

데이터베이스 및 설계

Chap 5. 관계 대수와 관계 해석

#2. Relational Calculus



2014.03.19.

오 병 우

컴퓨터공학과

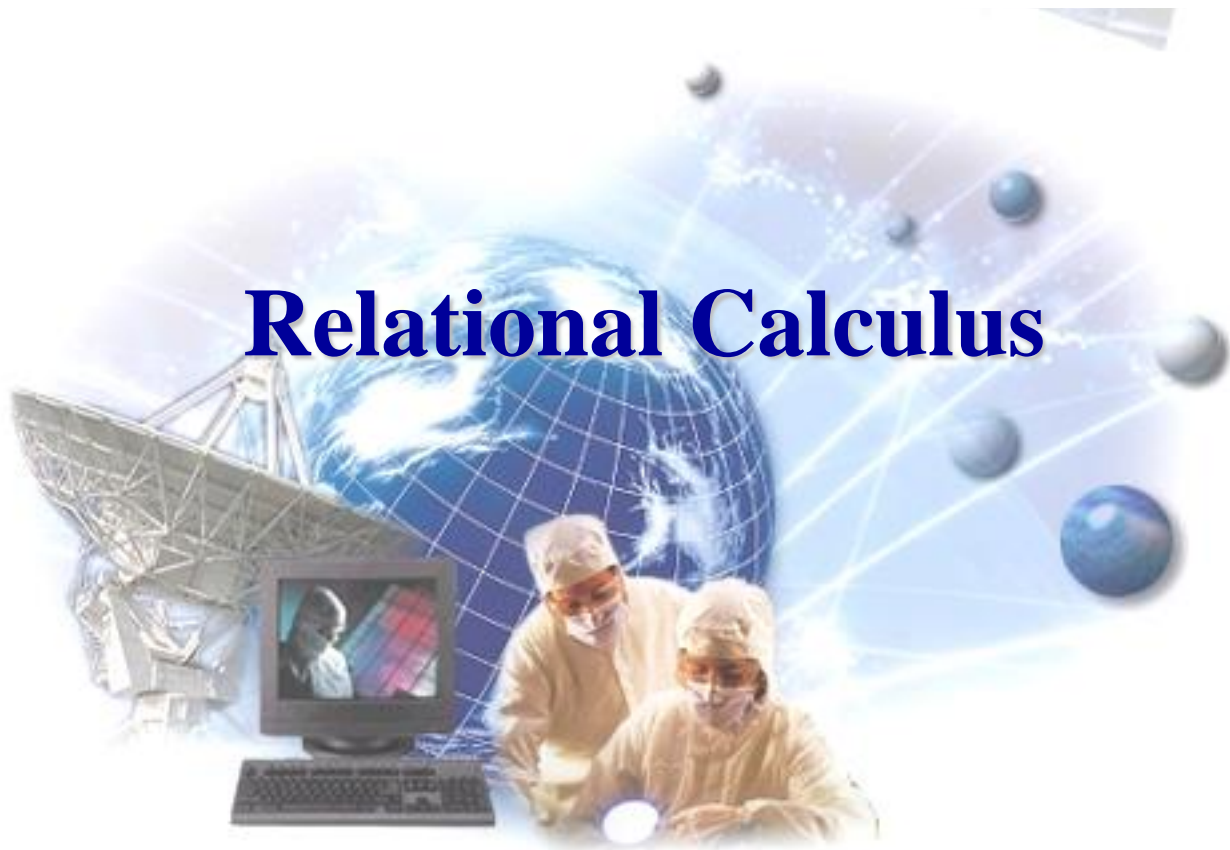
Relational Algebra & Calculus

● Relational Algebra (관계 대수)

- ◆ *Algebra is a type of mathematics in which letters are used to represent possible quantities.*
- ◆ Procedural language (절차 언어) : How

● Relational Calculus (관계 해석)

- ◆ *Calculus is a branch of advanced mathematics which deals with variable quantities.*
- ◆ Nonprocedural language (비절차 언어) : What
- ◆ 월 원하는 지만 기술하면 되므로 사용 편이



Relational Calculus

Introduction of Relational Calculus

● Relational calculus

- ◆ an applied predicate calculus specially tailored to relational databases
- ◆ first-order predicate calculus for query expression

● Relational algebra vs. relational calculus

- ◆ similarity
 - a formal basis for the manipulative part
 - 관계 데이터 모델의 연산 표현 방법
 - precisely equivalent to one another

