

Relational Algebra and Relational Calculus

Chapter 4

in the Course Textbook

Domain and Tuple Relational Calculus

ramesh@cs.ubc.ca

<http://www.cs.ubc.ca/~ramesh/cpsc304>

Relational Algebra (RA)

A Formal Query language for the Relational Data Model

Queries are composed using a collection of operators

OPERATOR:

- Accepts one or two relation instances as arguments
- Returns: A relational instance as the result

Compose operators for Complex Queries

- Leads to the notion of a **Relational Algebra Expression** - also referred to as RA Expression.

Relational Algebra Expression (RA Expression)

An RA Expression is defined as follows:

- A Relation is an RA expression.
- A **UNARY algebra operator** applied to a single RA expression is an RA expression.
- A **BINARY algebra operator** applied to two RA expressions is an RA expression.

More Formally: RA expressions can be defined as follows.

- R is an RA expression where R is any relation.
- $u(E)$ is an RA expression where u is a unary operator and E is an RA expression.
- $E\beta F$ is an RA expression where β is a binary operator and E, F are RA expressions.

BASIC OPERATORS

Selection (σ)

Projection (π)

Union (\cup)

Cross Product (\times)

Difference ($-$)

Additional operations can be defined in terms of these basic operations.

RA - Some notes

RA is **procedural**

- Gives the order in which operators are to be applied in the query.
- A **RECIPE** or **PLAN** for query evaluation.

Practical Databases use RA expressions for computing query evaluation plans.

SELECTION (σ)

The selection operator σ

- Selects rows from a relation
- Based on a *selection condition*

The Selection Condition specifies tuples to *retain* in the result

- Boolean combination of *terms* using \vee and \wedge
- A *term* is of the form *attribute op constant* or *attribute1 op attribute2*
- op is one of $\{<, >, =, \neq, \geq, \leq\}$

Reference to Attributes in a relation: By name (R.name) or by position (R.i or i)

The schema of the resulting relation is the same as the relation itself

Projection (π)

The projection operator π

- Extracts columns from a relation
- Specifies which columns to be retained
- The other fields are *Projected Out*

The result of projection may contain *duplicates* which are eliminated.

Relation - A *SET* of tuples.

Duplicate Elimination: Expensive step

- Real databases often OMIT this duplicate elimination step
- Leads to *multiset* semantics

WE ASSUME DUPLICATE ELIMINATION IS ALWAYS DONE

Example Schema

Sailors(*sid:integer*, *sname:string*, *rating:integer*, *age:real*)

Boats(*bid:integer*, *bname:string*, *color:string*)

Reserves(*sid:integer*, *bid:integer*, *day:date*)

Example

- Instance S_1 of Sailors:

sid	sname	rating	age
22	Dustin	7	45.0
31	Lubber	8	55.5
58	Rusty	10	35.0

- Instance S_2 of Sailors:

sid	sname	rating	age
28	Yuppy	9	35.0
31	Lubber	8	55.5
44	Guppy	5	35.0
58	Rusty	10	35.0

- Instance R_1 of Reserves:

sid	bid	day
22	101	10/10/96
58	103	11/12/96

Examples Continued

- $\sigma_{\text{rating}>8}(S_1)$:

sid	sname	rating	age
58	Rusty	10	35.0

- $\sigma_{(\text{age}>40)\vee(\text{rating}>8)}(S_1)$:

sid	sname	rating	age
22	Dustin	7	45.0
31	Lubber	8	55.5
58	Rusty	10	35.0

- $\sigma_{\text{bid}=103}(R_1)$:

sid	bid	day
58	103	11/12/96

- $\pi_{\text{age}}(S_2)$:

age
35.0
55.5

- $\pi_{\text{sname,age}}(S_1)$:

sname	age
Dustin	45.0
Lubber	55.5
Rusty	35.0

Composition of σ and π

$$\sigma_{\text{age} > 35}(\pi_{\text{age}}(S_2)) :$$

age
55.5

$$\pi_{\text{sname}, \text{rating}}(\sigma_{(\text{age} > 40) \vee (\text{rating} > 8)}(S_1)) :$$

sname	rating
Dustin	7
Lubber	8
Rusty	10

$$\sigma_{\text{age} > 40}(\pi_{\text{sname}, \text{age}}(S_1)) :$$

sname	age
Dustin	45.0
Lubber	55.5

SET OPERATIONS: UNION (\cup)

Let R, S be two relations.

