Updated 1/24/2023

Topic 1: Data models and query languages
Unit 2: Logic & relational calculus
Lecture 5

Wolfgang Gatterbauer

CS7240 Principles of scalable data management (sp23)

https://northeastern-datalab.github.io/cs7240/sp23/

1/24/2023

Pre-class conversations

- Last class recapitulation
- Scribe suggestion: post to Piazza after but without my feedback

- today:
  - logic continued (likely next time algebra and the connection)
  - logic is super important for our class; thus lots of practice today ☺
  - in particular the concept of "undecidability": intuition for why things can quickly get complicated without giving proofs

More practice

- "A small, happy dog is at home"

?

- "Every small dog that is at home is happy."

?

- "Jiahui owns a small, happy dog"

?

- "Jiahui owns every small, happy dog."

?

Example adopted from Barker-Plummer, Barwise, Etchemendy - Language, Proof, And Logic (book, 2nd ed), 2011. https://www.gradegrinder.net/

More practice

- "A small, happy dog is at home"
  - $\exists x\ [(Small(x) \land Happy\ (x) \land Dog\ (x)) \land Home(x)]$

associativity of conjunction: no need of evaluation to follow blue parentheses

- "Every small dog that is at home is happy."

?

- "Jiahui owns a small, happy dog"

?

- "Jiahui owns every small, happy dog."

?

More practice

- "A small, happy dog is at home"
  - $\exists x\ [(Small(x) \land Happy(x) \land Dog(x)) \land Home(x)]$

associativity of conjunction: no need of evaluation to follow blue parentheses

- "Every small dog that is at home is happy."
  - $\forall x\ [(Small(x) \land Dog(x) \land Home(x)) \rightarrow Happy(x)]$

here evaluation needs to follow blue parentheses

- "Jiahui owns a small, happy dog"

?

- "Jiahui owns every small, happy dog."

?

More practice

- "A small, happy dog is at home"
  - $\exists x\, [(\text{Small}(x) \land \text{Happy}(x) \land \text{Dog}(x)) \land \text{Home}(x)]$

associativity of conjunction: no need of evaluation to follow blue parentheses

- "Every small dog that is at home is happy."
  - $\forall x\, [(\text{Small}(x) \land \text{Dog}(x) \land \text{Home}(x)) \rightarrow \text{Happy}(x)]$

here evaluation needs to follow blue parentheses

- "Jiahui owns a small, happy dog"
  - $\exists x\, [(\text{Small}(x) \land \text{Happy}(x) \land \text{Dog}(x)) \land \text{Owns}(\text{'Jiahui'}, x)]$

notice that we deviate here from the usual notation in logics of constants like 'Jiahui' written w/o quotation marks

- "Jiahui owns every small, happy dog."

?

More practice

- "A small, happy dog is at home"
  - ∃x [(Small(x) ∧ Happy (x) ∧ Dog (x)) ∧ Home(x)]

associativity of conjunction: no need of evaluation to follow blue parentheses

- "Every small dog that is at home is happy."
  - ∀x [(Small(x) ∧ Dog (x) ∧ Home(x)) → Happy (x)]

here evaluation needs to follow blue parentheses

- "Jiahui owns a small, happy dog"
  - ∃x [(Small(x) ∧ Happy (x) ∧ Dog (x)) ∧ Owns('Jiahui', x)]
- "Jiahui owns every small, happy dog."
  - ∀x [(Small(x) ∧ Happy (x) ∧ Dog (x)) → Owns('Jiahui', x)]

notice that we deviate here from the usual notation in logics of constants like 'Jiahui' written w/o quotation marks

Two more examples

- "There are infinitely many prime numbers"

?

Two more examples

- "There are infinitely many prime numbers"
  - $\forall x \, \exists y \, [y > x \land \text{Prime}(y)]$

Source first example: Vasco Brattka. Logic and computation (lecture notes), 2007. http://cca-net.de/vasco/lc/

Two more examples

- "There are infinitely many prime numbers"
  - $\forall x\ \exists y\ [y > x \land \mathrm{Prime}(y)]$

- $\forall x\ \exists y\ [y = \mathrm{sqrt}(x)]$

?

Source first example: Vasco Brattka. Logic and computation (lecture notes), 2007. http://cca-net.de/vasco/lc/

Two more examples

- "There are infinitely many prime numbers"
  - $\forall x\, \exists y\, [y > x \land \text{Prime}(y)]$

- $\forall x\, \exists y\, [y = \text{sqrt}(x)]$
  - Truth of this expression depends on domain:
    - evaluates to false if x and y have the domain of the real numbers $\mathbb{R}$
    - evaluates to true if their domain is the complex numbers $\mathbb{C}$

Source first example: Vasco Brattka. Logic and computation (lecture notes), 2007. http://cca-net.de/vasco/lc/

Semantics of First-Order Logic on Graphs

$E(x,y)$
$A(x,y)$
Parent('Alice','Bob')

Semantics:

- First-order variables range over (can be " bound to") elements of the universe of structures
- To evaluate a formula φ, we need a graph G and a binding α that maps the free variables of φ to nodes of G

Notation: $G \vDash_{\alpha} \varphi(x_1, \dots, x_k)$

Relational Databases

Codd's Two Fundamental Ideas:

- Tables are relations: a row in a table is just a tuple in a relation; order of rows/tuples does not matter!
- Formulas are queries: they specify the **What** rather then the **How**! That's declarative programming

order. For example, when setting goals, just set goals. Don't think about how you will achieve them or what you will do if something goes wrong. When you are diagnosing problems, don't think about how you will solve them—just diagnose them. Blurring the steps leads to suboptimal outcomes because it interferes with uncovering the true problems. The process is iterative: Doing each step thoroughly will provide you with the information you need to move on to the next step and do it well.

a. **Focus on the "what is" before deciding "what to do about it."** It is a common mistake to move in a nanosecond from identifying a tough problem to proposing a solution for it. Strategic thinking requires both diagnosis and design. A good diagnosis typically takes between fifteen minutes and an hour, depending on how well it's done and how complex the issue is. It involves speaking with the relevant people and looking at the evidence together to determine the root causes. Like principles, root causes manifest themselves over and over again in seemingly different situations. Finding them and dealing with them pays dividends again and again.

f. **Recognize that it doesn't take a lot of time to design a good plan.** A plan can be sketched out and refined in just hours or spread out over days or weeks. But the process is essential because it determines what you will have to do to be effective. Too many people make the mistake of spending virtually no time on designing because they are preoccupied with execution. Remember: Designing precedes doing!

b. **Good work habits are vastly underrated.** People who push through successfully have to-do lists that are reasonably prioritized, and they make certain each item is ticked off in order.

