Introduction to Data Management CSE 344

Lecture 13: Relational Calculus

Announcements

- WQ 4 is out
- HW 4 is out
- Midterm review session in class next Fri (11/4)
- OHs changes next few weeks
  - Check website for most updated info

Big Picture

- Relational data model
  - Instance
  - Schema
  - Query language
    - SQL
    - Relational algebra
    - Relational calculus
    - Datalog
- Query processing
  - Logical & physical plans
  - Indexes
  - Cost estimation
  - Query optimization

Why bother with another QL?

- SQL and RA are good for query planning
  - They are not good for *formal reasoning*
  - How do you show that two SQL queries are equivalent / non-equivalent?
  - Two RA plans?
- RC was the first language proposed with the relational model (Codd)
- Influenced the design of datalog as we will see

Relational Calculus

- Aka *predicate calculus* or *first order logic*
  - 311 anyone?

- TRC = Tuple Relational Calculus
  - See book

- DRC = Domain Relational Calculus
  - We study only this one
  - Also see *Query Language Primer* on course website

Relational Calculus

Query Q:

$$Q(x1, \ldots, xk) = P$$

Relational predicate P is a formula given by this grammar:

$$P ::= atom \mid P \wedge P \mid P \vee P \mid P \Rightarrow P \mid not(P) \mid \forall x.P \mid \exists x.P$$

Atomic predicate is either a relational or interpreted predicate:

$$atom ::= R(x1, \ldots, xk) \mid x = y \mid x > k \mid \ldots$$

R(x,y) means (x,y) is in R

Actor(pid,fName,lName)
Casts(pid,mid)
Movie(mid,title,year)

Relational Calculus

Query Q:

$Q(x1, \ldots, xk) = P$

Relational predicate P is a formula given by this grammar:

$P ::= \text{atom} \mid P \wedge P \mid P \vee P \mid P \Rightarrow P \mid \text{not}(P) \mid \forall x.P \mid \exists x.P$

Atomic predicate is either a relational or interpreted predicate:

$\text{atom} ::= R(x1, \ldots, xk) \mid x = y \mid x > k \mid \ldots$

R(x,y) means (x,y) is in R

Example: find the first/last names of actors who acted in 1940

$Q(f,l) = \exists x.\ \exists y.\ \exists z.\ (\text{Actor}(z,f,l) \wedge \text{Casts}(z,x) \wedge \text{Movie}(x,y,1940))$

What does this query return ?

$Q(f,l) = \exists z.\ (\text{Actor}(z,f,l) \wedge \forall x.(\text{Casts}(z,x) \Rightarrow \exists y.\text{Movie}(x,y,1940)))$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

Important Observation

Find all bars that serve all beers that Fred likes

$$A(x) = \forall y.\ Likes(\text{"Fred"}, y) \Rightarrow Serves(x,y)$$

- Note: $P \Rightarrow Q$ (read P implies Q) is the same as (not P) $\vee$ Q

  In this query: If Fred likes a beer the bar must serve it ($P \Rightarrow Q$)
  In other words: Either Fred does not like the beer (not P) OR the bar serves that beer (Q).

$$A(x) = \forall y.\ not(Likes(\text{"Fred"}, y)) \vee Serves(x,y)$$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

More Examples

Average Joe

Find drinkers that frequent some bar that serves some beer they like.

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

More Examples

Average Joe

Find drinkers that frequent <u>some</u> bar that serves <u>some</u> beer they like.

$$\boxed{Q(x) = \exists y.\ \exists z.\ \text{Frequents}(x,\ y) \wedge \text{Serves}(y,z) \wedge \text{Likes}(x,z)}$$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

More Examples

Average Joe

Find drinkers that frequent some bar that serves some beer they like.

$$Q(x) = \exists y.\ \exists z.\ \text{Frequents}(x, y) \wedge \text{Serves}(y,z) \wedge \text{Likes}(x,z)$$

Prudent Peter

Find drinkers that frequent only bars that serves some beer they like.

