TUPLE RELATIONAL CALCULUS

Sibel Adalı

Rensselaer Polytechnic Institute

Tuple Relational Calculus

- A logical language with variables ranging over tuples:

$$\{ T \mid Cond \}$$

  Return all tuples T that satisfy the condition $Cond$.

- $\{ T \mid R(T) \}$: returns all tuples T such that T is a tuple in relation R.

- $\{ T.name \mid FACULTY(T) \, \text{AND} \, T.DeptId =' CS' \}$. returns the values of name field of all faculty tuples with the value 'CS' in their department id field.
  - The variable T is said to be free since it is not bound by a quantifier (for all, exists).
  - The result of this statement is a relation (or a set of tuples) that correspond to all possible ways to satisfy this statement.
  - Find all possible instances of T that make this statement true.

Quantified Statements

- Each variable T ranges over all possible tuples in the universe.
- Variables can be constrained by quantified statements to tuples in a single relation:
  - **Existential Quantifier.** $\exists T \in R(Cond)$ will succeed if $Cond$ succeeds for at least one tuple in T.
  - **Universal Quantifier.** $\forall T \in R(Cond)$ will succeed if $Cond$ succeeds for at all tuples in T.
- Any variable that is not bound by a quantifier is said to be **free**.
- A tuple relational calculus expression may contain at most one free variable.
- The following two expressions are equivalent:

  $\{\, T.name \mid FACULTY(T)\ \text{AND}\ T.DeptId =' CS' \,\}$

  is the same as:

  $\{R \mid \exists T \in FACULTY(T.DeptId =' CS'\ \text{AND}\ R.name = T.name)\}$

Quantified Statements

- $\{\, T.name \mid FACULTY(T)\, \mathrm{AND}\, T.DeptId =' CS' \,\}$ can be read as:

  "Find all tuples T field such that T is a tuple in the FACULTY relation and the value of $DeptId$ field is 'CS'. Return a tuple with a single field name which is equivalent to the name field of one such T tuple".

- $\{R \mid \exists T \in FACULTY(T.DeptId =' CS'\, \mathrm{AND}$ $R.name = T.name)\}$

  can be read as:

  "Find all tuples R such that there exists a tuple T in FACULTY with the $DeptId$ field value 'CS', and the value of the name field of R is equivalent to the name field of this tuple T."

  alternative read:

  "Find all tuples R that can be obtained by copying the name field of **SOME** a tuple in FACULTY with the value 'CS' in its DeptId attribute."

Tuple Relational Calculus Syntax

An atomic query condition is any of the following expressions:

- $R(T)$ where T is a tuple variable and R is a relation name.
- $T.A\, oper\, S.B$ where T, S are tuple variables and A, B are attribute names, *oper* is a comparison operator.
- $T.A\, oper\, const$ where T is a tuple variable, A is an attribute name, *oper* is a comparison operator, and *const* is a constant.

The satisfaction of atomic query conditions is defined in the usual way:

- $R(T)$ evaluates to true if T is a tuple in relation R.

  If T is a variable, then $R(T)$ can evaluate to true by substituting T to one of the tuples in R.
- $T.A\, oper\, const$ evaluates to true if the condition is true.

  If $T.A$ is an unbound variable, then this expression can evaluate to true by all possible substitutions of $T.A$ to some value that satisfy this condition.

Query Conditions

- Any atomic query condition is a query condition.
- If C_1 and C_2 are query conditions, then so are $C_1 \,\mathrm{AND}\, C_2$, $C_1 \,\mathrm{OR}\, C_2$, and $\mathrm{NOT}\, C_1$.
- If C is a query condition, R is a relation name, and T is a tuple variable, then $\forall T \in R(C)$ and $\exists T \in R(C)$ are both query conditions.

A well formed tuple relational calculus query is an expression of the form:

$\{T \,|\, C\}$ where C is a query condition where all the variables except for T are bound to quantified expressions, and T is restricted a finite domain.