Separation of concerns: WHAT from HOW

Topic 4: Normalization, Information theory & Axioms of Uncertainty

PRELIMINARY

- Unit 1: Normal Forms & Information Theory
  - [Cow'03] Ramakrishnan, Gehrke. Database Management Systems. 3rd ed 2003. Ch 19: Normalization.
  - [Complete'08] Garcia-Molina, Ullman, Widom. Database Systems. 2nd ed. 2008. Ch 3: Design theory.
  - [Elmasri, Navathe'15] Fundamentals of Database Systems. 7th ed 2015. Ch 14 & 15: Normal forms,
  - [Silberschatz+'20] Silberschatz, Korth, Sudarshan. Database system concepts. 7th ed 2020. Ch 7.5 & 7.6: Relational design with decomposition.
  - [Arenas, Libkin'05] An information-theoretic approach to normal forms for relational and XML data. JACM 2005.
  - [Lee'87] An Information-Theoretic Analysis of Relational Databases-Part I: Data Dependencies and Information Metric. IEEE Transactions on Software Engineering 1987.
  - [Olah'15] Visual Information Theory. Beautiful blog post on the intuition behind entropy.
- Unit 2: Axioms for Uncertainty
  - [Cox'46] Probability, frequency and reasonable expectation, American Journal of Physics, 1946.
  - [Shannon'48] A Mathematical Theory of Communication, The Bell System Technical Journal, 1948.
  - [Van Horn'03] Constructing a logic of plausible inference: a guide to Cox's theorem, International Journal of Approximate Reasoning, 2003.

3 Components of FOL

1. Syntax (or language) ?

2. Interpretation ?

3. Semantics ?

3 Components of FOL

1. Syntax (or language)
   – *What are the allowed syntactic expressions?*

2. Interpretation
   – *Mapping symbols to an actual world*

3. Semantics
   – *When is a statement “true” under some interpretation?*

3 Components of FOL

1. Syntax (or language)
   - *What are the allowed syntactic expressions?*
   - For DB's: ?
2. Interpretation
   - *Mapping symbols to an actual world*
   - For DB's: ?
3. Semantics
   - *When is a statement "true" under some interpretation?*
   - For DB's: ?

3 Components of FOL

1. Syntax (or language)
   - *What are the allowed syntactic expressions?*
   - For DB's: schema, constraints, query language
2. Interpretation
   - *Mapping symbols to an actual world*
   - For DB's : database
3. Semantics
   - *When is a statement "true" under some interpretation?*
   - For DB's : meaning of integrity constraints and query results

Components of FOL: (1) Syntax = First-order language

vocabulary

- Alphabet: symbols in use
  - Variables, constants, **function** symbols, **predicate** symbols, connectives, quantifiers, punctuation symbols

x Alice   MotherOf(x)   Friends(x,y) $\wedge$ $\exists$ ( )

terms   relation b/w objects

- Term: expression that stands for an element or object
  - Variable, constant
  - Inductively $f(t_1,\ldots,t_n)$ where t_i are terms, f a function symbol   MotherOf(MotherOf(x))
- (Well-formed) formula: parameterized statement
  - Atom $p(t_1,\ldots,t_n)$ where p is a predicate symbol, t_i terms (atomic formula, together with predicates $t_1=t_2$)   x = 'Alice'
  - Inductively, for formulas F, G, variable x:

$$F \wedge G \quad F \vee G \quad \neg F \quad F \longrightarrow G \quad F \longleftrightarrow G \quad \forall x\, F \quad \exists x\, F$$

- A first-order language refers to the set of all formulas over an alphabet

Components of FOL: (2) Interpretation

- How to assign meaning to the symbols of a formal language
- An interpretation INT for an alphabet consists of:
  - A non-empty set **Dom**, called domain
    - {Alice, Bob, Charly}
  - An assignment of an element in **Dom** to each constant symbol
    - Alice (recall we often write constants with quotation marks 'Alice')
  - An assignment of a function $\mathbf{Dom}^n \longrightarrow \mathbf{Dom}$ to each n-ary function symbol
    - Alice = MotherOf(Bob)
  - An assignment of a function $\mathbf{Dom}^n \longrightarrow \{true, false\}$ (i.e., a relation) to each n-ary predicate symbol
    - Friends(Bob, Charly) = TRUE

Components of FOL: (3) Semantics

- A variable assignment V to a formula in an interpretation INT assigns to each free variable X a value from **Dom**
  - Recall, a free variable is one that is not quantified

Person(X) $\exists Y$ Married(X, $\underbrace{Y}_{\text{bound}}$)

- Truth value for formula F under interpretation INT and variable assignment V:
  - Atom $\mathrm{p}(\mathrm{t}_1,\ldots,\mathrm{t}_n)$: $\mathrm{q}(\mathrm{s}_1,\ldots,\mathrm{s}_n)$ where q is the interpretation of the predicate p and s_i the interpretation of t_i
  - $\mathrm{F} \wedge \mathrm{G}$ $\mathrm{F} \vee \mathrm{G}$ $\neg \mathrm{F}$ $\mathrm{F} \longrightarrow \mathrm{G}$ $\mathrm{F} \longleftrightarrow \mathrm{G}$: according to truth table
  - $\exists X F$: true iff there exists d∈**Dom** such that if V assigns d to X then the truth value of F is true; otherwise false
  - $\forall X F$: true iff for all d∈**Dom**, if V assigns d to X then the truth value of F is true; otherwise false
- If a formula has no free vars (closed formula or sentence), we can simply refer to its truth value under INT

$\forall X$: $\underbrace{\text{Person}(X)}_{F} \longrightarrow \underbrace{\text{Mortal}(X)}_{G}$

Formula — Sentence
Formula — Query

Operator precedence

Operator precedence is an ordering of logical operators designed to allow the dropping of parentheses in logical expressions. The following table gives a hierarchy of precedences for the operators of propositional logic. The $\neg$ operator has higher precedence than $\wedge$; $\wedge$ has higher precedence than $\vee$; and $\vee$ has higher precedence than $\Rightarrow$ and $\Leftrightarrow$.

$$\neg$$
$$\wedge$$
$$\vee$$
$$\Rightarrow \; \Leftrightarrow$$

In unparenthesized sentences, it is often the case that an expression is flanked by operators, one on either side. In interpreting such sentences, the question is whether the expression associates with the operator on its left or the one on its right. We can use precedence to make this determination. In particular, we agree that an operand in such a situation always associates with the operator of higher precedence. When an operand is surrounded by operators of equal precedence, the operand associates to the right. The following examples show how these rules work in various cases. The expressions on the right are the fully parenthesized versions of the expressions on the left.

