Relational Calculus

Chapter 4, Part B

1

Relational Calculus

- Comes in two flavours: *Tuple relational calculus* (TRC) and *Domain relational calculus* (DRC).
- Calculus has *variables*, *constants*, *comparison ops*, *logical connectives* and *quantifiers*.
  - *TRC*: Variables range over (i.e., get bound to) *tuples*.
  - *DRC*: Variables range over *domain elements* (= field values).
  - Both TRC and DRC are simple subsets of first-order logic.
- Expressions in the calculus are called *formulas*. An answer tuple is essentially an assignment of constants to variables that make the formula evaluate to *true*.

relational algebra is procedural

relational calculus is declarative (just say what you want)

relational calculus: tuple calculus and domain calculus

TRC: tuple, sql from clause, select * from student s where s.sid =1, s.sid is the variable range over tuple s

DRC: attribute

2

Domain Relational Calculus

- *Query* has the form:

$$\left\{\langle x1, x2, ..., xn \rangle \mid p\left(\langle x1, x2, ..., xn \rangle\right)\right\}$$

the set we query

- *Answer* includes all tuples $\langle x1, x2, ..., xn \rangle$ that make the *formula* $p\left(\langle x1, x2, ..., xn \rangle\right)$ be *true*.
- *Formula* is recursively defined, starting with simple *atomic formulas* (getting tuples from relations or making comparisons of values), and building bigger and better formulas using the *logical connectives*.

3

DRC Formulas

- *Atomic formula:*
  - $\langle x1, x2, ..., xn \rangle \in Rname$, or X *op* Y, or X *op* constant
  - *op* is one of $<, >, =, \leq, \geq, \neq$
- *Formula:*
  - an atomic formula, or
  - $\neg p,\ p \wedge q,\ p \vee q$, where p and q are formulas, or
  - $\exists X\,(p(X))$, where variable X is *free* in p(X), or
  - $\forall X\,(p(X))$, where variable X is *free* in p(X)
- The use of quantifiers $\exists X$ and $\forall X$ is said to <u>*bind*</u> X.
  - A variable that is not bound is free.

Database Management Systems, R. Ramakrishnan 4

4

Free and Bound Variables

- The use of quantifiers $\exists X$ and $\forall X$ in a formula is said to <u>*bind*</u> X. bound means it has a scope
  - A variable that is not bound is <u>free</u>.
- Let us revisit the definition of a query:

$$\{\langle x1, x2, ..., xn \rangle \mid p(\langle x1, x2, ..., xn \rangle)\}$$

only free variables left side

- There is an important restriction: the variables x1, ..., xn that appear to the left of \`|' must be the *only* free variables in the formula p(...).

Database Management Systems, R. Ramakrishnan 5

5

Find all sailors with a rating above 7

<I,m,p,q> | <I,m,p,q> $ sailors ^ p > 7 is equivalent to below

$$\{\langle I, N, T, A \rangle \mid \langle I, N, T, A \rangle \in Sailors \wedge T > 7\}$$

such that

(variables are just placeholders, you can choose any names)

Equivalent in TRC:
{ t | t $ Sailors ^ t.rating > 7}
{ t.sname Sailors ^ t.rating > 7}

- The condition $\langle I, N, T, A \rangle \in Sailors$ ensures that the domain variables *I*, *N*, *T* and *A* are bound to fields of the same Sailors tuple.
- The term $\langle I, N, T, A \rangle$ to the left of \`|' (which should be read as *such that*) says that every tuple $\langle I, N, T, A \rangle$ that satisfies *T*>7 is in the answer.
- Modify this query to answer:
  - Find sailors who are older than 18 or have a rating under 9, and are called 'Joe'.

