Chapter 6

Relational Calculus

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

fit@hcmus

fit@hcmus

Content

- Introduction
- Tuple Relational Calculus (TRC)
- Domain Relational Calculus (DRC)

Introduction

- Is the formal query language
- Introduced by Codd in 1972, “Data Base Systems”, Prentice Hall, p33-98

- Properties
  - Nonprocedural language – declarative language
    - Calculus expression specifies *what is to be retrieved* rather than *how to retrieve*
  - One declarative expression to specify a retrieval request
    - There is no description of how to evaluate query
  - A calculus expression may be written in different way
    - The way it is written has no bearing on how a query should be evaluated

fit@hcmus

Introduction

- Categories
  - **Tuple relational calculus – TRC**
    - SQL
  - Domain relational calculus – DRC
    - QBE (Query By Example)
    - DataLog (Database Logic)

fit@hcmus

Content

- Introduction
- **Tuple relational calculus**
- Domain relational calculus

Tuple relational calculus – TRC

- A simple tuple calculus query is of the form

$$\boxed{\{\ t.A \mid P(t)\ \}}$$

- *t* is a tuple variable
  - Its value is any individual tuple from a relation
  - t.A is a value of a tuple *t* at an attribute *A*
- | *(vertical bar) is used to divide the query into two parts:*
  - *P* is a conditional expression involving *t*
    - P(t) has the TRUE or FALSE value depending on *t*
  - The result is the set of all tuples *t* that satisfy *P(t)*

Example 1

- Find employees whose salary is larger than 30000

$$\{\ t \mid \underbrace{EMPLOYEE(t)}_{P(t)} \land \underbrace{t.SALARY > 30000}_{P(t)}\ \}$$

  - EMPLOYEE(t) : TRUE
    - If *t* is an instance of relation EMPLOYEE
  - t.SALARY > 30000 : TRUE
    - If the attribute SALARY of tuple *t* has a value being larger than 30000
- The result is all tuples t which satisfy:
  - $t \in$ EMPLOYEE and t.SALARY > 30000

fit@hcmus

Example 2

- Retrieve the SSN and first name of employees whose salary is larger than 30000

$\{ t.SSN, t.FNAME \mid EMPLOYEE(t) \wedge t.SALARY > 30000 \}$

  - The set of SSNs and first names of employees of tuples *t* such that *t* are instances of EMPLOYEE and their values are larger than 30000 at the attribute SALARY

Example 3

- Find employees (SSN) who work for the department 'Nghien cuu'

$$\{t.SSN \mid EMPLOYEE(t) \wedge \exists s \in DEPARTMENT \wedge s.DNAME = \text{'Nghien cuu'}\}$$

  - Select tuples ***t*** that belong to relation *EMPLOYEE*
  - Compare ***t*** to a certain tuple ***s*** to find employees working for the department *'Nghien cuu'*
  - Use the existential quantifier

$$\boxed{(\exists t)(Q(t))}$$

Existing a tuple t of the relation R such that the expression Q(t) is TRUE → the result of the existential quantifier is TRUE

fit@hcmus

Example 3

- Find employees (SSN) who work for the department 'Nghien cuu'

$$\{\ t.SSN \mid EMPLOYEE(t) \wedge (\exists s)\ (DEPARTMENT(s) \wedge \underbrace{s.DNAME = \text{'Nghien cuu'} \wedge s.DNUMBER = t.DNO\ )}_{Q(s)}\ \}$$

Example 4

- Find employees (FNAME) who work on projects or who have dependents

$$\{\ t.FNAME \mid EMPLOYEE(t) \wedge (\ (\exists s)\ (WORKS\_ON(s) \wedge (t.SSN = s.ESSN)\ ) \vee (\exists u)\ (DEPENDENT(u) \wedge (t.SSN = u.ESSN)\ )\ )\ \}$$

fit@hcmus

Example 5

- Retrieve the FNAME of employees who participate in projects and have dependents