$\neg p \wedge q$	$((\neg p) \wedge q)$
$p \wedge \neg q$	$(p \wedge (\neg q))$
$p \wedge q \vee r$	$((p \wedge q) \vee r)$
$p \vee q \wedge r$	$(p \vee (q \wedge r))$
$p \Rightarrow q \Rightarrow r$	$(p \Rightarrow (q \Rightarrow r))$
$p \Rightarrow q \Leftrightarrow r$	$(p \Rightarrow (q \Leftrightarrow r))$

Queries and the connection to logic

- Why logic?
- A crash course in FOL
- Relational Calculus (RC)
  - Syntax and Semantics
  - Domain RC (DRC) vs Tuple RC (TRC)
  - Domain Independence and Safety

Entire Story in One Slide

1. RC = FOL over DB
2. RC can express “bad queries” that depend not only on the DB, but also on the domain from which values are taken (domain dependence)
3. We cannot test whether an RC query is “good,” but we can use a ”good” subset of RC that captures all “good” queries (safety)
4. “Good” RC and RA can express the same queries! (equivalence = Codd's theorem)

Relational Calculus (RC)

- RC is, essentially, first-order logic (FOL) over the schema relations
  - A query has the form "find all tuples (x_1,...,x_k) that satisfy an FOL condition"
- RC is a declarative query language
  - Meaning: a query is not defined by a sequence of operations, but rather by a condition that the result should satisfy

Queries and the connection to logic

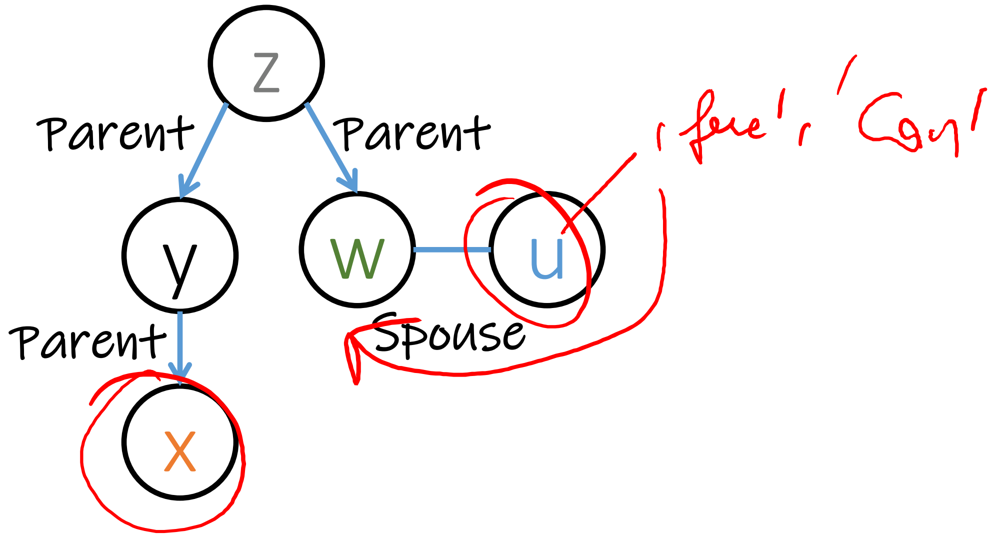
- Why logic?
- A crash course in FOL
- Relational Calculus (RC)
  - Syntax and Semantics
  - Domain RC (DRC) vs Tuple RC (TRC)
  - Domain Independence and Safety

RC Query

Person(id, gender, country)
Parent(parent, child)
Spouse(person1, person2)

assume symmetric relation
$(a,b)\in \text{Spouse} \Leftrightarrow (b,a)\in \text{Spouse}$

$$\{(x,u) \mid \text{Person}(u, \text{'female'}, \text{'Canada'}) \wedge \exists z,y\,[\text{Parent}(z,y) \wedge \text{Parent}(y,x) \wedge \exists w\,[\text{Parent}(z,w) \wedge y \neq w \wedge (u=w \vee \text{Spouse}(u,w))]]\}$$



Which relatives does this query find?

?

RC Symbols

- **Constant values:** a, b, c, female, Canada, ...
  - Values that may appear in table cells (optionally with quotation marks)
- **Variables:** x, y, z, ...
  - Range over the values that may appear in table cells
- **Relation symbols:** R, S, T, Person, Parent, ...
  - Each with a specified arity
  - Will be fixed by the relational schema at hand
  - No attribute names, only attribute positions (= unnamed perspective)!
- Unlike general FOL, no function symbols!

RC Formulas (atomic and non-atomic)

- Atomic formulas:
  - $R(t_1,\ldots,t_k)$
    - R is a k-ary relation, Each t_i is a variable or a constant
    - Semantically it states that $(t_1,\ldots,t_k)$ is a tuple in R
  - x op u
    - x is a variable, u is a variable/constant, op is one of >, <, =, ≠
    - Simply binary predicates, predefined interpretation
- Formula:
  - Atomic formula
  - If ϕ and ψ are formulas then these are formulas:

    $\phi \land \psi \quad \phi \lor \psi \quad \phi \rightarrow \psi \quad \phi \rightarrow \psi \quad \neg\phi \quad \exists x\, \phi \quad \forall x\, \phi$

Person(x, 'female', 'Canada')

x=y, y≠w, z>5, z='female'

Free Variables

$\exists x \, (x = y)$

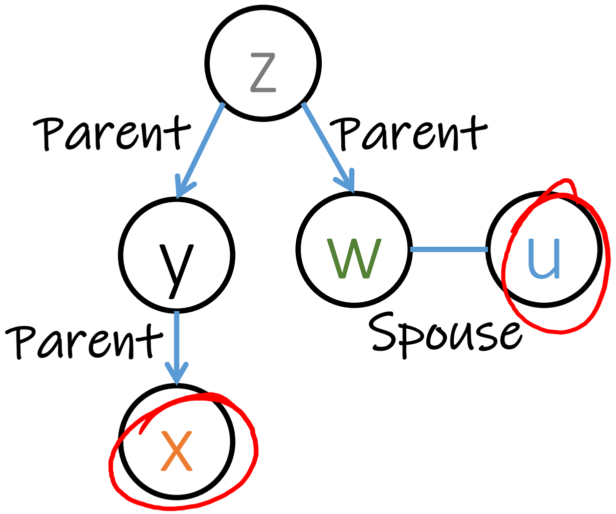
$\uparrow$

- Intuitively: free variable are not bound to quantifiers
- Formally:
  - A free variable of an atomic formula is a variable that occurs in the atomic formula
  - A free variable of $\varphi \wedge \psi, \varphi \vee \psi, \varphi \rightarrow \psi$ is a free variable of either φ or ψ
  - A free variable of $\neg\varphi$ is a free variable of φ
  - A free variable of $\exists x\ \varphi$ and $\forall x\ \varphi$ is a free variable y of φ such that $y \neq x$
- We write $\varphi(x_1,\ldots,x_k)$ to state that $x_1,\ldots,x_k$ are the free variables of formula φ (in some order)

Back to our earlier example

This is a formula!

$$
\begin{aligned}
&\text{Person}(u, \text{'female'}, \text{'Canada'}) \land \\
&\exists z,y\ \big[\text{Parent}(z,y) \land \text{Parent}(y,x) \land \\
&\qquad \exists w\ [\text{Parent}(z,w) \land y \neq w \land (u=w \lor \text{Spouse}(u,w))]\ \big]
\end{aligned}
$$

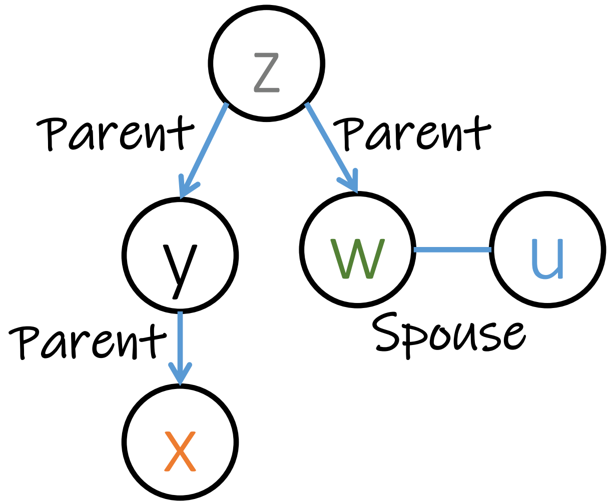


What are the free variables?