◆ differences

<u>relational algebra</u>	<u>relational calculus</u>
- a collection of explicit operations (join, union, projection, etc)	-a notation for formulating the definition of the desired relation
-procedural (how)	-descriptive, nonprocedural (what)
-1. join, 2. selection 3. projection	-원하는 정보가 무엇이라는 것만 선언

Introduction

Two kinds of relational calculus

◆ Tuple calculus (TRC: Tuple Relational Calculus)

- Tuple (range) variables: range over tuples
 - 지정된 릴레이션의 한 튜플만을 그 값으로 취할 수 있는 변수

변수가
tuple

◆ Domain calculus (DRC: Domain Relational Calculus)

- domain variables: range over domain elements (= field values)
 - 지정된 애트리뷰트의 도메인의 한 원소 값만을 값으로 취하는 변수
 - QBE(Query-By-Example)

변수가
attribute

Variable (변수)

- ◆ 변하는 수: 저장된 값이 변함
- ◆ 한 번에는 한 개만 저장

Tuple Variables and Qualified Attribute

● Tuple Variables

- ◆ 지정된 릴레이션의 한 튜플만을 그 값으로 취할 수 있는 변수
- ◆ 튜플 변수(tuple variable) 또는 범위변수(range variable): t
- ◆ 범위식 (range formula) : $R(t)$
 - t 는 R 의 튜플 변수
- ◆ $R : t$ 의 범위 릴레이션 (range relation)

● 한정 애트리뷰트(qualified attribute) : $t.A$ 또는 $t[A]$

- 튜플 변수 t 가 나타내는 튜플의 어떤 애트리뷰트 A 의 값
- Student(s)
 - s.Sno

First-order Predicate Calculus

● Predicate (서술, 술어, 서술어)

- ◆ a function whose value is true or false
 - a function that maps object arguments into TRUE or FALSE

```
bool Predicate(int a)
{
    if (a > 0 && a < 10)
        return true;
    else
        return false;
}
```

● first-order predicate calculus

- ◆ forbid variables that represent predicates (i.e., objects only)
- ◆ 문장의 주어가 개별 개체 (소크라테스는 죽는다)
- ◆ Quantifier가 변수에만 적용되고 predicate나 함수에 대해서는 허용되지 않음
 - $(\forall x)P(x)$
- ◆ 서술어, 함수, 변수, 상수, 한정자, 논리적 연결자 등으로 구성된 Symbol로 표현

● second-order 또는 higher order predicate calculus

- ◆ permit variables that represent predicates
- ◆ 주어가 술어(predicate)로 구성
 - 죽는다는 것은 비극이다, $(\forall P)P(x)$

```
Predicate( func1(b) )
```

Tuple Relational Calculus

- 원하는 릴레이션을 tuple calculus expression으로 정의할 수 있는 표기법

- Query : $\{ t \mid P(t) \}$ 으로 표현

- ◆ $P(t)$ 는 tuple variable t 에 대한 formula

- Expressions in the calculus are called formulas

```
bool Predicate(int a)
{
    if (a > 0 && a < 10)
        return true;
    else
        return false;
}
```



```
{ t | Predicate(t) }
= { 1,2,3,4,5,6,7,8,9 }
```

- Answer

- ◆ the set of all tuples t for which the formula $P(t)$ evaluates to TRUE

- 우리가 얻으려는 것은 t 의 집합임, $P(t)$ 는 t 가 될 수 있는 값을 제한함

- Formula

- ◆ Recursively defined (nested formula)

- Start with simple atomic formulas

- Get tuples from relations or making comparisons of values

- Build bigger and better formulas using the logical connectives (\neg , \wedge , \vee , $=$)

Formulas

- Atomic formula, atomic expression, proposition (명제), atom

- ① $R(t)$

t : 튜플 변수, R : t 의 범위 릴레이션

- ② $t.A \theta u.B$

t, u : 튜플 변수, A, B : t 와 u 에 대한 한정 애트리뷰트
 θ : 비교 연산자($=, \neq, <, \leq, >, \geq$)

- ③ $t.A \theta c$

A : 튜플 변수 t 에 대한 한정 애트리뷰트, c : 상수

◆ Atom의 결과는 반드시 참(True) 또는 거짓(False)

- A formula can be:

◆ An atomic formula

◆ $\neg p, p \wedge q, p \vee q$ where p and q are formulas

◆ $(\exists t)P(t)$ where variable t is a tuple variable

◆ $(\forall t)P(t)$ where variable t is a tuple variable

Free and Bound Variables

Quantifiers (정량자)

- ◆ \forall : 전칭 정량자(Universal quantifier) – “for all”
- ◆ \exists : 존재 정량자(Existential quantifier) – “there exists”

Bound Variable (속박 변수)

- ◆ Formula에 $\exists x$ 및 $\forall x$ 를 포함하고 있다면 x 는 bound variable
 - cf.) bound variable이 아닌 것은 free variable
- ◆ 막대(|)의 오른쪽에만 있는 변수는 모두 속박(\forall, \exists)되어야 함

Free Variable (자유 변수)

- ◆ Quantifiers (\forall and \exists)로 한정되지 않는 tuple variable
- ◆ 막대(|)의 왼쪽에 존재
- ◆ 막대(|)의 왼쪽에 있는 free variable이 막대의 오른쪽에서 속박 (\forall, \exists) 되면 안됨

WFF: Well-Formed Formula

정형식(WFF, Well-formed formula)

◆ Atom, Boolean operator (\wedge, \vee, \neg), quantifier (\forall, \exists)가 다음 규칙에 따라 결합된 식

- ① 모든 atomic formula(atom)는 WFF
- ② F가 WFF이면, (F)와 $\neg F$ 도 WFF
- ③ F와 G가 WFF이면, $F \wedge G$ 와 $F \vee G$ 도 WFF
- ④ 튜플 변수 t가 free variable로 사용된 F(t)가 WFF이면, $\forall t(F(t))$ 와 $\exists t(F(t))$ 도 WFF
- ⑤ 위의 규칙만을 반복 적용해서 만들어진 식은 WFF

◆ WFF의 예제

$$s.Sno = 100$$

$$c.Cno \neq e.Cno$$

$$s.Sno = e.Sno \wedge e.Cno \neq c.Cno$$

$$(\exists e)(e.Sno = s.Sno \wedge e.Cno = 'C413')$$

Tuple Calculus Expression

형식

$$\{ t_1.A_1, t_2.A_2, \dots, t_n.A_n \mid F(t_1, \dots, t_n, t_{n+1}, \dots, t_{n+m}) \}$$

◆ t_i : 튜플 변수

– There is an important restriction

- The variable t that appears to the left of ‘|’ must be the only **free variable** in the formula $P(t)$
- 즉, 다른 모든 tuple variable들은 quantifier를 사용한 bound variable 이어야 함

◆ $F(t_1, \dots, t_n, t_{n+1}, \dots, t_{n+m})$: t_i 가 연관된 WFF로 조건을 명세

◆ Target list

- 막대 (|) 왼편에 나온 qualified attribute 들
 - a list of “target items” separated by commas
- 막대 (|) 오른편에 명세된 조건을 만족하는 결과로 추출 됨

Target list

WFF

막대의 왼쪽에 있는 변수는 오른쪽에서 속박(∀, ∃)되면 안됨

example

◆ $\{ s.Sname \mid STUDENT(s) \}$

◆ $\{ s.Sname \mid STUDENT(s) \wedge s.Dept = \text{‘컴퓨터’} \}$

◆ $\{ s.Sname, s.Dept \mid STUDENT(s) \wedge (\exists e)(ENROL(e) \wedge s.Sno = e.Sno \wedge e.Grade = \text{‘A’}) \}$

예제

막대의 왼쪽에
있는 변수는
오른쪽에서
속박(∀, ∃)되
면 안됨

- 과목 C413에서 성적이 A인 학생의 학번을 모두 검색하라

$$\{ e.Sno \mid ENROL(e) \wedge e.Cno='C413' \wedge e.Grade='A' \}$$
- 과목 C413을 등록한 학생의 이름과 학과를 모두 검색하라

$$\{ s.Sname, s.Dept \mid STUDENT(s) \wedge \exists e(ENROL(e) \wedge s.Sno=e.Sno \wedge e.Cno='C413') \}$$
- 모든 과목에 등록한 학생의 이름을 전부 검색하라.

$$\{ s.Sname \mid STUDENT(s) \wedge (\forall c)(\exists e)(COURSE(c) \wedge ENROL(e) \wedge e.Sno=s.Sno \wedge e.Cno=c.Cno) \}$$
- 과목 C413에 등록하지 않은 학생의 이름 전부를 검색하라.

