \exists s \in instructor\ (t[ID] = s[ID] \wedge s[salary] > 80000)\}$

Example Queries

Example 3: Find the names of all instructors whose department is in the Watson building:

$\{t \mid \exists s \in instructor\ (t[name] = s[name] \land \exists u \in department\ (u[dept\ name] = s[dept\ name] \land u[building] = \text{"Watson"}))\}$

Example Queries

Example 4: Find the set of all courses taught in the Fall 2009 semester, the Spring 2010 semester, or both

$\{t \mid \exists s \in section\ (t[course\ id\ ] = s[course\ id\ ]) \land s[semester] = \text{"Fall"} \land s[year] = 2009)\} \lor$

$\exists u \in section\ (u[course\ id\ ] = t[course\ id\ ])\} \land u[semester] = \text{"Spring"} \land u[year] = 2010)\}$

Example 5: Find only those course id values for courses that are offered in both the Fall 2009 and Spring 2010 semesters

$\{t \mid \exists s \in section\ (t[course\ id\ ] = s[course\ id\ ]) \land s[semester] = \text{"Fall"} \land s[year] = 2009)\} \land$

$\exists u \in section\ (u[course\ id\ ] = t[course\ id\ ])\} \land u[semester] = \text{"Spring"} \land u[year] = 2010)\}$

Example Queries

Example 5: Find only those course id values for courses that are offered in both the Fall 2009 and Spring 2010 semesters

$$\{t \mid \exists s \in section\ (t[course\ id\ ] = s[course\ id\ ]) \wedge s[semester] = \text{"Fall"} \wedge s[year] = 2009)\} \wedge$$
$$\exists u \in section\ (u[course\ id\ ] = t[course\ id\ ])\} \wedge u[semester] = \text{"Spring"} \wedge u[year] = 2010)\}$$

Example 6: Find all the courses taught in the Fall 2009 semester but not in Spring 2010 semester

$$\{t \mid \exists s \in section\ (t[course\ id\ ] = s[course\ id\ ]) \wedge s[semester] = \text{"Fall"} \wedge s[year] = 2009 \wedge$$
$$\{\neg \exists u \in section\ (u[course\ id\ ] = t[course\ id\ ])\} \wedge u[semester] = \text{"Spring"} \wedge u[year] = 2010)\}$$

2.DOMAIN RELATION CALCULUS

Domain Relational Calculus

- A second form of relational calculus.
- Domain variables.
- Closely related.
- Theoretical basis for QBE (Query By Example).

Domain Relational Calculus

- ★ Expression in DRC:

$$\{< x_1, x_2, \ldots, x_n > \mid P(x_1, x_2, \ldots, x_n)\}$$

- ★ $< x_1, x_2, \ldots, x_n > \in r$
- ★ $x\ (<, \leq, =, \neq, >, \geq)\ y$
- ★ $x\ (<, \leq, =, \neq, >, \geq)\ c$

Domain Relational Calculus – Rules

- An atom is a formula.
- If P_1 is a formula, then so are $\neg P_1$ and (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 $\forall x\ (P_1(x))$.

  $\exists a, b, c\ (P(a, b, c))$ for $\exists a\ (\exists b\ (\exists c\ (P(a, b, c))))$.

Safety of Expressions

- $\{\langle i, n, d, s \rangle \mid \neg(\langle i, n, d, s \rangle \in instructor)\}$.
- Unsafe expressions.

3.7.2 Example Queries

We now give domain-relational-calculus queries for the examples that we considered earlier. Note the similarity of these expressions and the corresponding tuple-relational-calculus expressions.

- Find the loan number, branch name, and amount for loans of over $1200:

$$\{< l, b, a > \mid < l, b, a > \in \textit{loan} \wedge a > 1200\}$$

- Find all loan numbers for loans with an amount greater than $1200:

$$\{< l > \mid \exists\, b, a\, (< l, b, a > \in \textit{loan} \wedge a > 1200)\}$$

Although the second query appears similar to the one that we wrote for the tuple relational calculus, there is an important difference. In the tuple calculus, when we write $\exists\, s$ for some tuple variable s, we bind it immediately to a relation by writing $\exists\, s \in r$. However, when we write $\exists\, b$ in the domain calculus, b refers not to a tuple, but rather to a domain value. Thus, the domain of variable b is unconstrained until the subformula $< l, b, a > \in$ *loan* constrains b to branch names that appear in the *loan* relation. For example,

- Find the names of all customers who have a loan from the Perryridge branch and find the loan amount:

$$\{< c, a > \mid \exists\, l\, (< c, l > \in \mathit{borrower} \\ \wedge \exists\, b\, (< l, b, a > \in \mathit{loan} \wedge b = \text{“Perryridge”}))\}$$

- Find the names of all customers who have a loan, an account, or both at the Perryridge branch:

$$\{< c > \mid \exists l\, (< c, l > \in \mathit{borrower} \\ \wedge \exists b, a\, (< l, b, a > \in \mathit{loan} \wedge b = \text{“Perryridge”})) \\ \vee \exists a\, (< c, a > \in \mathit{depositor} \\ \wedge \exists b, n\, (< a, b, n > \in \mathit{account} \wedge b = \text{“Perryridge”}))\}$$

THANK YOU