?

Back to our earlier example

$$\text{Person}(u, \text{'female'}, \text{'Canada'}) \land$$
$$\exists z,y \left[\text{Parent}(z,y) \land \text{Parent}(y,x) \land\right.$$
$$\left.\exists w \left[\text{Parent}(z,w) \land y \neq w \land (u=w \lor \text{Spouse}(u,w))\right]\right]$$

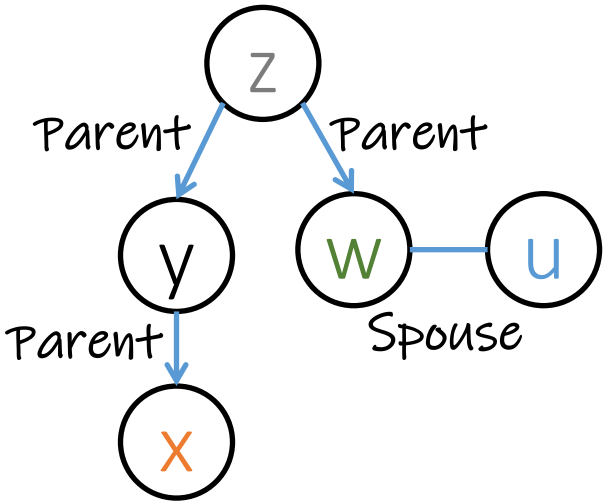


Notation:

$\varphi(x,u)$ / CanadianAunt(u,x)

RC query

$$\{\ (x,u)\ |\ \text{Person}(u, \text{'female'}, \text{'Canada'}) \land$$
$$\exists z,y\ [\text{Parent}(z,y) \land \text{Parent}(y,x) \land$$
$$\exists w\ [\text{Parent}(z,w) \land y \neq w \land\ (u = w \lor \text{Spouse}(u,w))]\ ]\ \}$$



$$\{\ (x_1,\ldots,x_k)\ |\ \varphi(x_1,\ldots,x_k)\ \}$$

$$\varphi(x,u)\ /\ \text{CanadianAunt}(u,x)$$

Relation Calculus Query

some condition on the variables
$COND(x_1,\ldots,x_k)$

- An RC query is an expression of the form

$$\{\ (x_1,\ldots,x_k) \mid \varphi(x_1,\ldots,x_k)\ \}$$

where $\varphi(x_1,\ldots,x_k)$ is an RC formula

- An RC query is *over* a relational schema $\mathbf{S}$ if all the relation symbols belong to $\mathbf{S}$ (with matching arities)

Queries and the connection to logic

- Why logic?
- A crash course in FOL
- Relational Calculus (RC)
  - Syntax and Semantics
  - Domain RC (DRC) vs Tuple RC (TRC)
  - Domain Independence and Safety

DRC vs. TRC

- There are two common variants of RC:
  - DRC (Domain RC): attributes as sets (what we have seen so far)
  - TRC (Tuple RC): tuples as sets
- DRC applies vanilla FO: terms interpreted as attribute values, relations have arity but no attribute names (= unnamed perspective)
- TRC is more "database friendly": terms interpreted as tuples with named attributes
- There are easy conversions between the two formalisms

DRC vs. TRC

Schema: $R(A,B)$

domain variables range over the domain

$\{\ (x,y)\ |\ R(x,y) \wedge y>2\ \}$

$\{\ r\ |\ \exists r[r \in R \wedge r.B>2\ \}$

$\{\ r\ |\ \exists r \in R[r.B>2]\}$

predicate

tuple variables range over relations

R	A	B
	1	7
	1	8

$\{\ (x)\ |\ \exists y[R(x,y) \wedge y>2]\ \}$

?

DRC vs. TRC

Schema: R(A,B)

domain variables range over the domain

$\{ (x,y) \mid R(x,y) \land y>2 \}$

$\{ r \mid r \in R \land r.B>2 \}$

$\{ r \mid \exists r \in R[r.B>2]\}$

predicate

tuple variables range over relations

$\{ (x) \mid \exists y[R(x,y) \land y>2] \}$

$\{ q \mid \exists r \in R[q.A=r.A \land r.B>2] \}$

Which are here bound and which are free variables ?

DRC vs. TRC

Schema: R(A,B)

domain variables range over the domain

$\{ (x,y) \mid R(x,y) \wedge y>2 \}$

$\{ r \mid r \in R \wedge r.B>2 \}$

$\{ r \mid \exists r \in R[r.B>2] \}$

predicate

$\{ q \mid r \in R[q.A=r.A \wedge q.B=r.B \wedge r.B>2] \}$

tuple variables range over relations

free bound

$\{ (x) \mid \exists y[R(x,y) \wedge y>2] \}$

$\{ q \mid \exists r \in R[q.A=r.A \wedge r.B>2] \}$

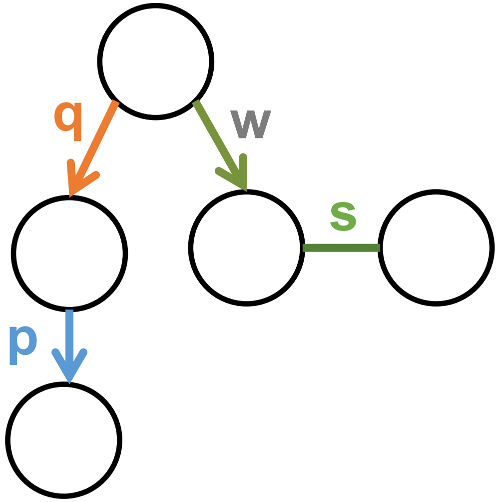
free bound

Our Example in TRC

Person(id, gender, country)
Parent(parent, child)
Spouse(person1, person2)

optionally "t(nephew, aunt)"

$$\{ t \mid \exists a \in \text{Person}\ [a.\text{gender} = \text{'female'} \land a.\text{country} = \text{'Canada'}] \land \\ \exists p,q,w \in \text{Parent}\ [p.\text{child} = t.\text{nephew} \land q.\text{child} = p.\text{parent} \land \\ w.\text{parent} = q.\text{parent} \land w.\text{child} \neq q.\text{child} \land a.\text{id} = t.\text{aunt} \land \\ (w.\text{child} = a.\text{id} \lor \exists s\ [s \in \text{Spouse} \land s.\text{person1} = w.\text{child} \land s.\text{person2} = a.\text{id}])\ ]\ ] \}$$



tuple variables like in SQL instead of domain variables: $\{t \mid COND(t)\}$

often used short forms:

