RELATIONAL CALCULUS

- Relational calculus is an alternative to relational algebra.
- Relational calculus is nonprocedural, or *declarative*,
  - it allows us to describe the set of answers without being explicit about how they should be computed.
-

- Two Variants of the calculus
  - *Tuple relational calculus* (TRC)
  - *Domain relational calculus* (DRC).
- *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.
- Calculus has *variables, constants, comparison ops*, *logical connectives* and *quantifiers*.

Tuple Relational Calculus

- A **tuple variable** is a variable that takes on tuples of a particular relation schema as values.
- Tuple variable is a variable that 'ranges over' a named relation:
  - i.e., variable whose only permitted values are tuples of the relation
- Every value assigned to a given tuple variable has the same number and type of fields.
- A tuple relational calculus query has the form $\{ T \mid p(T) \}$,
  - where T is a tuple variable and $p(T)$ denotes a *formula* that describes T;

- The result of this query is the set of all tuples t for which the formula $p(T)$ evaluates to true with $T = t$.
- For example, the query "Find all sailors with rating above 7" is represented in TRC as follows:

$$\{S \mid S \in \text{Sailors} \wedge S.\text{rating} > 7\}$$

  - When this query is evaluated on an instance of the Sailors relation, the tuple variable S is instantiated successively with each tuple, and the test $S.rating>7$ is applied.

Syntax of TRC Queries

- Let *Rel* be a relation name, *R* and *S* be tuple variables,
  - *a* an attribute of *R*, and *b* an attribute of *S*.
- Let op denote an operator in the set $\{<, >, =, \leq, \geq, \neq\}$
- An **atomic formula** is one of the following:
  - *R Rel*
  - *R.a* op *S.b*
  - *R.a* op *constant*, or *constant* op *R.a*

- A **formula** is recursively defined to be one of the following,
- where *p* and *q* are themselves formulas, and *p*(*R*) denotes a formula in which the variable *R* appears:

any atomic formula

$\neg p$, $p \wedge q$, $p \vee q$, or $p \Rightarrow q$

$\exists R(p(R))$, where R is a tuple variable

$\forall R(p(R))$, where R is a tuple variable

- **Can use two *quantifiers* to tell how many instances the predicate applies to:**
  - **Existential quantifier ∃ ('there exists')**
  - **Universal quantifier ∀ ('for all')**
- **Tuple variables qualified by ∀ or ∃ are called *bound* variables, otherwise called *free* variables.**

- **Can recursively build up formulae from atoms:**
  - **An atom is a formula**
  - **If F_1 and F_2 are formulae, so are their conjunction, $F_1 \wedge F_2$; disjunction, $F_1 \vee F_2$; and negation, $\sim F_1$**
  - **If *F* is a formula with free variable *X*, then $(\exists X)(F)$ and $(\forall X)(F)$ are also formulae.**

Examples of TRC Queries

- *Find the names and ages of sailors with a rating above 7.*

$$\{P \mid \exists S \in Sailors(S.rating > 7 \wedge P.name = S.sname \wedge P.age = S.age)\}$$

Find the sailor name, boat id, and reservation date for each reservation.

$$\{P \mid \exists R \in Reserves \ \exists S \in Sailors$$
$$(R.sid = S.sid \wedge P.bid = R.bid \wedge P.day = R.day \wedge P.sname = S.sname)\}$$

- *Find the names of sailors who have reserved boat 103.*

$$\{P \mid \exists S \in Sailors \ \exists R \in Reserves(R.sid = S.sid \wedge R.bid = 103 \wedge P.sname = S.sname)\}$$

- *Find the names of sailors who have reserved a red boat.*

$$\{P \mid \exists S \in Sailors \ \exists R \in Reserves(R.sid = S.sid \wedge P.sname = S.sname$$
$$\wedge \exists B \in Boats(B.bid = R.bid \wedge B.color = 'red'))\}$$

- *OR*

$$\{P \mid \exists S \in Sailors \ \exists R \in Reserves \ \exists B \in Boats$$
$$(R.sid = S.sid \wedge B.bid = R.bid \wedge B.color = 'red' \wedge P.sname = S.sname)\}$$

Find the names of sailors who have reserved at least two boats.