UNION (\cup): denoted by $R \cup S$

- Returns all tuples occurring either in R or in S or both.
- R and S have to be **union compatible**
- Union Compatible relations: Have same number of fields
- And Corresponding fields in Left to Right order have the same *domains*

Schema of $R \cup S$ is *identical* to R (or S)

From the Boats, Sailors, Reserves example

- S_1 and S_2 are *union compatible*
- S_1 and R_1 (likewise S_2 and R_1) are not

Example of Union

$\sigma_{\text{rating}>8}(S_1) \cup \sigma_{\text{rating}<8}(S_2) :$

sid	sname	rating	age
58	Rusty	10	35.0
44	Guppy	5	35.0

For this UNION: $\sigma_{\text{rating}>8}(S_1)$ and $\sigma_{\text{rating}<8}(S_2)$ must be *union compatible* and THEY ARE.

SET OPERATIONS: Intersection (\cap), Set Difference ($-$)

Intersection (\cap): $R \cap S$

- Returns all tuples occurring in both R and S
- R, S must be *union compatible*
- Schema of $R \cap S$ is identical to R (or S)

Set Difference ($-$): $R - S$

- Returns a relation instance of all tuples occurring in R *but not in* S
- R, S must be *union compatible*
- Schema of $R \cap S$ is identical to R (or S)

Examples

INTERSECTION:

- $\sigma_{\text{rating}>8}(S_1) \cap \sigma_{\text{rating}>5}(S_2)$

sid	sname	rating	age
58	Rusty	10	35.0

Is this the same as $\sigma_{\text{rating}>8}(S_1 \cap S_2)$?

Think about what can we say in general?

SET DIFFERENCE:

- $\pi_{\text{age}}(\sigma_{\text{rating}>5}(S_1)) - \pi_{\text{age}}(\sigma_{\text{rating}>5}(S_2)) :$

age
45.0

- $\sigma_{\text{rating}>5}(S_2 - S_1) :$

sid	sname	rating	age
28	Yuppy	9	35.0

- $\pi_{\text{sname}}(S_1 - S_2) :$

sname
Dustin

SET OPERATIONS: Cross Product (\times)

Returns a relational instance whose schema contains

- All fields of R (in the same order as they appear in R) followed by all the fields of S (in the same order as they appear in S)
- $R \times S$ contains ONE tuple $\langle r, s \rangle$ (Concatenation of tuples r and s) for each pair of tuples $r \in R$ and $s \in S$ (Cartesian Product)
- Fields of $R \times S$ *inherit* names from R and S

What if fields in R and S contain the same name? - Leads to a **NAMING CONFLICT**

Resolved by using position instead of name

Example

$\sigma_{\text{rating} < 8}(S_1) \times R_1 :$

(sid)	sname	rating	age	(sid)	bid	day
22	Dustin	7	45.0	22	101	10/10/96
22	Dustin	7	45.0	58	103	11/12/96

Columns 1 and 5 have the same name `sid` and hence a *naming conflict*

To resolve naming conflicts, a *naming operator* ρ is introduced.

Renaming Operator (ρ)

$\rho(R(\bar{F}), E)$:

- Takes an arbitrary RA expression E
- Returns an instance of a new relation called R

Returned relation R

- Same tuples as the result of E and the same schema as E
- *but* SOME fields are renamed

\bar{F} - are the fields that are *renamed* and is specified as `oldname \rightarrow newname` or `position \rightarrow newname`

R and \bar{F} are both optional.

Example of Renaming

$\rho(C(1 \rightarrow \text{sid1}, 5 \rightarrow \text{sid2}), S_1 \times R_1) :$

- $S_1 : \text{Sailors}(\text{sid}, \text{sname}, \text{rating}, \text{age})$
- $R_1 : \text{Reserves}(\text{sid}, \text{bid}, \text{day})$

$S_1 \times R_1 :$

- $C(\text{sid1}, \text{sname}, \text{rating}, \text{age}, \text{sid2}, \text{bid}, \text{day})$
- The fields `sname`, `rating`, `age`, `bid` and `day` are left unchanged.

Domains of each field are the same as before.

JOINS

One of the MOST USEFUL operations

Most common way to combine information from two or more relations.