$\forall x \in R[\varphi]$ same as $\forall x[x \in R \Rightarrow \varphi]$

$\exists x \in R[\varphi]$ same as $\exists x[x \in R \land \varphi]$

Based on material by Benny Kimelfeld and Oded Shmueli for 236363 Database Management Systems, Technion, 2018. However, notice I prefer and follow here the notation of [Ramakrishnan, Gehrke' 03] and [Elmasri, Navathe'15] of using **a.country = 'Canada'**, instead of the alternative notation **a[country]='Canada'** used by [Silberschatz, Korth, Sudarshan 2010]

Different TRC notations

Likes(person, drink)
Frequents(person, bar)
Serves(bar, drink)

331

Find persons that frequent <u>only</u> bars that serve <u>only</u> drinks they like.
(Find persons who like all drinks that are served in all the bars they visit.)
(Find persons for which there does not exist a bar they frequent that serves a drink they do not like.)

$$\{q(\text{person}) \mid \exists f \in \text{Frequents}\,[f.\text{person}=q.\text{person} \wedge \neg(\exists f2 \in \text{Frequents}\,[f2.\text{person}=f.\text{person} \wedge \neg(\exists l \in \text{Likes}, \exists s \in \text{Serves}\,[l.\text{drink}=s.\text{drink} \wedge f2.\text{bar}=s.\text{bar} \wedge f2.\text{person}=l.\text{person}])])]\}$$

my preferred notation

$$\{F.\text{person} \mid F \in \text{Frequents}.(\nexists F2 \in \text{Frequents}.(F2.\text{person}=F.\text{person} \wedge (\nexists L \in \text{Likes}, \nexists S \in \text{Serves}.(L.\text{drink}=S.\text{drink} \wedge F2.\text{bar}=S.\text{bar} \wedge F2.\text{person}=L.\text{person}))))\}$$

my earlier preferred notation

$$\{t\text{: Person} \mid \exists f \in \text{Frequents}\,[t(\text{Person})=f(\text{Person}) \wedge \neg\exists f2 \in \text{Frequents}\,[F2(\text{person})=F(\text{person}) \wedge \neg(\exists l \in \text{Likes}\ \exists s \in \text{Serves})\,[l(\text{Drink})=s(\text{Drink}) \wedge f2(\text{Bar})=s(\text{Bar}) \wedge f2(\text{Person})=l(\text{Person})]]]\}$$

[Deutsch 2019]

$$\{f.\text{Person} \mid \text{Frequents}(f) \text{ AND } (\text{NOT}(\exists f2)(\text{Frequents}(f2) \text{ AND } f2.\text{person}=f.\text{person} \wedge (\text{NOT}(\exists l)(\exists s)(\text{Likes}(l) \text{ AND } \text{Serves}(s) \text{ AND } l.\text{drink}=s.\text{drink} \text{ AND } f2.\text{bar}=s.\text{bar} \text{ AND } f2.\text{person}=l.\text{person}))))\}$$

[Elmasri 2015]

$$\{\mu^{(1)} \mid (\exists \rho^{(2)})\,(\text{Frequents}(\rho) \wedge \rho[1]= \mu[1] \wedge \neg((\exists \lambda^{(2)})(\text{Frequents}(\lambda) \wedge \lambda[1] = \rho[1] \wedge \neg((\exists \nu^{(2)})(\exists \theta^{(2)})(\text{Likes}(\nu) \wedge \text{Serves}(\theta) \wedge \nu(2)= \theta(2) \wedge \lambda(2)=\theta(1) \wedge \lambda(1)=\nu(1))))))\}$$

[Ullman 1988]

$$\{P \mid \exists F \in \text{Frequents}\,(F.\text{person}=P.\text{person} \wedge \neg\exists F2 \in \text{Frequents}(F2.\text{person}=F.\text{person} \wedge \neg(\exists L \in \text{Likes}\ \exists S \in \text{Serves}\,(L.\text{drink}=S.\text{drink} \wedge F2.\text{bar}=S.\text{bar} \wedge F2.\text{person}=L.\text{person}))))\}$$

[Ramakrishnan 2003]

Queries and the connection to logic

- Why logic?
- A crash course in FOL
- Relational Calculus (RC)
  - Syntax and Semantics
  - Domain RC (DRC) vs Tuple RC (TRC)
  - Domain Independence and Safety

Intuition for what we are trying to avoid

Q_1: $\{\ (x) \mid \neg B(x)\ \}$

$B = \{3, 4\}$

1) What's the answer to Q_1?

Intuition for what we are trying to avoid

Q_1: { (x) | ¬B(x) }

$B = \{3, 4\}$

$\text{Dom} = \mathbb{N}_1^{100}$

1) What's the answer to Q_1?

2) What now?

Intuition for what we are trying to avoid

Q_1: $\{\ (x) \mid \neg B(x)\ \}$

$B = \{3, 4\}$ 1) What's the answer to Q_1?

$\mathrm{Dom} = \mathbb{N}_1^{100}$ 2) What now?

Q_2: $\{\ (x) \mid A(x) \wedge \neg B(x)\ \}$

$A = \{1, 2, 3\}$ 3) What's the answer to Q_2?

Intuition for what we are trying to avoid

Q_1: $\{ (x) \mid \neg B(x) \}$

$B = \{3, 4\}$ 1) What's the answer to Q_1?

$\text{Dom} = \mathbb{N}_1^{100}$ 2) What now?

Q_2: $\{ (x) \mid A(x) \land \neg B(x) \}$

$A = \{1, 2, 3\}$ 3) What's the answer to Q_2?

$\text{Dom} = \mathbb{N}_1^{1000}$ 4) What now?

Intuition for what we are trying to avoid

Q_1: $\{ (x) \mid \neg B(x) \}$

$B = \{3, 4\}$ 1) What's the answer to Q_1?

$\text{Dom} = \mathbb{N}_1^{100}$ 2) What now?

That's easy to see, but it gets more complicated ☹

Q_2: $\{ (x) \mid A(x) \wedge \neg B(x) \}$

$A = \{1, 2, 3\}$ 3) What's the answer to Q_2?

$\text{Dom} = \mathbb{N}_1^{1000}$ 4) What now?

Q_2 is "domain-independent", i.e. we don't care whether Dom is $\mathbb{N}_1^{100}$ or $\mathbb{N}_1^{1000}$. We only care about the database D:

A

a
1
2
3

B

a
3
4

Bringing in the Domain

- Let $\mathbf{S}$ be a schema, D a database over $\mathbf{S}$, and Q an RC query over $\mathbf{S}$
- Then D gives an unambiguous interpretation for the underlying FOL
  - Predicates $\longrightarrow$ relations; constants copied; no functions

Is this true ?

Bringing in the Domain

- Let $\mathbf{S}$ be a schema, D a database over $\mathbf{S}$, and Q an RC query over $\mathbf{S}$
- ~~Then D gives an unambiguous interpretation for the underlying FOL~~
  - Predicates $\longrightarrow$ relations; constants copied; no functions
  - Not yet! We need to answer first: What is the domain?

