

튜플 관계해석

관계 해석

- “어떻게 검색할 것인가 (relational algebra)” 보다 “무엇을 검색할 것인가” 만을 기술하는 선언적 표현법 (Declarative)을 사용하는 비절차적 질의어
- SQL을 포함한 많은 상업용 관계 언어들이 관계 해석에 기반을 두고 있음.
- 튜플 관계 해석 (tuple relational calculus)와 도메인 관계 해석 (domain relational calculus)으로 구분됨.

관계 대수와의 차이점

- 관계 해석은 하나의 선언적 (declarative) 해석식으로 검색 질의를 명시하며 비절차적 언어
- 관계 대수 (8가지 연산)에서는 연산들을 순차적으로 사용하므로 절차적인 성질을 가짐
- 두 언어의 표현력 (expressive power)은 동등.

SQL 문장을 받으면 Query Tree (관계 대수식)를 만든다. 관계 해석은 관계 대수 연산자로 바꿀 수 있다.

관계적 완전성 (relationally completeness)

- 어떤 관계 질의어 L이 관계 해석 또는 관계 대수로 표현 가능한 어떤 질의도 표현할 수 있으면 L은 “관계적으로 완전 (relationally complete) 하다.”라고 한다.
- 대부분의 관계 질의어들은 관계적으로 완전하며, 집단 함수 (aggregate functions), 그룹화 (grouping), 순서화(ordering) 등의 연산들을 제공하므로 관계 해석보다 표현력이 강해진다.

튜플 관계해석의 표현과 식

튜플 관계해석의 일반식 형태

{t1.A1, t2.A2, ... , tn.An | COND(t1, t2, ... , tn, tn+1, tn+2, ... , tn+m)}

- n개는 자유변수, m개는 속박변수

존재 정량자와 전체 정량자

- 정량자 (quantifiers)가 식에 사용될 수 있음
- 전체 정량자 (universal quantifier) (\forall) (for all)
- 존재 정량자 (existential quantifier) (\exists) (there exists)

Example)

F1 : d.Dname = 'Research'

F2 : ($\exists t$)(d.Dnumber = t.DNO)

d는 F1과 F2에서 자유롭다.

t는 F2에서 \exists 에 속박.

자유변수는 질의

$\{y | \forall x \ x \in \{1,2,3,4\}\}$ => y는 자유(free)변수.

$\{y | \forall x \ (x < y) \ x \in \{1,2,3,4\}\}$ => y는 공집합

$\{y | \forall x \ (x \leq y) \ x \in \{1,2,3,4\}\}$ => y는 최댓값

$\{y | \forall x \ (x \geq y) \ x \in \{1,2,3,4\}\}$ => y는 최솟값

$\exists x \exists y \ (x, y \in \{1,2,3,4\}, x \geq y)$ => True

$\exists x \forall y \ (x, y \in \{1,2,3,4\}, x \geq y)$ => True. 최댓값이 반드시 존재하는지

$\exists x \forall y \ (x, y \in \{1,2,3,4\}, x \leq y)$ => True. 최솟값이 반드시 존재하는지

프로젝션의 표현

Example)

Employee에 남자인 이름을 찾으시오

- $\{t.fname, t.lname \mid Employee(t) \text{ and } t.sex='M'\}$

Research 부서에 속하는 직원 lname

- $\{e.lname \mid Employee(e) \text{ and } \exists d(Department(d) \text{ and } d.dname='Research') \text{ and } e.dname=d.dname\}$

$e.dno=d.dnumber) \}$

- $\text{Select } e.lname \text{ from Employee } e \text{ where EXISTS (select } * \text{ from Department } d \text{ where } d.name='Research' \text{ and } e.dno=d.dnumber)$

모두가 남자로서만 구성되어 있는 부서

- $\{d.dname \mid \text{Department}(d) \text{ and } \forall e(\text{not } E(e) \text{ or not}(d.dnumber=e.dno) \text{ or } e.sex='M')\}$