Several Variants of Join - (OUTER JOINS when we do SQL)

Join operator is denoted by \bowtie

Conditional JOIN

A **join condition** - C , R and S are two relations

Conditional Join is given by $R \bowtie_C S = \sigma_C(R \times S)$

Conditional join is expressed as a selection based on the condition C from the cross product $R \times S$ of the two relations R and S

C can refer to attributes of both R and S

Reference to an attribute in R (or S) can be by name ($R.name$) or by position ($R.i$)

Example: $S_1 \bowtie_{S_1.sid < R_1.sid} R_1$

(sid)	sname	rating	age	(sid)	bid	day
22	Dustin	7	45.0	58	103	11/12/96
31	Lubber	8	55.5	58	103	11/12/96

EQUIJOIN

An EQUIJOIN is a JOIN with the following conditions:

- When JOIN Condition consists **solely** of EQUALITIES (possibly connected by \wedge)
- Additionally, one of the equated fields (from each equality) is *projected out* from the result.

Example of EQUIJOIN: $S_1 \bowtie_{S_1.sid=R_1.sid} R_1$

sid	sname	rating	age	bid	day
22	Dustin	7	45.0	101	10/10/96
58	Rusty	10	35.0	103	11/12/96

Note that one of the sid fields is projected out.

NATURAL JOIN

A *NATURAL JOIN* is an equijoin where *all* equalities are specified on ALL fields of R and S having the SAME NAME.

- We OMIT the join condition C
- No two fields in the result will have the same name

Example of NATURAL JOIN: $S_1 \bowtie R_1$

sid	sname	rating	age	bid	day
22	Dustin	7	45.0	101	10/10/96
58	Rusty	10	35.0	103	11/12/96

Answer is the Same as EQUIJOIN example before.

$S_1 \bowtie S_2 :$

sid	sname	rating	age
31	Lubber	8	55.5
58	Rusty	10	35.0

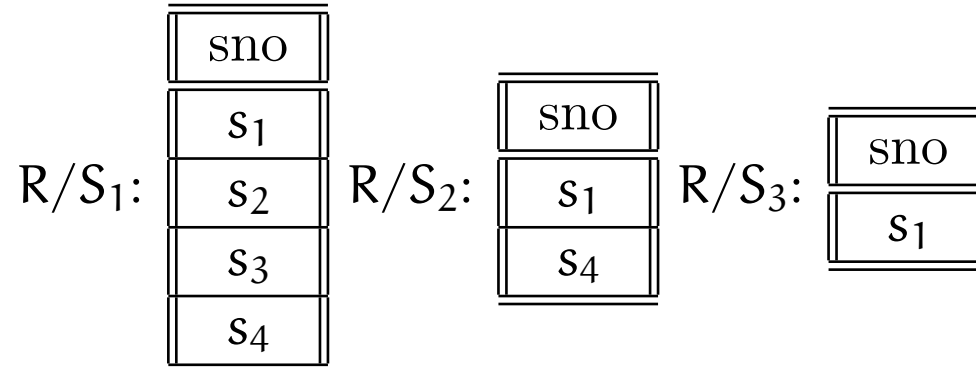
DIVISION: R/S

- Illustrate it with an EXAMPLE:

	sno	pno
R:	s ₁	p ₁
	s ₁	p ₂
	s ₁	p ₃
	s ₁	p ₄
	s ₂	p ₁
	s ₂	p ₂
	s ₃	p ₂
	s ₄	p ₂
	s ₄	p ₄

S ₁ :	pno	S ₂ :	pno	S ₃ :	pno
	p ₂		p ₂		p ₁
	p ₂		p ₄		p ₂
	p ₂		p ₄		p ₄

DIVISION EXAMPLE CONTINUED



Each of s_1, s_2, s_3, s_4 occur in R with all the values in the instance S_1
ONLY s_1 and s_4 occur in R with all values $\{p_2, p_4\}$ of instance S_2

Expression Division in terms of RA operators

R/S :

- A value X in R is **disqualified** if by attaching a value y in S , $\langle x, y \rangle$ is NOT in R .
- Compute x values in R that are *NOT DISQUALIFIED*

$\pi_x((\pi_x(R) \times S) - R)$ gives the set of *disqualified values*

Therefore, $R \times S$ is given by

$$\pi_x(R) - \pi_x((\pi_x(R) \times S) - R)$$

This is for SINGLE attributes. Think about generalizing to multiple attributes.