- The active domain $\mathbf{ADom}$ (of D and Q) is the set of all the values that occur in either D or Q
- The query Q is evaluated over D with respect to a domain $\mathbf{Dom}$ that contains the active domain ($\mathbf{Dom} \supseteq \mathbf{ADom}$)
- Denote by $\mathrm{Q}^{\mathbf{Dom}}(\mathrm{D})$ the result of evaluating Q over D relative to the domain $\mathbf{Dom}$

Domain Independence

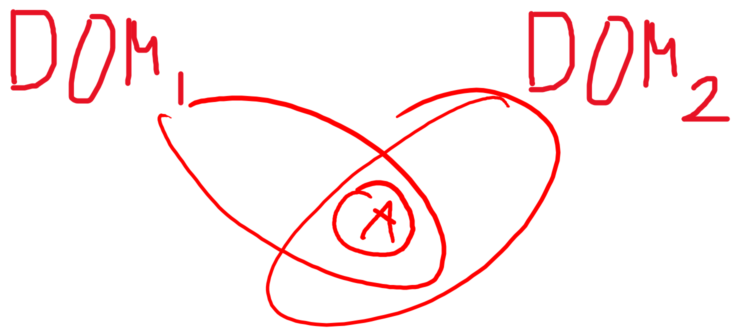
- Let $\mathbf{S}$ be a schema, and let Q be an RC query over $\mathbf{S}$
- We say that Q is domain independent if for every database D over $\mathbf{S}$ and ...

How could we continue the definition ?

Domain Independence

- Let $\mathbf{S}$ be a schema, and let Q be an RC query over $\mathbf{S}$
- We say that Q is domain independent if for every database D over $\mathbf{S}$ and every two domains $\mathbf{Dom_1}$ and $\mathbf{Dom_2}$ that contain the active domain, we have:

$$Q^{\mathbf{Dom}_1}(D) = Q^{\mathbf{Dom}_2}(D) = Q^{\mathbf{ADom}}(D)$$



Bad News...

- We would like be able to tell whether a given RC query is domain independent, and then reject "bad queries"
- Alas, this problem is undecidable!
  - That is, there is no algorithm that takes as input an RC query and returns true iff the query is domain independent

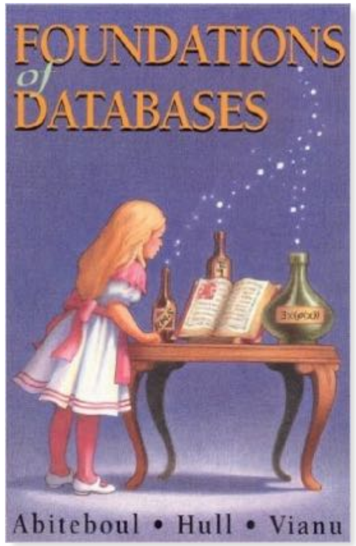
Good News

Domain-independent RC has an "effective syntax", that is:

- A syntactic restriction of RC in which every query is domain independent
- Restricted queries are said to be safe

- Safety can be tested automatically (and efficiently)
  - Most importantly, for every domain independent RC query there exists an equivalent safe RC query!

Safety



- We don't cover the formal definition of the safe syntax
- Details on the safe syntax can be found e.g. in [Alice'95]
- Example:
  - Every variable x_i is guarded by $R(x_1,\ldots,x_k)$
  - In $\exists x\ \varphi$, the variable x should be guarded by φ
  - In $\psi \wedge (x=y)$, the variable x is guarded iff either x or y is guarded by ψ
  - …

The basic idea of these definitions is to ensure that every free variable in the query is somehow bound to an element in the active domain of the database or, in the presence of nontrivial operations, to one of a finite number of domain elements. In the absence of operations, this is typically done by ensuring that every free or existentially quantified variable in a query occurs positively in its scope, every universally quantified variable occurs negatively in its scope, and that the same free variables occur in every component of a disjunction. For example, the query $\{x \mid P(x) \wedge \forall y(Q(x, y) \rightarrow R(x, y))\}$ is safe according to these ideas.

[Alice'95] Abiteboul, Hull, Vianu. Foundations of Databases, 1995. Chapter 5.4 Syntactic Restrictions for Domain Independence. http://webdam.inria.fr/Alice/
An accessible overview of issues involving safety is: Topor, Safety and Domain Independence, Encyclopedia of Database Systems. https://doi.org/10.1007/978-0-387-39940-9_1255

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

ADom = {1, 2, 3, 'female', 'Canada'}

Dom = ADom ∪ {'elefant', 'car', 'lemon', π, ...}

$\{ (x) \mid \neg\text{Person}(x, \text{'female'}, \text{'Canada'}) \}$?

$\{ (x,y) \mid \exists z\, [\text{Spouse}(x,z) \wedge y=z] \}$?

$\{ (x,y) \mid \exists z\, [\text{Spouse}(x,z) \wedge y\neq z] \}$?

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

What are example fixes: ?

$\{\ (x) \mid \neg \text{Person}(x, \text{'female'}, \text{'Canada'})\ \}$ Not DI

$\{\ (x,y) \mid \exists z\ [\text{Spouse}(x,z) \land y=z]\ \}$?

$\{\ (x,y) \mid \exists z\ [\text{Spouse}(x,z) \land y \neq z]\ \}$?

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

What are example fixes:

$\land \exists y,z.\text{Person}(x,y,z)$

$\land \text{Person}(x,\_,\_)$

$\land \text{Person}(x,\_,\text{'Canada'})$

$\land$ x='Alice' or x='Beatrice'

$$\{ (x) \mid \neg \text{Person}(x, \text{'female'}, \text{'Canada'}) \}$$

Not DI

$$\{ (x,y) \mid \exists z\, [\text{Spouse}(x,z) \land y=z] \}$$

?

$$\{ (x,y) \mid \exists z\, [\text{Spouse}(x,z) \land y \neq z] \}$$

?

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

What are example fixes:

$\wedge\ \exists y,z.\text{Person}(x,y,z)$
$\wedge\ \text{Person}(x,\_,\_)$
$\wedge\ \text{Person}(x,\_,\text{'Canada'})$
$\wedge\ x=\text{'Alice'} \text{ or } x=\text{'Beatrice'}$

$\{\ (x) \mid \neg\text{Person}(x, \text{'female'}, \text{'Canada'})\ \}$ **Not DI**

$\{\ (x,y) \mid \exists z\ [\text{Spouse}(x,z) \wedge y=z]\ \}$ **DI**

same as $\{(x,y) \mid \text{Spouse}(x,y)\} = \text{Spouse}(x,y)$

$\{\ (x,y) \mid \exists z\ [\text{Spouse}(x,z) \wedge y\neq z]\ \}$ **?**

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

What are example fixes:

$\wedge\ \exists y,z.\text{Person}(x,y,z)$
$\wedge\ \text{Person}(x,\_,\_)$
$\wedge\ \text{Person}(x,\_,\text{'Canada'})$
$\wedge\ x=\text{'Alice'}$ or $x=\text{'Beatrice'}$

$$\{ (x) \mid \neg\text{Person}(x, \text{'female'}, \text{'Canada'}) \}$$

Not DI

$$\{ (x,y) \mid \exists z\, [\text{Spouse}(x,z) \wedge y=z] \}$$

DI

same as $\{(x,y) \mid \text{Spouse}(x,y)\} = \text{Spouse}(x,y)$

$$\{ (x,y) \mid \exists z\, [\text{Spouse}(x,z) \wedge y\neq z] \}$$

?