$$\{\ t.FNAME \mid EMPLOYEE(t) \wedge$$

$$(\exists s)\ (WORKS\_ON(s) \wedge (t.SSN = s.ESSN)\ ) \wedge$$

$$(\exists u)\ (DEPENDENT(u) \wedge (t.SSN = u.ESSN)\ )\ \}$$

Example 6

- Find the FNAME of employees who work on projects and have no dependents

$$\{\ t.FNAME \mid EMPLOYEE(t) \wedge$$

$$(\exists s)\ (WORKS\_ON(s) \wedge (t.SSN = s.ESSN)\ ) \wedge$$

$$\neg(\exists u)\ (DEPENDENT(u) \wedge (t.SSN = u.ESSN))\ \}$$

fit@hcmus

Example 7

- For each project in 'TP HCM', find the project number, the department number that controls the project and the FNAME of the manager

$$\{ s.PNUMBER, s.DNUM, t.FNAME \mid PROJECT(s) \wedge EMPLOYEE(t) \wedge (s.PLOCATION = \text{'TP HCM'}) \wedge (\exists u) (DEPARTMENT(u) \wedge (u.DNUMBER = s.DNUM \wedge u.MGRSSN = t.SSN)) \}$$

fit@hcmus

Example 8

- Find employees (SSN) who work on <u>all</u> projects

  - Use the universal quantifier

$$\boxed{(\forall t)\ (Q(t))}$$

If Q is TRUE with all tuples t of relation R, the universal quantifier is TRUE; otherwise FALSE.

fit@hcmus

Example 8a

- Find employees whose salary is highest.

$$\{\ t.SSN,\ t.LNAME,\ t.FNAME \mid EMPLOYEE(t) \wedge$$

$$(\forall e)\ (EMPLOYEE(e)\ (t.Salary >= e.Salary))\ \}$$

Example 8

- Find employees (SSN, FNAME, LNAME) who work on all projects

$$\{ t.SSN, t.LNAME, t.FNAME \mid EMPLOYEE(t) \wedge$$
$$(\forall s)\ (PROJECT(s) \wedge (\exists u)\ (WORKS\_ON(u) \wedge$$
$$u.PNO = s.PNUMBER \wedge$$
$$u.ESSN = t.SSN\ ))\ \}$$

Example 9

- Find employees (SSN, LNAME, FNAME) who work on all projects controlled by the department 4

$$
\begin{aligned}
\{\ & t.SSN, t.LNAME, t.FNAME \mid EMPLOYEE(t) \land \\
& \quad (\forall s)\ (PROJECT(s) \land \\
& \quad\quad s.DNUM = 4 \land (\exists u)\ (WORKS\_ON(u) \land \\
& \quad\quad\quad u.PNO = s.PNUMBER \land \\
& \quad\quad\quad u.ESSN = t.SSN\ ))\ \}
\end{aligned}
$$

cdio 4.0
fit@hcmus

Example 9

- Find employees (SSN, LNAME, FNAME) who work on all projects controlled by the department 4
  - Use the “implies” operator

$$\boxed{P \Rightarrow Q}$$

If P then Q

Example 9

- Find employees (SSN, LNAME, FNAME) who work on all projects controlled by the department 4

$\{ t.SSN, t.LNAME, t.FNAME \mid EMPLOYEE(t) \wedge$

$(\forall s) (PROJECT(s) \wedge$

$(s.DNUM = 4) \Rightarrow ((\exists u) (WORKS\_ON(u) \wedge$

$u.PNO = s.PNUMBER \wedge$

$u.ESSN = t.SSN))) \}$

Example 9 – Solution 2

- Find employees (SSN, LNAME, FNAME) who work on all projects controlled by the department 4

$$\{\ t.SSN,\ t.LNAME,\ t.FNAME \mid EMPLOYEE(t) \wedge$$
$$(\forall s)\ (PROJECT(s) \wedge$$
$$s.DNUM \neq 4 \vee (\exists u)(WORKS\_ON(u) \wedge$$
$$u.PNO = s.PNUMBER \wedge$$
$$u.ESSN = t.SSN\ ))\ \}$$

cdio 4.0
fit@hcmus

Example 10

- a. Find employees whose salary is larger than at least one employee of department 4.

- b. Find employees whose salary is larger than all employees of department 4.

