Universidade Federal da Fronteira Sul - UFFS
Computer Science - Chapecó
Database I
Prof.: Denio Duarte

Fast Guide to (Tuple) Relational Calculus

In the examples of this guide, we use the following database schema (extract from Ramakrishnan's book):
`sailor(sid,sname,rating,age)`   `boat(bid,bname,color)`   `reserve(sid,bid,day)`
Catalog:
`sailor`: all attributes are mandatory. `sid` is `sailor`'s pk
`boat`: all attributes are mandatory. `bid` is `boat`'s pk
`reserve`: all attributes are mandatory. `sid`, `sid` and `day` are `reserve`'s pk. `sid` is a fk from `sailor` and `bid` is a fk from `boat`

Introduction

Relational calculus (RC) is high-level, first-order logic description. It is a formal definition of what you want from the database (while relational algebra is used to describe the query optimizer, RC is basis for SQL).
There are two relational calculi: tuple relational calculus (TRC) and domain relational calculus (DRC). In this text, we study TRC.

TRC works with variables ranging over tuples: $\{t \mid P(t)\}$, where t is a tuple variable (it stores the query answer) and $P(t)$ is the predicate (or formula) over t that must be satisfied.
Example: *find all sailor with rating over 7.* TRC: $\{s \mid s \in Sailor \wedge s.rating > 7\}$, where s is the tuple variable, $s \in Sailor$ says that s is the same type of $Sailor$ schema and $s.rating > 7$ is the selection condition.
TRC formula - is recursively defined:

- Atomic formulas get tuples from relations or compare values
- Formulas built from other formulas using logical operators

An atomic formula can be:

- $r \in S$ (where r is a tuple variable and S is a relation)
- $r.a$ op $s.b$ (where a is an attribute from tuple r, b is an attribute from tuple s and op is one of the operators $=, \neq, >, <, \leq, \geq$)
- $r.a$ op k or k op $r.a$ (where k is a constant)

A (well formed) formula (wff) can be:

- an atomic formula
- $\neg p$, $p \wedge q$ or $p \vee q$ (where p and q are formulas)
- If $P(r)$ is a formula, so is $\exists r P(r)$ (r is a tuple variable)
- If $P(r)$ is a formula, so $\forall r P(r)$ (r is a tuple variable)

Restrictions:

- The quantifiers $\exists t$ and $\forall t$ in a formula bind t in the formula
- A tuple variable that is not **bound** is **free**.
- In a TRC expression $E = \{t \mid P(t)\}$, t must be the only free variable (not bound by a quantifier) in E
- Unsafe expression (queries): an expression like $\{s \mid s \notin Sailor\}$ is considered unsafe since its answer is infinite. We use the same reasoning for $\{t \mid t > 10\}$

Reminders:

- DeMorgan law: $p \wedge q \equiv \neg(\neg p \vee \neg q)$
- Implication: $p \rightarrow q \equiv \neg p \vee q$
- Double negation: $\forall s \in R(P(s)) \equiv \neg\exists s \in R(\neg P(s))$ (*every human is mortal : no human is immortal*)

TRC semantics - An expression in TRC is evaluated as true is one the following conditions is satisfied:

- F is an atomic formula $s \in R$ and s is an instance of a tuple from R
- F is an atomic formula $s.a\ op\ t.b$, $s.a\ op\ k$ or $k\ op\ s.a$, the tuples associated to s and r satisfy the condition of F
- F is a formula $\neg p$ and p is not true; or $p \wedge q$ and p and q are true; or $p \vee q$ and p or q is true; or $p \rightarrow q$ and q is true if p is true
- F is a formula $\exists r(P(r))$, there is some free tuple in $P(r)$ that one (or more) tuple(s) from a relation satisfies $P(r)$
- F is a formula $\forall r(P(r))$, there is some free tuple in $P(r)$ that all tuples from a relation satisfy $P(r)$

Some examples:

- Find sid and name of sailors who've reserved a red boat:
  $\{t \mid \exists s \in sailor, \exists r \in reserve(s.sid = r.sid \wedge \exists b \in boat(r.bid = b.bid \wedge b.color = red \wedge t.sid = s.sid \wedge t.sname = s.sname))\}$
- Find sailors rated > 7 who have reserved boat 103:
  $\{t \mid \exists s \in sailor, \exists r \in reserve(s.sid = r.sid \wedge s.rating > 7 \wedge \exists b \in boat(r.bid = b.bid \wedge b.bid = 103 \wedge t.sid = s.sid \wedge t.sname = s.sname))\}$
- Find sailors who've reserved all boats:
  $\{s \mid s \in sailor, \forall b \in boat(\exists r \in reserve(s.sid = r.sid \wedge r.bid = b.bid))\}$
  $\{t \mid \exists s \in sailor, \forall b \in boat(\exists r \in reserve(s.sid = r.sid \wedge r.bid = b.bid \wedge t.sname = s.sname))\}$
- Find sailors who've reserved all red boats:
  $\{t \mid \exists s \in sailor, \forall b \in boat(b.color = red \rightarrow (\exists r \in reserve(s.sid = r.sid \wedge r.bid = b.bid \wedge t.sname = s.sname)))\}$
  by using the equivalence implication law: $p \rightarrow q \equiv \neg p \vee q$
  $\{t \mid \exists s \in sailor, \forall b \in boat(b.color \neq red \vee (\exists r \in reserve(s.sid = r.sid \wedge r.bid = b.bid \wedge t.sname = s.sname)))\}$

- Find sailors who've reserved red and green boats:
  $\{t \mid \exists s \in sailor, \exists r_1 \in reserve(s.sid = r_1.sid \wedge \exists b_1 \in boat(r_1.bid = b_1.bid \wedge b.color = red \wedge \exists r_2 \in reserve(s.sid = r_2.sid \wedge \exists b_2 \in boat(r_2.bid = b_2.bid \wedge b.color = green \wedge t.sname = s.sname))))\}$

- Find sailor who've reserved at least two boats:
  $\{t \mid \exists s \in sailor, \exists r_1 \in reserve, \exists r_2 \in reserve(s.sid = r_1.sid \wedge s.sid = r_2.sid \wedge r_1.bid \neq r_2.bid \wedge t.sname = s.sname)\}$