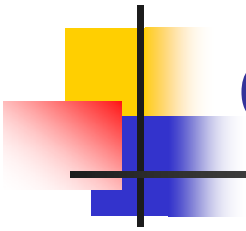
CS589 Principles of DB Systems
Fall 2008
Lecture 4e:
Logic (Model-theoretic view of a DB)

Lois Delcambre
lmd@cs.pdx.edu
503 725-2405



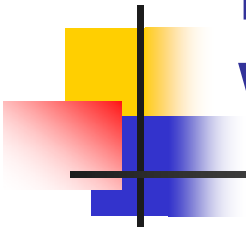
Goals for today

- Review propositional logic (including truth assignment)
- Review 1st order predicate calculus (including theory vs. interpretation)
- Introduce the model-theoretic description of a relational database and consider how tuple calculus, domain calculus, and datalog use that view.
- Mention the proof-theoretic description of a relational database.

Propositional Logic

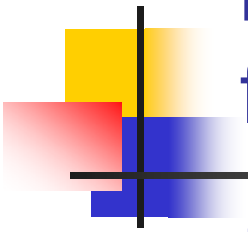
- Two constants: *true* and *false*
- Infinite set of variables (called atomic formulae in our textbook) each variable can take the value *true* or *false*
- Well-formed propositional formulas:
  - An atom is either a variable or a constant (there are only two constants, *true* or *false*)
  - An atom is a propositional formula
  - If F is formula, then $\neg$F is a propositional formula
  - If F and G are propositional formulas, then F $\wedge$ G and F $\vee$ G (and F $\rightarrow$ G and F $\leftrightarrow$ G) are propositional formulas

Propositional Formulas
Which ones are well-formed? (wffs?)



1. false
2. true $\wedge$ false
3. p $\wedge$ p
4. p $\wedge$ q $\wedge$ p
5. $\neg$ $\neg$p
6. p $\neg$ p
7. $\neg$false $\wedge$ true
8. p $\wedge$ $\neg$ q $\wedge$ p $\vee$ p $\wedge$ $\neg$ q $\wedge$ p $\vee$

Do we know whether formulas are true or false?



- The constant “true” is true
- The constant “false” is false
- Tautologies are (always) true
- Contradictions are (always) false
- On the previous slide, which wffs are tautologies? Which ones are contradictions?

- What about everything else? It has variables.
- We need a truth assignment to indicate which propositional variables are true and which are false.

Truth assignments

- A truth assignment $\mathcal{A}$ is a function that maps a set of propositional variables to {true, false}
  - The formula "true" has the truth value *true* under any truth assignment. That is $\mathcal{A}$(true)=true, for any $\mathcal{A}$. Similarly for false; the value is always false.
  - The formula "$\varphi \wedge \psi$" is true under $\mathcal{A}$ if and only if $\mathcal{A}(\varphi)$ is true and $\mathcal{A}(\psi)$ is true
  - The formula "$\varphi \vee \psi$" is true under $\mathcal{A}$ if and only if $\mathcal{A}(\varphi)$ is true or $\mathcal{A}(\psi)$ is true
  - The formula "$\neg\varphi$" is true under $\mathcal{A}$ iff $\mathcal{A}(\varphi)$ is false

For each truth assignment, we can find out whether a formula is true or false

Consider the well-formed formula: $\neg q \wedge p$

For a truth assignment $\mathcal{A}_1$ where
$\mathcal{A}_1(q)$ is true and
$\mathcal{A}_1(p)$ is false.

Then this formula is false.

If we use a truth assignment $\mathcal{A}_2$ where
$\mathcal{A}_2(q)$ is false and
$\mathcal{A}_2(p)$ is true

then what is the truth value of this formula?

For any propositional wff, we can tell if it's a tautology using a truth table

	(p	∧	q)	∧	r
	true		*true*		
	true		*false*		
	false		*true*		
	false		*false*		

Note: a truth table exhaustively considers all truth assignments.