Formal definition

- A general expression is of the form

$$\{ t_1.A_i,\ t_2.A_j,\ \ldots,\ t_n.A_m \mid P(t_1,\ t_2,\ \ldots,\ t_n,\ \ldots,\ t_{n+m}) \}$$

  - $t_1, t_2, \ldots, t_n$ are tuple variables
  - $A_i, A_j, \ldots, A_m$ are attributes of tuples *t*
  - P is a condition or well-formed formula
    - P is made up of predicate calculus <u>atoms</u>

fit@hcmus

Tuple variable

Free variable

$\{ t \mid t \in EMPLOYEE \wedge t.SALARY > 30000 \}$

t is a free variable

Bound variable

$\{ t \mid t \in EMPLOYEE \wedge (\exists s)\ (DEPARTMENT(s) \wedge (s.DNUMBER = t.PNO)) \}$

Free variable

Bound variable

fit@hcmus

Atoms

- (i) $\boxed{t \in R}$
  - t is a tuple variable
  - R is a relation

$t \in$ EMPLOYEE

- (ii) $\boxed{t.A\ \theta\ s.B}$
  - A is an attribute of the tuple variable t
  - B is an attribute of the tuple variable s
  - θ is comparison operators, eg. $<, >, \leq, \geq, \neq, =$

t.SSN = s.ESSN

- (iii) $\boxed{t.A\ \theta\ c}$
  - C is a constant
  - A is an attribute of the tuple variable t
  - θ is comparison operators, eg. $<, >, \leq, \geq, \neq, =$

t.SALARY > 30000

Atoms

- Each of atoms evaluates to either TRUE or FALSE for a specific combination of tuples

- Formula (i) $\boxed{t \in R}$
  - TRUE value if *t* is a tuple of the specified relation *R*
  - FALSE value if *t* does not belong to *R*

R	A	B	C
	α	10	1
	α	20	1

$t1 = <\alpha, 10, 1>$ $t1 \in R$ has the TRUE value

$t2 = <\alpha, 20, 2>$ $t2 \in R$ has the FALSE value

fit@hcmus

Atoms

- Formula (ii) $\boxed{t.A \ \theta \ s.B}$ and (iii) $\boxed{t.A \ \theta \ c}$
  - If the tuple variables are assigned to tuples such that they satisfy the condition, then the atom is TRUE

R	A	B	C
	α	10	1
	α	20	1

If *t* is the tuple <α, 10, 1>

Then $t.B > 5$ has the TRUE value (10 > 5)

fit@hcmus

Rules

- (1) Every atom is formula
- (2) If P is a formula then
  - $\neg P$ is a formula
  - (P) is a formula
- (3) If P_1 and P_2 are formulas then
  - $P_1 \vee P_2$ is a formula
  - $P_1 \wedge P_2$ is a formula
  - $P_1 \Rightarrow P_2$ is a formula

Rules

- (4) If P(t) is a formula then
  - $\forall t \in R\ (P(t))$ is a formula
    - TRUE when *P(t)* is TRUE for all tuples in *R*
    - FALSE when there is one tuple that makes *P(t)* FALSE
  - $\exists t \in R\ (P(t))$ is a formula
    - TRUE when there exists some tuple that makes *P(t)* TRUE
    - FALSE when *P(t)* is FALSE for all tuples *t* in *R*

Rules

- (5) If P is an atom then
  - Tuple variables *t* in *P* are free variables
- (6) Formulas $P=P_1 \wedge P_2$, $P=P_1 \vee P_2$, $P=P_1 \Rightarrow P_2$
  - A variable *t* in *P* is free or bound variable will depends on its role in P_1 and P_2

fit@hcmus

Transform

- (i) $P_1 \wedge P_2 = \neg (\neg P_1 \vee \neg P_2)$
- (ii) $\forall t \in R\ (P(t)) = \neg \exists t \in R\ (\neg P(t))$
- (iii) $\exists t \in R\ (P(t)) = \neg \forall t \in R\ (\neg P(t))$
- (iv) $P \Rightarrow Q = \neg P \vee Q$

Safe expression

- Examine

$$\{ t \mid \neg(EMPLOYEE(t)) \}$$

- Unsafe
  - Many tuples in the universe that are not EMPLOYEE tuples
  - Even though they do not exist in the database
  - The result is infinitely numerous