Database Management Systems, R. Ramakrishnan 6

6

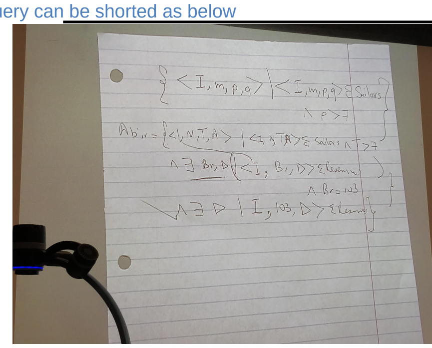
Find sailors rated > 7 who've reserved boat #103

$$\{\langle I,N,T,A\rangle \mid \langle I,N,T,A\rangle \in Sailors \wedge T>7 \wedge \exists\, Ir,Br,D\,\big(\langle Ir,Br,D\rangle \in Reserves \wedge Ir=I \wedge Br=103\big)\}$$

- We have used $\exists\, Ir,\, Br,\, D\; (\ldots)$ as a shorthand for $\exists\, Ir\,(\,\exists\, Br\,(\,\exists\, D(\ldots)))$
- Note the use of $\exists$ to find a tuple in Reserves that `joins with' the Sailors tuple under consideration.

Database Management Systems, R. Ramakrishnan 7

relational calculus
projection: the variable we are not projected should be bound
if the question is find sailor name instead of sailor
then the formula should be <N> there exists I,T,A such that
selection: br = 103
query can be shorted as below



7

Find sailors rated > 7 who've reserved a red boat

$$\{\langle I,N,T,A\rangle \mid \langle I,N,T,A\rangle \in Sailors \wedge T>7 \wedge \exists\, Ir,Br,D\,\big(\langle Ir,Br,D\rangle \in Reserves \wedge Ir=I \wedge \exists\, B,BN,C\,\big(\langle B,BN,C\rangle \in Boats \wedge B=Br \wedge C='red'\big)\big)\}$$

TRC equivalent:

{ s.sname | s $ sailors ^ s.rating > 7 ^
there exists r (r $ reserves ^ r.sid= s.sid ^
there exists b (b $ boats ^ b.bid = r.bid ^ b.color = "red"))

- Observe how the parentheses control the scope of each quantifier's binding.
- This may look cumbersome, but with a good user interface, it is very intuitive. ~~(Wait for QBE!)~~

Database Management Systems, R. Ramakrishnan 8

8

Find sailors who've reserved all boats

To make it simple

$$\{\langle I,N,T,A\rangle \mid \langle I,N,T,A\rangle \in Sailors \wedge \forall\, B,BN,C\,\big(\neg\big(\langle B,BN,C\rangle \in Boats\big) \vee \big(\exists\, \cancel{Ir},\cancel{Br},D\,\big(\langle \cancel{I},\cancel{Br},\cancel{D}\rangle \in Reserves \wedge \cancel{I=Ir \wedge Br=B}\big)\big)\big)\}$$

<I, B , D>

- Find all sailors I such that for each 3-tuple $\langle B,BN,C\rangle$ either it is not a tuple in Boats or there is a tuple in Reserves showing that sailor I has reserved it.

Database Management Systems, R. Ramakrishnan 9

9

Find sailors who've reserved all boats (again!)

$$\{\langle I,N,T,A\rangle \mid \langle I,N,T,A\rangle \in Sailors \wedge \forall \langle B,BN,C\rangle \in Boats \left(\exists \langle Ir,Br,D\rangle \in \mathrm{Re}serves\left(I=Ir \wedge Br=B\right)\right)\}$$

- Simpler notation, same query. (Much clearer!)
- To find sailors who've reserved all red boats:

$$\ldots\ldots \left(C \neq 'red' \vee \exists \langle Ir,Br,D\rangle \in \mathrm{Re}serves\left(I=Ir \wedge Br=B\right)\right)\}$$

Unsafe Queries, Expressive Power

- It is possible to write syntactically correct calculus queries that have an infinite number of answers! Such queries are called *unsafe*.
  - e.g., $\{S \mid \neg\left(S \in Sailors\right)\}$
- It is known that every query that can be expressed in relational algebra can be expressed as a safe query in DRC / TRC; the converse is also true.
- *Relational Completeness*: Query language (e.g., SQL) can express every query that is expressible in relational algebra/calculus.

Summary

- Relational calculus is non-operational, and users define queries in terms of what they want, not in terms of how to compute it. (Declarativeness.)
- Algebra and safe calculus have same expressive power, leading to the notion of relational completeness.