EXAMPLES

Find the names of sailors who have reserved a red boat

- $\pi_{sname}((\sigma_{color='red'}Boats) \bowtie Reserves \bowtie Sailors)$
- $\pi_{sname}(\pi_{sid}((\pi_{bid}\sigma_{color='red'}Boats) \bowtie Reserves) \bowtie Sailors)$

Find the colors of boats reserved by Lubber

- $\pi_{color}((\sigma_{sname='Lubber'}Sailors) \bowtie Reserves \bowtie Boats)$

Find the name of sailors who have reserved at least one boat

- $\pi_{sname}(Sailors \bowtie Reserves)$

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

- $\rho(Reservations, \pi_{sid,sname,bid}(Sailors \bowtie Reserves))$

$$\rho(ResPairs(1 \rightarrow sid_1, 2 \rightarrow sname_1, 3 \rightarrow bid_1, 4 \rightarrow sid_2, 5 \rightarrow sname_2, 6 \rightarrow bid_2), Reservations \times Reservations)$$

$$\pi_{sname_1} \sigma_{(sid_1=sid_2) \wedge (bid_1 \neq bid_2)} ResPairs$$

MORE EXAMPLES

Find the names of sailors who have reserved a red boat or a green boat

- $\rho(\text{TBoats}, (\sigma_{\text{color}='red'}\text{Boats}) \cup (\sigma_{\text{color}='green'}\text{Boats}))$
 $\pi_{\text{sname}}(\text{TBoats} \bowtie \text{Reserves} \bowtie \text{Sailors})$
- **Equivalently:** $\rho(\text{TBoats}, (\sigma_{\text{color}='red'} \vee \sigma_{\text{color}='green'}}\text{Boats}))$
 $\pi_{\text{sname}}(\text{TBoats} \bowtie \text{Reserves} \bowtie \text{Sailors})$

Find the names of sailors who have reserved a red boat AND a green boat

- **INCORRECT:** $\rho(\text{TBoats}, (\sigma_{\text{color}='red'}\text{Boats}) \cap (\sigma_{\text{color}='green'}\text{Boats}))$
 $\pi_{\text{sname}}(\text{TBoats} \bowtie \text{Reserves} \bowtie \text{Sailors})$
- **The CORRECT Solution is:**
- $\rho(\text{TempRed}, \pi_{\text{sid}}((\sigma_{\text{color}='red'}\text{Boats} \bowtie \text{Reserves})))$
 $\rho(\text{TempGreen}, \pi_{\text{sid}}((\sigma_{\text{color}='green'}\text{Boats} \bowtie \text{Reserves})))$
 $\pi_{\text{sname}}((\text{TempRed} \cap \text{TempGreen}) \bowtie \text{Sailors})$

MORE EXAMPLES

Find the sids of sailors with age over 20 who have not reserved a red boat

- $\pi_{\text{sid}}(\sigma_{\text{age}>20}\text{Sailors}) - \pi_{\text{sid}}((\sigma_{\text{color}='red'}\text{Boats}) \bowtie \text{Reserves} \bowtie \text{Sailors})$

Find the names of sailors who have reserved all the boats

- $\rho(\text{TempSids}, (\pi_{\text{sid,bid}}\text{Reserves})/(\pi_{\text{bid}}\text{Boats}))$
 $\pi_{\text{sname}}(\text{TempSids} \bowtie \text{Sailors})$

Find the names of all sailors who have reserved all boats called Unicorn

- $\rho(\text{TempSids}, (\pi_{\text{sid,bid}}\text{Reserves})/(\pi_{\text{bid}}(\sigma_{\text{name}='Unicorn'}\text{Boats})))$
 $\pi_{\text{sname}}(\text{TempSids} \bowtie \text{Sailors})$

RELATIONAL CALCULUS (RC)

Non Procedural or Declarative

Allows description of answers based on "*What Users WANT?*"

RA - is Procedural and gives "*HOW the result should be computed?*"

There are TWO VARIANTS of Relational Calculus

- TUPLE RELATIONAL CALCULUS (TRC) - Variables take *tuples* as values
- DOMAIN RELATIONAL CALCULUS (DRC) - Variables range over field/attribute values

Tuple Relational Calculus (TRC)

TUPLE VARIABLE: Takes on values that are tuples of a particular relation schema

TRC QUERY: $\{T|p(T)\}$ where

- T - Tuple variable
- $p(T)$ - A *formula* that describes T

Result of the query:

- Set of all tuples t for which the formula $p(T)$ evaluates to *true* with $T = t$.
- Simple subset of *First Order Logic*

Syntax of TRC Queries

Rel : Relation Name R, S : - Tuple Variables

a : Attribute of R b : - Attribute of S

op : Operator in the set $\{<, >, =, \neq, \geq, \leq\}$

ATOMIC FORMULA: An atomic formula is defined as follows.