D: Spouse('Alice','Bob')

ADom={'Alice','Bob'} → {('Alice','Alice')}

Dom ⊇ ADom   Dom={'Alice','Bob','Charly'} → {('Alice','Alice'), ('Alice','Charly')}

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

What are example fixes:

$\wedge\ \exists y,z.\text{Person}(x,y,z)$
$\wedge\ \text{Person}(x,\_,\_)$
$\wedge\ \text{Person}(x,\_,\text{'Canada'})$
$\wedge\ x=\text{'Alice'} \text{ or } x=\text{'Beatrice'}$

$\{\ (x)\ |\ \neg\text{Person}(x, \text{'female'}, \text{'Canada'})\ \}$

Not DI

$\{\ (x,y)\ |\exists z\ [\text{Spouse}(x,z) \wedge y=z]\ \}$

DI

same as $\{(x,y)\ |\ \text{Spouse}(x,y)\} = \text{Spouse}(x,y)$

$\{\ (x,y)\ |\exists z\ [\text{Spouse}(x,z) \wedge y\neq z]\ \}$

Not DI

D: Spouse('Alice','Bob')

ADom={'Alice','Bob'} → {('Alice','Alice')}

Dom ⊇ ADom Dom={'Alice','Bob','Charly'} → {('Alice','Alice'), ('Alice','Charly')}

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

$$\{\, (x) \mid \exists z,w \; \text{Person}(x,z,w) \land \exists y\, [\neg \text{Likes}(x,y)] \,\}$$

$$\{\, (x) \mid \exists z,w \; \text{Person}(x,z,w) \land \forall y\, [\neg \text{Likes}(x,y)] \,\}$$

$$\{\, (x) \mid \exists z,w \; \text{Person}(x,z,w) \land \forall y\, [\neg \text{Likes}(x,y)] \land \exists y\, [\neg \text{Likes}(x,y)] \,\}$$

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

D

Person('Alice', 'female', 'Canada')	Likes('Alice', 'Beate')
Person('Beate', 'female', 'Canada')	Likes('Alice', 'Cecile')
Person('Cecile', 'female', 'Canada')	Likes('Alice', 'Alice')

ADom = ?

$$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \land \exists y\ [\neg \text{Likes}(x,y)] \}$$

$$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \land \forall y\ [\neg \text{Likes}(x,y)] \}$$

$$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \land \forall y\ [\neg \text{Likes}(x,y)] \land \exists y\ [\neg \text{Likes}(x,y)] \}$$

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

D

Person('Alice', 'female', 'Canada')	Likes('Alice', 'Beate')
Person('Beate', 'female', 'Canada')	Likes('Alice', 'Cecile')
Person('Cecile', 'female', 'Canada')	Likes('Alice', 'Alice')

ADom = {'Alice', 'Beate', 'Cecile', 'female', 'Canada'}

$$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \land \exists y\ [\neg \text{Likes}(x,y)] \}$$

$$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \land \forall y\ [\neg \text{Likes}(x,y)] \}$$

$$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \land \forall y\ [\neg \text{Likes}(x,y)] \land \exists y\ [\neg \text{Likes}(x,y)] \}$$

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

D

Person('Alice', 'Alice', 'Alice')	Likes('Alice', 'Beate')
Person('Beate', 'Beate', 'Beate')	Likes('Alice', 'Cecile')
Person('Cecile', 'Cecile', 'Cecile')	Likes('Alice', 'Alice')

ADom = {'Alice', 'Beate', 'Cecile'}

Dom = {'Alice', 'Beate', 'Cecile', 'Dora'}

$$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \wedge \exists y\ [\neg \text{Likes}(x,y)] \}$$?

$$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)] \}$$?

$$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)] \wedge \exists y\ [\neg \text{Likes}(x,y)] \}$$?

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

D

Person('Alice', 'Alice', 'Alice')	Likes('Alice', 'Beate')
Person('Beate', 'Beate', 'Beate')	Likes('Alice', 'Cecile')
Person('Cecile', 'Cecile', 'Cecile')	Likes('Alice', 'Alice')

ADom = {'Alice', 'Beate', 'Cecile'}

Dom = {'Alice', 'Beate', 'Cecile', 'Dora'}

Example fix:

$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \wedge \exists y\ [\neg \text{Likes}(x,y)] \}$?

Alice is in the output if Dom ⊃ ADom (e.g., Dora is in Dom)

Not DI

$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)] \}$?

$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)] \wedge \exists y\ [\neg \text{Likes}(x,y)] \}$?

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

D

Person('Alice', 'Alice', 'Alice')	Likes('Alice', 'Beate')
Person('Beate', 'Beate', 'Beate')	Likes('Alice', 'Cecile')
Person('Cecile', 'Cecile', 'Cecile')	Likes('Alice', 'Alice')

ADom = {'Alice', 'Beate', 'Cecile'}

Dom = {'Alice', 'Beate', 'Cecile', 'Dora'}

Person(y,_,_)

Example fix: $\wedge\ \exists u,v\ [\text{Person}(y,u,v)]$

$\{\ (x)\ |\ \exists z,w\ \text{Person}(x,z,w) \wedge \exists y\ [\neg \text{Likes}(x,y)]\ \}$ **Not DI**

Alice is in the output if Dom ⊃ ADom (e.g., Dora is in Dom)

$\{\ (x)\ |\ \exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)]\ \}$?

$\{\ (x)\ |\ \exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)] \wedge \exists y\ [\neg \text{Likes}(x,y)]\ \}$?

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

D

Person('Alice', 'Alice', 'Alice') Likes('Alice', 'Beate')
Person('Beate', 'Beate', 'Beate') Likes('Alice', 'Cecile')
Person('Cecile', 'Cecile', 'Cecile') Likes('Alice', 'Alice')

ADom = {'Alice', 'Beate', 'Cecile'}

Dom = {'Alice', 'Beate', 'Cecile', 'Dora'}

Person(y,_,_)

Example fix: … $\wedge\ \exists u,v$ [Person(y,u,v)]

$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \wedge \exists y\ [\neg \text{Likes}(x,y)] \}$ **Not DI**

Alice is in the output if Dom ⊃ ADom (e.g., Dora is in Dom)

$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)] \}$ **DI**

x never occurs in Likes(x,_): Beate, Cecile

$\{ (x) \mid \exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)] \wedge \exists y\ [\neg \text{Likes}(x,y)] \}$?

Which One is Domain Independent?