$$\{ s.Sname \mid STUDENT(s) \wedge (\neg \exists e)(ENROL(e) \wedge s.Sno=e.Sno \wedge e.Cno='C413') \}$$

오른쪽에서
연산에
사용되면
속박되어야
함

Domain Relational Calculus

- 원하는 릴레이션을 domain calculus expression으로 표현하는 방법

- Domain variable

- 지정된 애트리뷰트의 도메인의 한 원소만을 값으로 취하는 변수

- 편의상 애트리뷰트 이름 앞에 x 붙임: xSno, xSname, ...

- 범위식을 사용하여 도메인 선언

- STUDENT(xSno, xSname, xDept, xYear)

- Atomic formula

- ① $R(x_1, x_2, \dots, x_n)$

- x_i : 도메인 변수, R : x_i 의 range relation (범위 릴레이션)

- $\langle x_1, x_2, \dots, x_n \rangle$ 에 해당하는 값의 리스트는 릴레이션 R 의 튜플

- ② $x \theta y$

- x, y : 도메인 변수, θ : 비교 연산자(=, \neq , $<$, \leq , $>$, \geq)

- ③ $x \theta c$

- x : 도메인 변수, θ : 비교 연산자, c : x 가 정의된 도메인 값의 상수

- Atomic formula의 실행 결과는 반드시 참(True) 또는 거짓(False)

Domain Calculus Expression

표식

Target list

WFF

$$\{ x_1, x_2, \dots, x_n \mid F(x_1, \dots, x_n, x_{n+1}, \dots, x_{n+m}) \}$$

– x_i : 도메인 변수

– $F(x_1, \dots, x_n, x_{n+1}, \dots, x_{n+m})$: x_i 에 대한 WFF

◆ Target list

– 막대 (|) 왼편에 나온 domain variable 들

– 막대 (|) 오른편에 명세된 조건을 만족하는 domain 값으로 만들어진 tuple

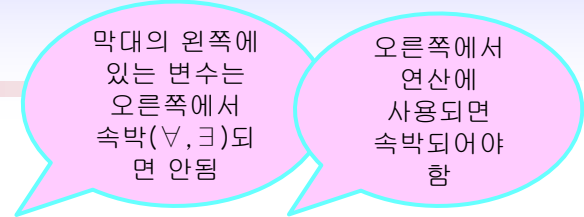
◆ example

① $\{ xSname \mid STUDENT(xSno, xSname, xYear, xDept) \}$

② $\{ xSname \mid (\exists xDept)(STUDENT(xSno, xSname, xYear, xDept) \wedge xDept='컴퓨터') \}$

③ $\{ xSno, xDept \mid STUDENT(xSno, xSname, xYear, xDept) \wedge (\exists xxSno)(\exists xGrade)(ENROL(xxSno, xCno, xGrade, xMidterm, xFinal) \wedge xSno=xxSno \wedge xGrade='A') \}$

예제



- 컴퓨터학과 3,4 학년의 이름을 검색하라.

$$\{ xSname \mid (\exists xYear)(\exists xDept)(STUDENT(xSno, xSname, xYear, xDept) \wedge xYear \geq 3 \wedge xDept='컴퓨터') \}$$

- 과목 C413에서 성적이 A인 학생의 학번을 모두 검색하라

$$\{ xSno \mid (\exists xCno)(\exists xGrade)(ENROL(xSno, xCno, xGrade, xMidterm, xFinal) \wedge xCno='C413' \wedge xGrade='A') \}$$

- 기말 성적이 90점 이상인 학생의 학번과 이름을 검색하라.

$$\{ xSno, xSname \mid (STUDENT(xSno, xSname, xYear, xDept) \wedge (\exists xFinal)(\exists xxSno) (ENROL(xxSno, xCno, xGrade, xMidterm, xFinal) \wedge xSno=xxSno \wedge xFinal \geq 90)) \}$$

- 과목 C324에 등록하지 않은 학생의 이름을 검색하라.

$$\{ xSname \mid (\exists xSno)((STUDENT(xSname, xSno, xYear, xDept) \wedge (\neg \exists xxSno) (\exists xCno) (ENROL(xxSno, xCno, xGrade, xMidterm, xFinal) \wedge xSno=xxSno \wedge xCno='C324')))) \}$$