$$\{P \mid \exists S \in Sailors \ \exists R1 \in Reserves \ \exists R2 \in Reserves$$
$$(S.sid = R1.sid \wedge R1.sid = R2.sid \wedge R1.bid \neq R2.bid \wedge P.sname = S.sname)\}$$

Find the names of sailors who have reserved all boats.

$$\{P \mid \exists S \in Sailors \ \forall B \in Boats$$
$$(\exists R \in Reserves(S.sid = R.sid \wedge R.bid = B.bid \wedge P.sname = S.sname))\}$$

- *Find sailors who have reserved all red boats.*

$$\{S \mid S \in Sailors \wedge \forall B \in Boats$$
$$(B.color = 'red' \Rightarrow (\exists R \in Reserves(S.sid = R.sid \wedge R.bid = B.bid)))\}$$

DRC (Domain Relational Calculus)

- **A domain variable** is a variable that ranges over the values in the domain of some attribute
  - e.g., the variable can be assigned an integer if it appears in an attribute whose domain is the set of integers
- A DRC query has the form $\{<x_1, x_2,...,x_n> \mid p(<x_1, x_2,...,x_n>)\}$,
  - where each x_i is either a domain variable or a constant and $p(<x_1, x_2,...,x_n>)$ denotes a DRC formula whose only free variables are the variables among the x_i, $1 \leq i \leq n$.
- The result of this query is the set of all tuples $<x_1, x_2,...,x_n>$ for which the formula evaluates to true.

DRC Formulas

- A DRC formula is defined in a manner that is very similar to the definition of a TRC formula.
- The main difference is that the variables are now domain variables.
- Let op denote an operator in the set {<, >, =, ≤, ≥, !=} and let X and Y be domain variables.
- An **atomic formula** in DRC is one of the following:
  - $<x_1, x_2, ..., x_n> \in$ Rel, where Rel is a relation with n attributes;
    - each x_i, $1 \leq i \leq n$ is either a variable or a constant.
  - X op Y
  - X op constant, or constant op X

- A formula is recursively defined to be one of the following, where p and q are themselves formulas, and p(X) denotes a formula in which the variable X appears
  - any atomic formula
  - $\neg p$, $p \wedge q$, $p \vee q$, or $p \Rightarrow q$
  - $\exists X(p(X))$, where X is a domain variable
  - $\forall X(p(X))$, where X is a domain variable

Examples of DRC Queries

- Find all sailors with a rating above 7.
  - {<I,N,T,A>|<I,N,T,A>∈ Sailors ∧ T > 7}

- Find the names of sailors who have reserved boat 103
  - {<N>|∃I,T,A(<I,N,T,A> ∈ Sailors ∧ ∃<Ir, Br, D> ∈ Reserves (Ir = I ∧ Br = 103))}

    OR
  - {<N>|∃I,T,A(<I,N,T,A> ∈ Sailors ∧ ∃D(<I, 103, D> ∈ Reserves))}

- Find the names of sailors who have reserved a red boat.
  - {<N>|∃I,T,A(<I,N,T,A> ∈ Sailors ∧ ∃<I, Br, D> ∈ Reserves ∧ ∃<Br, BN,'red' > ∈ Boats)}

- Find the names of sailors who have reserved at least two boats
  - {<N>|∃I,T,A(<I,N,T,A> ∈ Sailors ∧ ∃Br1,Br2, D1, D2(<I,Br1, D1> ∈ Reserves ∧ <I,Br2, D2> ∈ Reserves ∧ Br1 != Br2))}

- Find the names of sailors who have reserved all boats.
  - {<N>| ∃I,T,A(<I,N,T,A> ∈ Sailors ∧ ∀B,BN,C(¬(<B,BN,C> ∈ Boats) ∨ (∃<Ir, Br, D> ∈ Reserves(I = Ir ∧ Br = B))))}

OR

  - {<N>| ∃I,T,A(<I,N,T,A> ∈ Sailors ∧ ∀<B,BN,C> ∈ Boats (∃<Ir, Br, D> ∈ Reserves(I = Ir ∧ Br = B)))}

- Find sailors who have reserved all red boats.
  - {<I,N,T,A>|<I,N,T,A> ∈ Sailors ∧ ∀<B,BN,C> ∈ Boats (C ='red' ⇒ ∃<Ir, Br, D> ∈ Reserves(I = Ir ∧ Br = B))}