Module - I

Relational Databases

Lecture-10

RELATIONAL CALCULUS

RELATIONAL CALCULUS

- Tuple Relational Calculus
- Domain Relational Calculus

Relational Calculus

- Relational Calculus is non procedural or declarative relational query language.
- It has a big influence on the design of commercial query languages such as SQL and especially Query-by-Example (QBE).
- It has two variants:
  1. Tuple Relational Calculus (TRC)
     - Tuples as values. Strong influence on SQL
  2. Domain Relational Calculus (DRC)
     - Variables *range* over field value. Strong influence on QBE

Tuple Relational Calculus (TRC)

- A tuple variable is a variable that takes on tuples of a particular relation schema as values.
- That is, every value assigned to a given tuple variable has the same number and type of fields.

Tuple Relational Calculus (TRC)

- TRC is a nonprocedural query language, where each query is of the form:

  $\{t \mid P(t)\}$

  It is the set of all tuples t such that predicate P is true for t

Where t is a *tuple variable* and $P(t)$ denotes a *formula* that describes t.

- $t[A]$ denotes the value of tuple t on attribute A
- $t \in r$ denotes that tuple t is in relation r
- P is a *formula* similar to that of the predicate calculus

Predicate Calculus Formulas

- Set of attributes and constants
- Set of comparison operators: (e.g., $<, \leq, =, \neq, >, \geq$)
- Set of connectives: **and ($\wedge$), or (v), not ($\neg$)**
- Implication ($\Rightarrow$): $x \Rightarrow y$, if x if true, then y is true

$$x \Rightarrow y \equiv \neg x \vee y$$

- Set of quantifiers:
  - $\exists t \in r(Q(t)) \equiv$ "there exists" a tuple in t in relation r such that predicate $Q(t)$ is true
  - $\forall t \in r(Q(t)) \equiv Q$ is true "for all" tuples t in relation r

Syntax of Tuple Relational Calculus (TRC) Queries

- Let ***Rel*** be a relation name, ***R*** and ***S*** be tuple variables, ***a*** an attribute of ***R*** and ***b*** an attribute of ***S***.
- Let ***op*** denote an operator in the set **{<, ≤, =, ≠, >, ≥}**.
- An atomic formula is one of the following:

1. ***R ∈ Rel***
2. ***R.a op S.b***
3. ***R.a op constant*** **or**
4. **Constant op *R.a***

Semantics of Tuple Relational Calculus (TRC) Queries

- A formula is recursively defined to be one of the following, where **p** and **q** are themselves formulas, and **p(R)** denotes a formula on which the variable **R** appears:

1. Any atomic formula
2. $\neg$ **p**, **p** $\wedge$ **q**, **p** $\vee$ **q** or **p** $\Rightarrow$ **q**
3. $\exists$**R(p(R))**, where **R** is a tuple variable
4. $\forall$**R(p(R))**, where **R** is a tuple variable

Examples of Tuple Relational Calculus (TRC) Queries

1. Find the *ID, name, dept_name, salary* for instructors whose salary is greater than $80,000

- Query:

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

Notice that a relation on schema (*ID*, *name*, *dept_name*, *salary*) is implicitly defined by the query

2. As in the previous query, but output only the *ID* attribute value

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

Notice that a relation on schema (*ID*) is implicitly defined by the query

Examples of Tuple Relational Calculus (TRC) Queries

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

- Query:

$$\{t \mid \exists s \in instructor\ (t[name] = s[name] \wedge \exists u \in department(u[dept\_name] = s[dept\_name] \wedge u[building] = \text{“Watson”}))\}$$

Examples of Tuple Relational Calculus (TRC) Queries

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

- Query:

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

Examples of Tuple Relational Calculus (TRC) Queries

5. Find the set of all courses taught in the Fall 2009 semester, and in the Spring 2010 semester

- Query:

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

Examples of Tuple Relational Calculus (TRC) Queries

6. Find the set of all courses taught in the Fall 2009 semester, but not in the Spring 2010 semester

- Query:

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

Universal Quantification

7. Find all students who have taken all courses offered in the Biology department

- Query:

$$\{t \mid \exists r \in student(t[ID] = r[ID]) \wedge (\forall u \in course(u[dept\_name] = \text{“Biology”} \Rightarrow \exists s \in takes(t[ID] = s[ID] \wedge s[course\_id] = u[course\_id]))\}$$

Safety of Expressions

- It is possible to write tuple calculus expressions that generate infinite relations.
- For example, $\{t \mid \neg t \in r\}$ results in an infinite relation if the domain of any attribute of relation r is infinite.
- To guard against the problem, we restrict the set of allowable expressions to safe expressions.

Safety of Expressions

- An expression $\{t \mid P(t)\}$ in the tuple relational calculus is ***safe*** if every component of t appears in one of the relations, tuples, or constants that appear in P.
  - NOTE: this is more than just a syntax condition.
    - ✓ E.g., $\{t \mid t[A] = 5 \vee \textbf{true}\}$ is not safe - it defines an infinite set with attribute values that do not appear in any relation or tuples or constants in P.

Safety of Expressions

- Consider that query to find all students who have taken all courses offered in the Biology department.

$$\{t \mid \exists r \in student(t[ID] = r[ID]) \wedge (\forall u \in course(u[dept\_name] = \text{“Biology”} \Rightarrow \exists s \in takes(t[ID] = s[ID] \wedge s[course\_id] = u[course\_id]))\}$$

Without the existential quantification on student, the above query would be unsafe if the Biology department has not offered any courses.