Safe expression

- Safe expression
  - Guarantee to yield *a finite number of tuples*
- A formula P is called safe expression
  - If its resulting values are from <u>the domain of P</u>
    - The domain of a tuple relational calculus expression: DOM(P)
    - The set of all values
      - Either appear as constant values in P
      - Or exist in any tuple in the relation referenced in P

Safe expression

- Example

  $\{ t \mid EMPLOYEE(t) \wedge t.SALARY > 30000 \}$

  - DOM(EMPLOYEE(t)$\wedge$ t.SALARY > 30000)
  - The set of values
    - Lager than 30000 at the attribute SALARY
    - Other values at the remaining attributes that appear in EMPLOYEE
  - Safe expression

fit@hcmus

Content

- Introduction
- Tuple relational calculus
- **Domain relational calculus**

Domain relational calculus

- An expression of the domain calculus is of the form

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

- $x_1, x_2, \ldots, x_n$ are domain variables
  - Accepting single values from the domain of attributes
- P is a formula of variables $x_1, x_2, \ldots, x_n$
  - P is formed from atoms
- The result
  - The set of values such that when assigned to variables x_i, they make *P* TRUE

Example 1

- Find employees whose salary is larger than 30000

$$\{\ r, s \mid (\exists x)\ (EMPLOYEE(p, q, r, s, t, u, v, x, y, z) \wedge x > 30000\ )\ \}$$

fit@hcmus

Example 3

- Find employees (SSN) who work for the department ‘Nghien cuu’

$$\{\ s \mid (\exists z)\ (\text{EMPLOYEE}(p, q, r, s, t, u, v, x, y, z) \wedge$$
$$(\exists a, b)\ (\text{DEPARTMENT}(a, b, c, d) \wedge$$
$$a = \text{‘Nghien cuu’} \wedge b = z\ ))\ \}$$

Example 10

- Find employees (SSN, LNAME, FNAME) who have no dependents

$$\{\ p, r, s \mid (\exists s)\ (EMPLOYEE(p, q, r, s, t, u, v, x, y, z) \wedge \neg(\exists a)\ (DEPENDENT(a, b, c, d, e) \wedge a = s\ ))\ \}$$

Atoms

- (i) $\boxed{\langle x_1, x_2, \ldots, x_n \rangle \in R}$
  - x_i is a domain variable
  - R is a relation with *n* attributes
- (ii) $\boxed{x\ \theta\ y}$
  - x, y are domain variables
  - Domains of *x* and *y* are identical
  - θ is comparison operators, eg. $<, >, \leq, \geq, \neq, =$
- (iii) $\boxed{x\ \theta\ c}$
  - c is a constant
  - x is a domain variable
  - θ is comparison operators, eg. $<, >, \leq, \geq, \neq, =$

Discussion

- Atoms evaluate to either TRUE or FALSE for a set of values
  - Called the truth values of the atoms

- Rules and transforms are in the similar way to the tuple calculus

Safe expression

- Examine

  $\{ p, r, s \mid \neg \text{EMPLOYEE}(p, q, r, s, t, u, v, x, y, z) \}$

  - Values in the result do not belong to the domain of the expression
  - Unsafe

Safe expression

- Examine

$$\{ x \mid \underbrace{\exists y\,(R(x, y))}_{\text{Formula 1}} \wedge \underbrace{\exists z\,(\neg R(x, z) \wedge P(x, z))}_{\text{Formula 2}} \}$$

  - *R* is a relation with a finite number of values
  - We also have a finite number of values that does not belong to *R*
  - Formula 1: examine values in *R* only
  - Formula 2: could not validate cause we do not know the finite number of values of variable *z*

Safe expresion

- Expression

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

is safe if :

- Values that appear in tuples of the expression must belong to the domain of *P*
- ∃ quantifiers: expression ∃x (Q(x)) is TRUE if
  - Values of *x* belong to DOM(Q) and make *Q(x)* TRUE
- ∀ quantifiers: expression ∀x (Q(x)) is TRUE if
  - *Q(x)* is TRUE for all values of *x* belonging to DOM(Q)

fit@hcmus