Person(id, gender, country)
Likes(person1, person2)
Spouse(person1, person2)

D

Person('Alice', 'Alice', 'Alice')	Likes('Alice', 'Beate')
Person('Beate', 'Beate', 'Beate')	Likes('Alice', 'Cecile')
Person('Cecile', 'Cecile', 'Cecile')	Likes('Alice', 'Alice')

ADom = {'Alice', 'Beate', 'Cecile'}

Dom = {'Alice', 'Beate', 'Cecile', 'Dora'}

Person(y,_,_)

Example fix: ... $\wedge\ \exists u,v\ [\text{Person}(y,u,v)]$

$\{\ (x)\ |\exists z,w\ \text{Person}(x,z,w) \wedge \exists y\ [\neg \text{Likes}(x,y)]\ \}$ **Not DI**

Alice is in the output if Dom ⊃ ADom (e.g., Dora is in Dom)

$\{\ (x)\ |\exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)]\ \}$ **DI**

x never occurs in Likes(x,_): Beate, Cecile

$\{\ (x)\ |\exists z,w\ \text{Person}(x,z,w) \wedge \forall y\ [\neg \text{Likes}(x,y)] \wedge \cancel{\exists y\ [\neg \text{Likes}(x,y)]}\ \}$ **DI**

implication (absorption) if Dom ≠ Ø, which is necessary for there to be Person(x,_,_)

What is the meaning of following unsafe expressions?

$\{\, x \mid \exists y.\ R(x) \,\}$?

$\{\, x \mid x \geq 10 \,\}$?

$\{\, x \mid \forall y\ R(x,y) \,\}$?

What is the meaning of following unsafe expressions?

$\{ x \mid \exists y.\ R(x) \}$ logically equivalent to $\{ x \mid R(x) \} = R(x)$

$\{ x \mid x \geq 10 \}$?

$\{ x \mid \forall y\ R(x,y) \}$?

What is the meaning of following unsafe expressions?

$\{x \mid \exists y.\, R(x)\}$ logically equivalent to $\{x \mid R(x)\} = R(x)$

$\{x \mid x \geq 10\}$ What if Dom=$\mathbb{N}$? DI: $\{x \mid A(x) \land x \geq 10\}$

$\{x \mid \forall y\, R(x,y)\}$?

What is the meaning of following unsafe expressions?

$\{\, x \mid \exists y.\ R(x) \,\}$ logically equivalent to $\{\, x \mid R(x) \,\} = R(x)$

$\{\, x \mid x \geq 10 \,\}$ What if Dom=$\mathbb{N}$? DI: $\{\, x \mid A(x) \wedge x \geq 10 \,\}$

$\{\, x \mid \forall y\ R(x,y) \,\}$

D: R('a','a')
ADom={'a'}
Dom={'a','Chile'}

DI ?: $\{\, x \mid \forall y\ [A(y) \rightarrow R(x,y)] \,\}$

 ?

What is the meaning of following unsafe expressions?

$\{\, x \mid \exists y.\, R(x) \,\}$ logically equivalent to $\{ x \mid R(x) \} = R(x)$

$\{\, x \mid x \geq 10 \,\}$ What if Dom=ℕ? DI: $\{\, x \mid A(x) \wedge x \geq 10 \,\}$

$\{\, x \mid \forall y\, R(x,y) \,\}$

D: R('a','a')
ADom={'a'}
Dom={'a','Chile'}

DI ?: $\{\, x \mid \forall y\, [A(y) \rightarrow R(x,y)] \,\}$

what if relation A is empty?

1. always true for A=∅

$\{\, x \mid \forall y\, [\neg A(y) \vee R(x,y)] \,\}$

What is the meaning of following unsafe expressions?

$\{ x \mid \exists y.\, R(x) \}$ logically equivalent to $\{ x \mid R(x) \} = R(x)$

$\{ x \mid x \geq 10 \}$ What if Dom=$\mathbb{N}$? DI: $\{ x \mid A(x) \land x \geq 10 \}$

$\{ x \mid \forall y\, R(x,y) \}$

D: R('a','a')
ADom={'a'}
Dom={'a','Chile'}

not DI: $\{ x \mid \forall y\, [A(y) \rightarrow R(x,y)] \}$

what if relation A is empty?

Neutral element for $\forall$ is TRUE

$\sum$: $0 + x = x$

$\prod$: $1 \cdot x = x$

$\lor$: FALSE $\lor\, x = x$ $\exists$: $x_1 \lor x_2 \lor \ldots \lor$ FALSE

$\land$: TRUE $\land\, x = x$ $\forall$: $x_1 \land x_2 \land \ldots \land$ TRUE

MIN: $\mathrm{MIN}(\infty, x) = x$

1. always true for A=Ø

$\{ x \mid \forall y\, [\neg A(y) \lor R(x,y)] \}$

2. alternative way to see that

What is the meaning of following unsafe expressions?

$\{ x \mid \exists y.\, R(x) \}$ logically equivalent to $\{ x \mid R(x) \} = R(x)$

$\{ x \mid x \geq 10 \}$ What if Dom=$\mathbb{N}$? DI: $\{ x \mid A(x) \land x \geq 10 \}$

$\{ x \mid \forall y\, R(x,y) \}$ D: R('a','a')
ADom={'a'}
Dom={'a','Chile'}

not DI: $\{ x \mid \forall y\, [A(y) \rightarrow R(x,y)] \}$

what if relation A is empty?

Neutral element for $\forall$ is TRUE

another way to see it:

$\forall y\, [R(y)]$

true if the domain for y is empty set!

$\forall y\, [y \in Dom \rightarrow R(y)]$

1. always true for $A=\emptyset$

$\{ x \mid \forall y\, [\neg A(y) \lor R(x,y)] \}$

← 2. alternative way to see that

What is the meaning of following unsafe expressions?

$\{ x \mid \exists y.\ R(x) \}$ logically equivalent to $\{ x \mid R(x) \} = R(x)$

$\{ x \mid x \geq 10 \}$ What if Dom=$\mathbb{N}$? DI: $\{ x \mid A(x) \land x \geq 10 \}$

$\{ x \mid \forall y\ R(x,y) \}$ not DI: $\{ x \mid \forall y\ [A(y) \rightarrow R(x,y)] \}$

DI: ?

What is the meaning of following unsafe expressions?

$\{\, x \mid \exists y.\, R(x) \,\}$ logically equivalent to $\{ x \mid R(x) \} = R(x)$

$\{\, x \mid x \geq 10 \,\}$ What if Dom=ℕ? DI: $\{\, x \mid A(x) \land x \geq 10 \,\}$

$\{\, x \mid \forall y\, R(x,y) \,\}$ not DI: $\{\, x \mid \forall y\, [A(y) \rightarrow R(x,y)] \,\}$

or A(x) or $\exists z[R(x,z) \land \ldots]$

DI: $\{\, x \mid R(x,\_) \land \forall y\, [A(y) \rightarrow R(x,y)] \,\}$

$\{\, x \mid R(x,\_) \land \nexists y\, [A(y) \land \neg R(x,y)] \,\}$

We will see this last expression again next class ☺
In the meantime, try for yourself. How to write in TRC?

Another example on domain-independence

More interestingly, if the domain is the set of natural numbers and the only operation on the domain is linear order, then the query

$$Q_4 = \{x \mid \forall y(\Delta(y) \rightarrow x > y) \\ \wedge \forall y(y < x \rightarrow \exists z(\Delta(z) \wedge z \geq y)),$$

where $\Delta(y)$ is true if and only if y is in the active domain of the database, defines the smallest integer greater than all the active domain elements, and is hence finite but not domain independent [7].