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

More Examples

Average Joe

Find drinkers that frequent <u>some</u> bar that serves <u>some</u> beer they like.

$$Q(x) = \exists y.\ \exists z.\ \text{Frequents}(x,\ y) \wedge \text{Serves}(y,z) \wedge \text{Likes}(x,z)$$

Prudent Peter

Find drinkers that frequent <u>only</u> bars that serves <u>some</u> beer they like.

$$Q(x) = \forall y.\ \text{Frequents}(x,\ y) \Rightarrow (\exists z.\ \text{Serves}(y,z) \wedge \text{Likes}(x,z))$$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

More Examples

Average Joe

Find drinkers that frequent <u>some</u> bar that serves <u>some</u> beer they like.

$$Q(x) = \exists y.\ \exists z.\ \text{Frequents}(x,\ y) \wedge \text{Serves}(y,z) \wedge \text{Likes}(x,z)$$

Prudent Peter

Find drinkers that frequent <u>only</u> bars that serves <u>some</u> beer they like.

$$Q(x) = \forall y.\ \text{Frequents}(x,\ y) \Rightarrow (\exists z.\ \text{Serves}(y,z) \wedge \text{Likes}(x,z))$$

Cautious Carl

Find drinkers that frequent <u>some</u> bar that serves <u>only</u> beers they like.

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

More Examples

Average Joe

Find drinkers that frequent some bar that serves some beer they like.

$$Q(x) = \exists y.\ \exists z.\ Frequents(x, y) \wedge Serves(y,z) \wedge Likes(x,z)$$

Prudent Peter

Find drinkers that frequent only bars that serves some beer they like.

$$Q(x) = \forall y.\ Frequents(x, y) \Rightarrow (\exists z.\ Serves(y,z) \wedge Likes(x,z))$$

Cautious Carl

Find drinkers that frequent some bar that serves only beers they like.

$$Q(x) = \exists y.\ Frequents(x, y) \wedge \forall z.(Serves(y,z) \Rightarrow Likes(x,z))$$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

More Examples

Average Joe

Find drinkers that frequent some bar that serves some beer they like.

$$Q(x) = \exists y.\ \exists z.\ \mathrm{Frequents}(x,\ y) \land \mathrm{Serves}(y,z) \land \mathrm{Likes}(x,z)$$

Prudent Peter

Find drinkers that frequent only bars that serves some beer they like.

$$Q(x) = \forall y.\ \mathrm{Frequents}(x,\ y) \Rightarrow (\exists z.\ \mathrm{Serves}(y,z) \land \mathrm{Likes}(x,z))$$

Cautious Carl

Find drinkers that frequent some bar that serves only beers they like.

$$Q(x) = \exists y.\ \mathrm{Frequents}(x,\ y) \land \forall z.(\mathrm{Serves}(y,z) \Rightarrow \mathrm{Likes}(x,z))$$

Paranoid Paul

Find drinkers that frequent only bars that serves only beer they like.

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

More Examples

Average Joe

Find drinkers that frequent <u>some</u> bar that serves <u>some</u> beer they like.

$$Q(x) = \exists y.\ \exists z.\ \text{Frequents}(x,\ y) \wedge \text{Serves}(y,z) \wedge \text{Likes}(x,z)$$

Prudent Peter

Find drinkers that frequent <u>only</u> bars that serves <u>some</u> beer they like.

$$Q(x) = \forall y.\ \text{Frequents}(x,\ y) \Rightarrow (\exists z.\ \text{Serves}(y,z) \wedge \text{Likes}(x,z))$$

Cautious Carl

Find drinkers that frequent <u>some</u> bar that serves <u>only</u> beers they like.

$$Q(x) = \exists y.\ \text{Frequents}(x,\ y) \wedge \forall z.(\text{Serves}(y,z) \Rightarrow \text{Likes}(x,z))$$

Paranoid Paul

Find drinkers that frequent <u>only</u> bars that serves <u>only</u> beer they like.

$$Q(x) = \forall y.\ \text{Frequents}(x,\ y) \Rightarrow \forall z.(\text{Serves}(y,z) \Rightarrow \text{Likes}(x,z))$$

Remember your logical equivalences!

- $A \Rightarrow B$ = not(A) $\vee$ B
- not(A $\wedge$ B) = not(A) $\vee$ not(B)
- not(A $\vee$ B) = not(A) $\wedge$ not(B)
- $\forall$x. P(x) = not($\exists$x. not(P(x)))

- Example:
  - $\forall$z. Serves(y,z) $\Rightarrow$ Likes(x,z)
  - $\forall$z. not(Serves(y,z)) $\vee$ Likes(x,z)
  - not ($\exists$z. Serves(y,z) $\wedge$ not(Likes(x,z))

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

Domain Independent Relational Calculus

- An *unsafe* RC query, aka *domain dependent,* returns an answer that does not depend just on the relations, but on the entire domain of possible values

Make sure x is a beer

$\boxed{A1(x) = \text{not Likes}(\text{"Fred"}, x)}$

$A1(x) = \exists y \text{ Serves}(y,x) \land \text{not Likes}(\text{"Fred"}, x)$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