Query Condition Examples

- $\{\, T \mid STUDENT(T) \,\text{AND}\, FACULTY(T) \,\}$ will evaluate to true if T is a tuple in both STUDENT and FACULTY relations. However, this is not possible since the schema of the two relations are different. Two tuples can never be identical.
- Correct way to write this statement:

  $$\begin{aligned}&\{T \mid STUDENT(T) \,\text{AND}\, \exists T2 \in FACULTY \\ &\quad (T.Name = T2.Name \,\text{AND}\, T.Address = T2.Address \\ &\quad\ \text{AND}\, T.Password = T2.Password \,\text{AND}\, T.Id = T2.Id)\}.\end{aligned}$$

- What is the result of the following statement?

  $$\{\, T.DeptId \mid STUDENT(T) \,\text{AND}\, T.DeptId =' CS' \,\}$$

  Since there is no $DeptId$ field of T, what is the value of $T.DeptId$? **Answer.** NULL

  How about the following statements?

  $Temp1 = \{\, T \mid T.A = 5 \,\}$
  $Temp2 = \{\, T \mid T.A > 5 \,\}$

Query Conditions

- $Temp2 = \{ T \mid T.A > 5 \}$ is an example of an unbounded expression, the tuple T can be instantiated to infinitely many values. This is not allowed. *All tuples variables should be restricted to the tuples of a specific relation, even if they are not quantified.*
- If a tuple variable T is bound to a relation R, then it only has values for the attributes in R. All other attribute values are null.
- A well formed query will have a single unbounded variable. All other variables will have a quantifier over them.

Examples

- Find the equivalent statement to this:

```
SELECT DISTINCT F.Name, C.CrsCode
FROM FACULTY F, CLASS C
WHERE F.Id = C.InstructorId AND C.Year = 2002
```

$$\{T \mid \exists F \in FACULTY(\exists C \in CLASS \\ (F.Id = C.InstructorId \,\text{AND}\, C.Year = 2002 \,\text{AND} \\ T.Name = F.Name \,\text{AND}\, T.CrsCode = C.CrsCode))\}$$

- Find the equivalent statement to this:

```
SELECT DISTINCT F.Name
FROM FACULTY F
WHERE NOT EXISTS
      (SELECT * FROM CLASS C
       WHERE F.Id = C.InstructorId AND
             C.Year = 2002)
```

$$\{F.Name \mid FACULTY(F) \,\text{AND}\, \textbf{NOT}(\exists C \in CLASS(\\ F.Id = C.InstructorId \,\text{AND}\, C.Year = 2002))\}$$

or carry the “NOT” inside the paranthesis:

$$\{F.Name \mid FACULTY(F) \,\text{AND}\, (\forall C \in CLASS(\\ F.Id <> C.InstructorId \,\text{OR}\, C.Year <> 2002))\}$$

Examples

- Find all students who have taken all the courses required by 'CSCI4380'.

  $$\begin{aligned}\{S.Name \,|\, & STUDENT(S)\,\text{AND} \\ & \forall R \in REQUIRES(\\ & R.CrsCode <>' CSCI4380'\,\text{OR} \\ & (\exists T \in TRANSCRIPT(\\ & \quad T.StudId = S.StudId\,\text{AND} \\ & \quad T.CrsCode = R.PrereqCrsCode\,\text{AND} \\ & \quad T.GradeIN('A','B','C','D'))\}.\end{aligned}$$

- Find all students who have never taken a course from 'Prof. Acorn'. Return the name of the student.

  $$\begin{aligned}\{S.Name \,|\, & STUDENT(S)\,\text{AND}\,\forall C \in CLASS \\ & (\exists F \in FACULTY(F.Id = C.InstructorId\,\text{AND} \\ & (\mathbf{NOT}(F.Namelike'\%Acorn')\,\text{OR} \\ & \mathbf{NOT}(\exists T \in TRANSCRIPT \\ & \qquad (S.Id = T.StudId\,\text{AND} \\ & \qquad C.CrsCode = T.CrsCode\,\text{AND} \\ & \qquad C.Year = T.Year\,\text{AND} \\ & \qquad C.SectionId = T.SectionId)))))\}\end{aligned}$$

Comparing Query Languages

- Relational algebra (RA) and tuple relational calculus (TRC) are equivalent in expressive power.

  In other words, **any query written in RA can be translated to an equivalent TRC expression and vice versa**.
- SQL is more powerfull than the previous two languages due to the **GROUP BY/HAVING** constructs and **aggregrate** functions.
- Extensions of RA and TRC have been proposed to overcome this limitation. They are straightforward extensions.