- $R \in \text{Rel}$
- $R.a \text{ op } S.b$
- $R.a \text{ op Constant}$ or $\text{Constant op } R.a$

FORMULA: Let p, q be any formulas.

- Any atomic formula is a formula.
- $\neg p, p \wedge q, p \vee q, p \Rightarrow q$ are formulas
- $\exists R(p(R))$ and $\forall R(p(R))$ are formulas where R is a tuple variable.
- $p(R)$ denotes a formula in which the variable R appears.

Syntax of TRC Queries

The quantifiers \exists, \forall are said to *BIND* the variable R.

A variable is said to be *FREE* in a formula/subformula, if the (sub)formula does not contain an occurrence of a quantifier that binds it.

A TRC Query:

- An expression of the form $\{T|p(T)\}$, where T is the *ONLY FREE VARIABLE* in the formula p.

Semantics of TRC Queries

What does a TRC Query MEAN?

- The answer to $\{T|p(T)\}$ is the set of all tuples t for which $p(T)$ evaluates to TRUE with $T = t$.

Which assignments of tuple values to free variables in a formula make it evaluate it to TRUE?

Let us set the concepts up for answering the above question.

- Query - evaluated on an INSTANCE of the database
- F - a formula
- Let each free variable in F be bound to a tuple value

Semantics of TRC Queries

Formula F evaluates to TRUE if *ONE* of the following holds. (Consider all the cases of formulas)

- F is $R \in \text{Rel}$: and R is assigned a tuple in the instance of the relation R
- F is $R.a \text{ op } S.b$, $R.a \text{ op Constant}$ or $\text{Constant op } R.a$ and the tuples assigned to R and S have the field values $R.a$ and $S.b$ that make the comparison TRUE
- F is:
 - ▷ $\neg p$ and p is false
 - ▷ $p \wedge q$ and both p and q are true
 - ▷ $p \vee q$ and one (or both) of them is true
 - ▷ $p \Rightarrow q$ and q is true whenever p is true
- F is $\exists R(p(R))$ and there is *SOME* assignment of tuples to free variables in $p(R)$ (including R), that makes $p(R)$ true
- F is $\forall R(p(R))$ and *NO* matter what tuple is assigned to R , there is some assignment of tuples to the free variables in $p(R)$ that makes the formula $p(R)$ true.

Examples

Schema:

- Sailors(sid:integer, sname:string, rating:integer, age:real)
- Boats(bid:integer, bname:string, color:string)
- Reserves(sid:integer, bid:integer, day:date)
- $\{P | (P \in \text{Sailors} \wedge P.\text{rating} > 8)\}$ applied on S_1
corresponds to $\sigma_{\text{rating}>8}(S_1)$
- $\{P | (P \in \text{Sailors} \wedge ((P.\text{age} > 40) \vee (P.\text{rating} > 8)))\}$ applied to S_1
corresponds to $\sigma_{\text{age}>40}(S_1)$
- $\{P | (P \in \text{Reserves} \wedge P.\text{bid} = 103)\}$ applied to R_1
corresponds to $\sigma_{\text{bid}=103}(R_1)$
- $\{P | \exists S \in \text{Sailors}(P.\text{age} = S.\text{age})\}$ applied to S_2
corresponds to $\pi_{\text{age}}(S_2)$
- $\{P | \exists S \in \text{Sailors}(P.\text{sname} = S.\text{sname} \wedge P.\text{age} = S.\text{age})\}$ applied to S_1
corresponds to $\pi_{\text{sname,age}}(S_1)$

MORE EXAMPLES

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

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

Find the names of sailors who have reserved at least 2 boats

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

Find the names of sailors who have reserved all boats

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

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

- $\{S | S \in \text{Sailors} \wedge \forall B \in \text{Boats} (B.\text{color} = 'red' \Rightarrow (\exists R \in \text{Reserves} (S.\text{sid} = R.\text{sid} \wedge R.\text{bid} = B.\text{bid})))\}$
- **Without using Implication:** The query can be written as $\{S | S \in \text{Sailors} \wedge \forall B \in \text{Boats} (B.\text{color} \neq 'red' \vee (\exists R \in \text{Reserves} (S.\text{sid} = R.\text{sid} \wedge R.\text{bid} = B.\text{bid})))\}$

DOMAIN RELATIONAL CALCULUS (DRC)

Domain Variable: A variable that ranges over the values in the *domain* of some attribute.