Domain Independent Relational Calculus

- An *unsafe* RC query, aka *domain dependent,* returns an answer that does not depend just on the relations, but on the entire domain of possible values

Make sure x is a beer

$$A1(x) = \text{not Likes}(\text{"Fred"}, x)$$

$$A1(x) = \exists y\ \text{Serves}(y,x) \land \text{not Likes}(\text{"Fred"}, x)$$

$$A2(x,y) = \text{Likes}(\text{"Fred"}, x) \lor \text{Serves}(\text{"Bar"}, y)$$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

Domain Independent Relational Calculus

- An *unsafe* RC query, aka *domain dependent,* returns an answer that does not depend just on the relations, but on the entire domain of possible values

Make sure x is a beer

$\boxed{A1(x) = \text{not } Likes(\text{"Fred"}, x)}$

$A1(x) = \exists y\ Serves(y,x) \land \text{not } Likes(\text{"Fred"}, x)$

$\boxed{A2(x,y) = Likes(\text{"Fred"}, x) \lor Serves(\text{"Bar"}, y)}$

Same here

$A2(x,y) = \exists u\ Serves(u,x) \land \exists w\ Serves(w,y) \land [Likes(\text{"Fred"}, x) \lor Serves(\text{"Bar"}, y)$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

Domain Independent Relational Calculus

- An *unsafe* RC query, aka *domain dependent,* returns an answer that does not depend just on the relations, but on the entire domain of possible values

$\boxed{A1(x) = \text{not Likes}(\text{"Fred"}, x)}$

Make sure x is a beer

$A1(x) = \exists y\ \text{Serves}(y,x) \wedge \text{not Likes}(\text{"Fred"}, x)$

$\boxed{A2(x,y) = \text{Likes}(\text{"Fred"}, x) \vee \text{Serves}(\text{"Bar"}, y)}$

Same here

$A2(x,y) = \exists u\ \text{Serves}(u,x) \wedge \exists w\ \text{Serves}(w,y) \wedge [\text{Likes}(\text{"Fred"}, x) \vee \text{Serves}(\text{"Bar"}, y)$

$\boxed{A3(x) = \forall y.\ \text{Serves}(x,y)}$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

Domain Independent Relational Calculus

- An *<u>unsafe</u>* RC query, aka *<u>domain dependent,</u>* returns an answer that does not depend just on the relations, but on the entire domain of possible values

Make sure x is a beer

$\boxed{A1(x) = \text{not Likes}(\text{"Fred"}, x)}$

$A1(x) = \exists y\ \text{Serves}(y,x) \wedge \text{not Likes}(\text{"Fred"}, x)$

$\boxed{A2(x,y) = \text{Likes}(\text{"Fred"}, x) \vee \text{Serves}(\text{"Bar"}, y)}$

Same here

$A2(x,y) = \exists u\ \text{Serves}(u,x) \wedge \exists w\ \text{Serves}(w,y) \wedge [\text{Likes}(\text{"Fred"}, x) \vee \text{Serves}(\text{"Bar"}, y)$

Likewise

$\boxed{A3(x) = \forall y.\ \text{Serves}(x,y)}$

$A3(x) = \exists u.\text{Serves}(x,u) \wedge \forall y. \exists z.\text{Serves}(z,y) \rightarrow \text{Serves}(x,y)$

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

Domain Independent Relational Calculus

- An *unsafe* RC query, aka *domain dependent,* returns an answer that does not depend just on the relations, but on the entire domain of possible values

Make sure x is a beer

$\boxed{A1(x) = \text{not Likes}(\text{"Fred"}, x)}$

$A1(x) = \exists y\ \text{Serves}(y,x) \wedge \text{not Likes}(\text{"Fred"}, x)$

$\boxed{A2(x,y) = \text{Likes}(\text{"Fred"}, x) \vee \text{Serves}(\text{"Bar"}, y)}$

Same here

$A2(x,y) = \exists u\ \text{Serves}(u,x) \wedge \exists w\ \text{Serves}(w,y) \wedge [\text{Likes}(\text{"Fred"}, x) \vee \text{Serves}(\text{"Bar"}, y)$

Likewise

$\boxed{A3(x) = \forall y.\ \text{Serves}(x,y)}$

$A3(x) = \exists u.\text{Serves}(x,u) \wedge \forall y.\ \exists z.\text{Serves}(z,y) \rightarrow \text{Serves}(x,y)$

Lesson: make sure your RC queries are domain independent