Introduction to Relational Calculus

Unit-2

Sources: Reference Book, NPTEL website, Internet

Tuple and domain relational calculus

- It is non procedural query language: described the desired information without giving a specific procedure for obtaining that information.
- A query in tuple relational calculus is expressed as:

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

this represents a set of all tuples **t** such that predicate **P** is true for **t**

  - t is a *tuple variable*, $t[A]$ 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
  - 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 is 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

Qualifying conditions: are made up of well formed formula(wff) of predicate logic. A wff contains a set of atoms and logical connections and quantifiers which connects these atoms.

Atoms:

- Tuple variable membership expression($t \in R$)
- Condition expressions:
  - t[a] op t[b] : t[name]=s[name]
  - t[a] op c : t[name]='amit', t[salary]>1000
  - Where op= $<, \leq, =, \neq, >, \geq$

Not p $= \neg$ p

P and Q = P ^ Q

P or Q = P v Q

Examples

Query: find all the employee whose salary is greater than 1000;

employee(eid, name, salary)

$\{ t \mid P(t) \} = \{ t \mid t \in employee \wedge t[salary] > 1000 \}$ or

$\{ t \mid P(t) \} = \{ t \mid t \in employee \wedge t.salary > 1000 \}$

Example: If want only name attribute from the previous example:

$\{ t \mid P(t) \} = \{ t[name] \mid t \in employee \wedge t[salary] > 1000 \}$

Or alternatively using free and bind variable

$\{ t \mid e\exists(e \in employee \wedge e[salary] > 1000) \wedge t[name]=e[name] \}$

Where t is free variable and e is bounded variable according to first order logic

Tuple RC-Example Queries

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

Instructor(id,name,dept_name,salary)

Department(dept_name,building,budget)

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

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

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

Example Queries

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

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

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

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

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.
- 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*.

Universal Quantification

- Find all students who have taken all courses offered in the Biology department
  - $$\{t \mid \exists\, r \in student\ (t[ID] = r[ID]) \wedge \\ (\forall\, u \in course\ (u[dept\_name]=\text{“Biology”} \Rightarrow \\ \exists\, s \in takes\ (t[ID] = s[ID] \wedge \\ s[course\_id] = u[course\_id]))\}$$
  - Note that without the existential quantification on student, the above query would be unsafe if the Biology department has not offered any courses.

Domain Relational Calculus

Domain Relational Calculus

- A nonprocedural query language equivalent in power to the tuple relational calculus
- Each query is an expression of the form:

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

  - $x_1, x_2, \ldots, x_n$ represent domain variables
  - P represents a formula similar to that of the predicate calculus

Example Queries

- Find the *ID, name, dept_name, salary* for instructors whose salary is greater than $80,000
  - $\{< i, n, d, s> \mid\ < i, n, d, s> \in instructor \wedge s > 80000\}$
- As in the previous query, but output only the *ID* attribute value
  - $\{< i> \mid < i, n, d, s> \in instructor \wedge s > 80000\}$
- Find the names of all instructors whose department is in the Watson building

$$\{< n > \mid\ \exists\, i, d, s\ (< i, n, d, s > \in instructor \\ \wedge \exists\, \text{b, a}\ (< d, b, a> \in department\ \wedge\ b = \text{“Watson”}\ ))\}$$

Example Queries

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

$$\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge s = \text{“Fall”} \wedge y = 2009\ )$$
$$\vee\ \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section\ ] \wedge s = \text{“Spring”} \wedge y = 2010)\}$$

This case can also be written as

$$\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge ((s = \text{“Fall”} \wedge y = 2009\ )\ \vee (s = \text{“Spring”} \wedge y = 2010))\}$$

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

$$\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge s = \text{“Fall”} \wedge y = 2009\ )$$
$$\wedge\ \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section\ ] \wedge s = \text{“Spring”} \wedge y = 2010)\}$$

Safety of Expressions

The expression:

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

is safe if all of the following hold:

1. All values that appear in tuples of the expression are values from *dom* (P) (that is, the values appear either in P or in a tuple of a relation mentioned in P).
2. For every "there exists" subformula of the form $\exists\, x\,(P_1(x))$, the subformula is true if and only if there is a value of x in *dom* (P_1) such that $P_1(x)$ is true.
3. For every "for all" subformula of the form $\forall_x\,(P_1\,(x))$, the subformula is true if and only if $P_1(x)$ is true for all values x from *dom* (P_1).

Universal Quantification

- Find all students who have taken all courses offered in the Biology department
  - $\{< i > \mid \exists\, n, d, tc\ (< i, n, d, tc > \in student\ \wedge$
    $(\forall\, ci, ti, dn, cr\ (< ci, ti, dn, cr > \in course \wedge dn = \text{"Biology"}$
    $\Rightarrow\ \exists\, si, se, y, g\ (<i, ci, si, se, y, g> \in takes\ ))\}$
  - Note that without the existential quantification on student, the above query would be unsafe if the Biology department has not offered any courses.