DRC Query:

- $\{\langle x_1, x_2, \dots, x_n \rangle \mid p(\langle x_1, \dots, x_n \rangle)\}$
- x_i - Domain variable or constant
- $p(\langle x_1, x_2, \dots, x_n \rangle)$ - a DRC formula

The **ONLY** free variables in the formula are the variables among the x_i , $1 \leq i \leq n$.

The *result* of the query is the set of all tuples $\langle x_1, x_2, \dots, x_n \rangle$ for which the formula evaluates to *true*.

Atomic Formulas, Formulas - DRC

Atomic Variables are defined as follows: x, y - Domain Variables, op - $\{<, >, =, \neq, \leq, \geq\}$

- $\langle x_1, x_2, \dots, x_n \rangle \in \text{Rel}$ where Rel is a relation with n attributes, Each $x_i, 1 \leq i \leq n$ is either a variable or a constant.
- $x \text{ op } y$
- $x \text{ op constant}$ or $\text{constant op } x$

Formulas are defined as follows: (Assume that p and q are formulas)

$p(X)$ - Formula in which variable X appears

- Any Atomic formula is a formula.
- $\neg p, p \vee q, p \wedge q, p \Rightarrow q$
- $\exists X(p(X))$ where X is a domain variable
- $\forall X(p(X))$ where X is a domain variable

Compare with TRC and define *semantics* for DRC

EXAMPLES

Find all sailors with a rating above 7

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

Find the names of sailors who have reserved boat 103

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

In the above example, Ir , Br and D appear in a single relation. A more compact notation would be $\exists \langle Ir, Br, D \rangle \in \text{Reserves}$ instead of $\exists Ir, Br, D(\dots)$.

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

MORE EXAMPLES

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

- $\{\langle N \rangle \mid \exists I, T, A (\langle I, N, T, A \rangle \in \text{Sailors} \wedge \exists Br1, Br2, D1, D2 (\langle I, Br1, D1 \rangle \in \text{Reserves} \wedge \langle I, Br2, D2 \rangle \in \text{Reserves} \wedge Br1 \neq Br2))\}$

Find the names of sailors who have reserved all the boats

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

Find sailors who have reserved all red boats

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

Expressive Power: RA and RC

Can every RA query be expressed in RC? **YES. THEY CAN.**

The other way: Can every RC query be expressed in RA?

Let us examine RC further before answering the question.

Consider the TRC query $\{S | \neg(S \in \text{Sailors})\}$

The query is *syntactically* OK!!!

- All tuples such that S is *not in* the given instance of Sailors.
- PROBLEM: What about infinite domains like integers? The answer is infinite.
- Such queries are UNSAFE.

Restrict Calculus queries to be **SAFE**

SAFE Queries

Let I - Set of relation instances, one instance per relation that appears in the query Q

$DOM(Q, I)$ - Set of all **CONSTANTS** that appear in these relation instances I or in the formulation of Q

I is finite and hence $DOM(Q, I)$ is finite.

A **SAFE TRC Formula Q** is:

- For any given I , the set of answers for Q contains only values in $DOM(Q, I)$
- For each subexpression $\exists R(p(R))$ in Q , if tuple r makes the formula true, then r contains only constants in $DOM(Q, I)$
- For each subexpression $\forall R(p(R))$ in Q , if a tuple r contains a constant that is NOT IN $DOM(Q, I)$ (assigned to R), then r MUST make the formula true

The BOTTOM LINE

Every query that can be expressed using a *SAFE* RC query can also be expressed as an RA query

Expressive power of RA - Often used as a metric for query languages of relational databases

Relational Completeness: If a query language can express *all* the queries that can be expressed in RA, it is said to be relationally complete.