1. Enter all possible combinations of truth values for p and q.

For any propositional wff, we can tell if it's a tautology using a truth table

	(p	∧	q)	∧	r
	true	*true*	*true*		
	true	*false*	*false*		
	false	*false*	*true*		
	false	*false*	*false*		

2. Compute the truth values for p ∧ q.

For any propositional wff, we can tell if it's a tautology using a truth table

	(p	∧	q)	∧	r
	true	*true*	*true*		*true*
	true	*false*	*false*		*true*
	false	*false*	*true*		*true*
	false	*false*	*false*		*true*
	true	*true*	*true*		*false*
	true	*false*	*false*		*false*
	false	*false*	*true*		*false*
	false	*false*	*false*		*false*

3. Enter all possible combinations of what we have so far – with all possible combinations of truth values for r. This doubles the size of the table

For any propositional wff, we can tell if it's a tautology using a truth table

	(p	∧	q)	∧	r
	true	*true*	*true*	true	*true*
	true	*false*	*false*	*false*	*true*
	false	*false*	*true*	*false*	*true*
	false	*false*	*false*	*false*	*true*
	true	*true*	*true*	*false*	*false*
	true	*false*	*false*	*false*	*false*
	false	*false*	*true*	*false*	*false*
	false	*false*	*false*	*false*	*false*

compare

4. Compute the truth value for the final ∧.
This is obviously not a tautology.

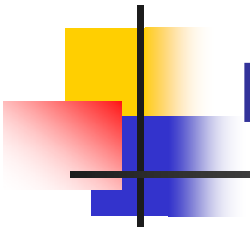
Logical Implication

- A formula φ logically implies formula ψ if for every truth assignment $\mathcal{A}$, if $\mathcal{A}(\varphi)$ is true, then $\mathcal{A}(\psi)$ is true.
- We write $\varphi \models \psi$

  Example: $p \models (\neg(\neg p))$

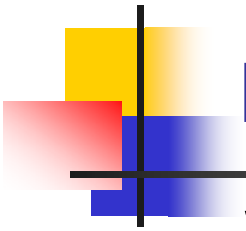
- If $\varphi \models \psi$ and $\psi \models \varphi$ then we say φ and ψ are logically equivalent, written $\psi \equiv \varphi$.

  Example: $p \equiv (\neg(\neg p))$



Definitions

- A wff is *satisfiable* if there is at least one truth assignment that makes it true.
- A wff is *unsatisfiable* if it is not satisfiable.
- A wff is valid if every truth assignment makes it true.
- A valid propositional wff is a *tautology*.
- The negation of a valid wff is *unsatisfiable*.
- The negation of a tautology is *unsatisfiable*.

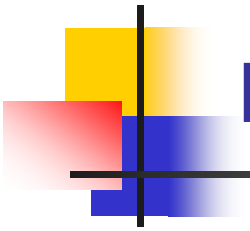


Discussion questions

We know that a Datalog program is a set of well-formed formulas. And we know that a tuple or domain calculus query uses a well-formed formula to describe the set membership criteria.

We can use logic to determine:

1. Whether a query is satisfiable.
2. Whether a query is a tautology.
3. Whether a query is a contradiction.
4. Whether one query logically implies another one.



First-Order theory

- An infinite set of variables
- A (possibly empty) set of constant symbols plus {*true*, *false*}
- A (possibly empty) set of n-ary predicates, for $n \geq 0$
- A (possibly empty) set of n-ary functions, for $n \geq 1$.
- Sometimes equality is included as one of the predicates, sometimes not.
- A set of axioms.

Well-formed, first-order formulas

- Variables, individual constants, and $f(t_1, \ldots, t_n)$ are terms. (Nothing else is a term.)
- An atomic formula is *true* or *false* or $R(t_1, \ldots, t_n)$ where t_i, i=1,n, are terms.
- Every atomic formula is a wff.
- If φ and ψ are wffs, then
  - $\neg\varphi$ is a wff,
  - $\varphi \wedge \psi$ is a wff,
  - $\varphi \vee \psi$ is a wff,
  - and if x is a variable, then $(\forall x)\varphi$ and $(\exists x)\varphi$ are wffs.
- (Nothing else is a wff.)

Example of a First-order theory

- No constant symbols
- No functions
- One predicate, p
- Two proper axioms:
  - $(\forall x)\neg p(x,x)$
  - $(\forall x)(\forall y)(\forall z)p(x,y)\wedge p(y,z) \rightarrow p(x,z)$

Interpretation for a 1st Order Theory

- An interpretation $I = (U, C, P, F)$ where
  - U is a nonempty set of elements (the domain)
  - C is a function that maps the constants of the theory into elements of U
  - P maps each n-ary predicate (from the theory) into an n-place relation over U
  - F maps each n-ary function (from the theory) into an function from $U^n \rightarrow U$
- Variables range over the set U
- An interpretation is a model (of the theory) iff all of the axioms are true.