RA vs. TRC

- **Selection:**

  Algebra: $\sigma_{Cond}(R)$

  Calculus: $\{T \,|\, R(T)\, \text{AND}\, Cond(T)\}$, i.e. replace attributes A in $Cond$ with $T.A$ to obtain $Cond(T)$.

- **Projection:**

  Algebra: $\Pi_{A_1,\ldots,A_k}(R)$

  Calculus: $\{T.A_1,\, \ldots\, T.A_k \,|\, R(T)\}$

- **Cartesian Product:** Given $R(A_1, \ldots, A_n)$ and $S(B_1, \ldots, B_m)$:

  Algebra: $R \times S$

  Calculus: $\{T \,|\, \exists T1 \in R, \exists T2 \in R($
  $T.A_1 = T1.A_1\, \text{AND}\, \ldots\, \text{AND}\, T.A_n = T1.A_n\, \text{AND}$
  $T.B_1 = T2.B_1\, \text{AND}\, \ldots\, \text{AND}\, T.B_m = T2.B_m)\}$

- **Union:**

  Algebra $R \cup S$

  Calculus $\{T \,|\, R(T)\, \text{AND}\, S(T)\}$

- **Set Difference:**

  Algebra $R - S$

  Calculus: $\{T \,|\, R(T)\, \text{AND}\, \forall T1 \in S(T1 <> T)\}$

  where $T <> T1$ is a shorthand for
  $T.A_1 <> T1.A_1\, \text{OR}\, \ldots\, \text{OR}\, T.A_n <> T1.A_n$.

Relational Algebra

Write following relational algebra expressions in tuple relational calculus (results of R_1 and R_2):

$T :=$
$\Pi_{CrsCode,SectionNo,Semester,Year,ClassroomId,InstructorId}(CLASS)$
$T_1 := T[CRS1, SNO1, SEM1, YEAR1, CLR1, INS1]$
$T_2 := T_1[CRS2, SNO2, SEM2, YEAR2, CLR2, INS2]$
$T_3 := T_1 \times T_2$
$T_4 := \sigma_{CRS1 <> CRS2 \text{ OR } SNO1 <> SNO2}(T_3)$
$T_5 := \sigma_{SEM1 = SEM2 \text{ AND } YEAR1 = YEAR2}(T_4)$

$R_1 := \Pi_{INS1,SEM1,YEAR1}(\sigma_{CLR1 = CLR2 \text{ AND } INS1 = INS2}(T_5))$

$R_2 := (\Pi_{ID}(FACULTY)[INS1]) - \Pi_{INS1}(R_1)$

Relational Algebra

R_1: all professors who taught at least two different courses or two different sections of the same course in the same classroom in the same semester and the same year. R_1 contains both the faculty id, and the semester/year information.

$$
\begin{aligned}
R_1 = \{ & T1.InstructorId, T1.Semester, T1.Year \mid \\
& CLASS(T1) \text{ AND} \\
& \exists T2 \in CLASS(T1.InstructorId = T2.InstructorId \text{ AND} \\
& \quad T1.Semester = T2.Semester \text{ AND} \\
& \quad T1.Year = T2.Year \text{ AND} \\
& \quad T1.ClassroomId = T2.ClassroomId \text{ AND} \\
& \quad (T1.CrsCode <> T2.CrsCode \text{ OR} \\
& \quad\; T1.SectionNo <> T2.SectionNo)\}.
\end{aligned}
$$

Relational Algebra

R_2: all professors who never taught two different courses or two different sections of the same course in the same classroom in the same semester and the same year.

$$
\begin{aligned}
R_2 = \{F.Id \mid\, & FACULTY(F)\,\text{AND}\,\forall T1 \in CLASS \\
& (F.Id <> T1.InstructorId\,\text{OR} \\
& (\forall T2 \in CLASS \\
& \quad F.Id <> T2.InstructorId\,\text{OR} \\
& \quad T1.ClassroomId <> T2.ClassroomId\,\text{OR} \\
& \quad T1.Semester <> T2.Semester\,\text{OR} \\
& \quad T1.Year <> T2.Year\,\text{OR} \\
& \quad (T1.CrsCode = T2.CrsCode\,\text{AND} \\
& \quad T1.SectionNo = T2.SectionNo)))\}.
\end{aligned}
$$