EXAMPLES

Schema:

- Sailors(sid:integer, sname:string, rating:integer, age:real)
- Boats(bid:integer, bname:string, color:string)
- Reserves(sid:integer, bid:integer, day:date)

Query: Find the names of Sailors who have reserved boat 103

- Relational Algebra: $\pi_{\text{sname}}((\sigma_{\text{bid}=103}\text{Reserves}) \bowtie \text{Sailors})$
- TRC: $\{P | \exists S \in \text{Sailors}, \exists R \in \text{Reserves} (R.\text{sid} = S.\text{sid} \wedge R.\text{bid} = 103 \wedge P.\text{sname} = S.\text{sname})\}$
- DRC: $\{\langle N \rangle | \exists I, T, A (\langle I, N, T, A \rangle \in \text{Sailors} \wedge \exists Ir, Br, D (\langle Ir, Dr, D \rangle \in \text{Reserves} \wedge Ir = I \wedge Br = 103))\}$
- SQL:

```
SELECT S.sname
FROM Sailors S
WHERE S.sid IN ( SELECT R.sid
                  FROM Reserves R
                  WHERE R.bid = 103 )
```

EXAMPLES

Query: Find the names of Sailors who have reserved a Red boat

- RA: $\pi_{\text{sname}}((\sigma_{\text{color}='Red'}\text{Boats}) \bowtie \text{Reserves} \bowtie \text{Sailors})$ or
- RA: $\pi_{\text{sname}}(\pi_{\text{sid}}((\pi_{\text{bid}}\sigma_{\text{color}='Red'}\text{Boats}) \bowtie \text{Reserves}) \bowtie \text{Sailors})$
- TRC: $\{P|\exists S \in \text{Sailors}, \exists R \in \text{Reserves}(R.\text{sid} = S.\text{sid} \wedge P.\text{sname} = S.\text{sname} \wedge \exists B \in \text{Boats}(B.\text{bid} = R.\text{bid} \wedge B.\text{color} = 'Red'))\}$
- DRC: $\{\langle N \rangle | \exists I, T, A(\langle I, N, T, A \rangle \in \text{Sailors} \wedge \exists \langle I, Br, D \rangle \in \text{Reserves} \wedge \exists \langle Br, BN, 'Red' \rangle \in \text{Boats})\}$
- SQL:

```

SELECT S.sname
FROM   Sailors S
WHERE  S.sid IN ( SELECT  R.sid
                  FROM    Reserves R
                  WHERE   IN  ( SELECT  B.bid
                              FROM    Boats B
                              WHERE   B.color = 'Red' ))

```

EXAMPLES

Query: Find the names of Sailors who have reserved atleast two boats

- RA: $\rho(\text{Reservations}, \pi_{\text{sid}, \text{sname}, \text{bid}}(\text{Reserves} \bowtie \text{Sailors}))$
- $\rho(\text{ResPairs}(1 \rightarrow \text{sid}_1, 2 \rightarrow \text{sname}_1, 3 \rightarrow \text{bid}_1, 4 \rightarrow \text{sid}_2, 5 \rightarrow \text{sname}_2, 6 \rightarrow \text{bid}_2), \text{Reservations} \times \text{Reservations})$
- $\pi_{\text{sname}_1}(\sigma_{\text{sid}_1 = \text{sid}_2 \wedge \text{bid}_1 \neq \text{bid}_2} \text{ResPairs})$
- TRC: $\{P | \exists S \in \text{Sailors}, \exists R_1 \in \text{Reserves} \exists R_2 \in \text{Reserves} S.\text{sid} = R_1.\text{sid} \wedge R_1.\text{sid} = R_2.\text{sid} \wedge R_1.\text{bid} \neq R_2.\text{bid} \wedge P.\text{sname} = S.\text{sname}\}$
- DRC: $\{\langle N \rangle | \exists I, T, A (\langle I, N, T, A \rangle \in \text{Sailors} \wedge \exists Br_1, Br_2, D_1, D_2 (\langle I, Br_1, D_1 \rangle \in \text{Reserves} \wedge \langle I, Br_2, D_2 \rangle \in \text{Reserves} \wedge Br_1 \neq Br_2))\}$
- SQL:

```
SELECT S.sname
FROM Sailors S
WHERE S.sid IN ( SELECT R.sid
                  FROM Reserves R, Reserves T
                  WHERE R.sid = T.sid AND
                        R.bid <> T.bid )
```