최대 급여를 받는 직원의 lname

- $\{e.lname \mid \text{Employee}(e) \text{ and } \forall t(\text{not } E(t) \text{ or } e.salary \geq t.salary)\}$

부서별 최대 급여를 받는 직원의 lname

- $\{e.lname \mid \text{Employee}(e) \text{ and } \forall t(\text{not } E(t) \text{ or not}(e.dno = t.dno) \text{ or } e.salary \geq t.salary)\}$

전체 정량자와 존재 정량자 사이의 변환

- $(\forall x) (P(x)) \equiv \text{not}(\exists x) (\text{not}(P(x)))$
- $(\exists x) (P(x)) \equiv \text{not}(\forall x) (\text{not}(P(x)))$
- $(\forall x) (P(x) \text{ and } Q(x)) \equiv \text{not}(\exists x) (\text{not}(P(x)) \text{ or } \text{not}(Q(x)))$
- $(\forall x) (P(x) \text{ or } Q(x)) \equiv \text{not}(\exists x) (\text{not}(P(x)) \text{ and } \text{not}(Q(x)))$
- $(\exists x) (P(x) \text{ or } Q(x)) \equiv \text{not}(\forall x) (\text{not}(P(x)) \text{ and } \text{not}(Q(x)))$
- $(\exists x) (P(x) \text{ and } Q(x)) \equiv \text{not}(\forall x) (\text{not}(P(x)) \text{ or } \text{not}(Q(x)))$

최대 급여를 받는 직원의 lname

- $\{e.lname \mid \text{Employee}(e) \text{ and not } \exists(t) (E(t) \text{ and } e.dno=t.dno \text{ and } e.salary < t.salary) \}$

$(\forall x) (P(x)) \Rightarrow (\exists x) (P(x))$ True 연역(deduction)

$(\exists x) (P(x)) \Rightarrow (\forall x) (P(x))$ False

$\text{not}(\exists x) (P(x)) \Rightarrow \text{not}(\forall x) (P(x))$ True

$\text{not}(\forall x) (P(x)) \Rightarrow \text{not}(\exists x) (P(x))$ False

5번 부서에 의해 관리되는 모든 프로젝트들에 참여하는 직원들의 이름을 찾아라

- $\{e.lname, e.fname \mid Employee(e) \text{ and } ((\forall x) (\text{not Project}(x)) \text{ or } (\text{not } x.dnum=5) \text{ or } (\exists x \text{ works_on}(w) \text{ and } w.ESSN=e.SSN \text{ and } x.PNUMBER=w.PNO))\}$
- 전체 정량자로부터 존재 정량자로의 변환
- $\{e.lname, e.fname \mid Employee(e) \text{ and } (\text{not } \exists x (\text{Project}(x)) \text{ and } (x.dnum=5) \text{ or } \text{not}(\exists x \text{ works_on}(w) \text{ and } w.ESSN=e.SSN \text{ and } x.PNUMBER=w.PNO))\}$

부양가족이 없는 직원들의 이름을 찾아라

- $\{e.fname, e.lname \mid Employee(e) \text{ and } \text{not } \exists d (\text{Dependent}(d) \text{ and } e.SSN = d.ESSN) \}$

부양가족이 적어도 한 명 있는 관리자들의 이름을 나열하라

- $\{e.fname, e.lname \mid Employee(e) \text{ and } \exists d \exists p (\text{Department}(d) \text{ and } \text{Dependent}(p) \text{ and } e.SSN = d.mgrssn \text{ and } p.essn = e.ssn)\}$

안전식 (safe expression)

- 결과로서 유한개의 튜플들을 생성하는 것이 보장된 식
- 불안전식은 무한개의 튜플들을 생성할 수 있고, 튜플들의 타입이 서로 다를 수 있음
- 불완전한 식의 예제 $\{t \mid \text{not}(\text{EMPLOYEE}(t))\}$
- 가능한 모든 튜플들 중에서 EMPLOYEE가 아닌 모든 튜플들을 생성함
- 이러한 튜플들은 무한개의 튜플들로 구성됨