Example interpretations

Axioms: $(\forall x)\neg p(x,x)$

$(\forall x)(\forall y)p(x,y)\wedge p(y,z) \rightarrow p(x,z)$

Interpretation 1: U = non-negative integers, p is the predicate not-equal.

Interpretation 2: U = non-negative integers, p is less-than-or-equal-to.

Interpretation 3: U = integers, p is less-than.

Which of these interpretations are models?

Define another model for this theory.

Another first-order theory

- One constant symbol {zero}
- One function, +
- One predicate, equality
- Proper axioms:
  - $(\forall x)(\forall y)(\forall z)\ \ x+(y+z) = (x+y)+z$
  - $(\forall x)\ x+zero = x$
  - $(\forall x)(\exists y)\ x+y=zero$
  - $(\forall x)(\forall y)\ x=y \rightarrow y=x$
  - $(\forall x)\ x=x$
  - $(\forall x)(\forall y)(\forall z)\ x=y \rightarrow (x=z \rightarrow y=z)$
  - $(\forall x)(\forall y)(\forall z)\ (x=y) \rightarrow ((x+z=y+z)\ AND\ (z+y=z+x))$

Interpretations for theory
(on previous slide) (Exercise to do at home)

Interpretation 1:

U = integers

zero is mapped to 0

+ is regular addition

= is regular equality

Interpretation 2:

U = non-negative integers

zero is mapped to 0

+ is regular addition

= is regular equality

Interpretation 3:

U = integers modulo 5

zero is mapped to 0

+ is modulo 5 addition

= is regular equality

Interpretation 4:

U = integers

zero is mapped to 1

+ is regular addition

= is regular equality

Which ones are models of the theory?

Logic Axioms and Rules of Inference

φ_1 φ_2 and φ_3 well-formed formulas of a 1st order theory

Axioms:

$\varphi_1 \rightarrow (\varphi_2 \rightarrow \varphi_1)$

$(\varphi_1 \rightarrow (\varphi_2 \rightarrow \varphi_3)) \rightarrow ((\varphi_1 \rightarrow \varphi_2) \rightarrow (\varphi_2 \rightarrow \varphi_3))$

$(\neg\varphi_2 \rightarrow \neg\varphi_1) \rightarrow ((\neg\varphi_2 \rightarrow \varphi_1) \rightarrow \varphi_2))$

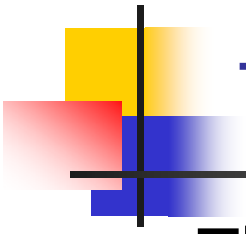
$((\forall x)\varphi_1(x)) \rightarrow \varphi_1$ (t), if t is a term in the theory and x is free in $\varphi_1(x)$.

$((\forall x)(\varphi_1 \rightarrow \varphi_2)) \rightarrow ((\varphi_1 \rightarrow (\forall x)\varphi_2))$ if x is not free in φ_1

Rules of Inference:

Modus ponens: φ_2 follows from φ_1 and $\varphi_1 \rightarrow \varphi_2$

Generalization: $(\forall x)\varphi_1$ follows from φ_1



Terminology

- The axioms shown on the previous slide are called the *logical axioms*.
- If you add extra axioms to your theory, they are called *proper axioms* or *non-logical axioms*.
- If you don't have any proper axioms, then the theory is called 1st order predicate calculus.
- If you show that's something is logically valid, then you are using a model-theoretic argument.
- If you prove a theorem, then you are using a proof-theoretic argument. A proof uses axioms and rules of inference, only.

How do we use relational databases for relational calculus and for Datalog?

- An interpretation $I = (U, C, P, F)$ where
  - U is a nonempty set of elements (the domain)
  - C is a function that maps the co… theory into elements of U
  - P maps each n-ary predicate (from the theory) into an n-place relation over U
  - F maps each n-ary function (from the theory) into an function from $U^n \rightarrow U$
- Variables range over the set U
- An interpretation is a model (of the theory) iff all of the axioms are true.

Union of all domains.

No constant symbols needed in the theory.

Function-free

What about relational databases?

- An interpretation $I = (U, C, P, F)$ where
  - U is a nonempt
  - C is a function
    theory into elements of U
  - P maps each n-ary predicate (from the theory) into an n-place relation over U
  - F maps each n-ary function (from the theory) into an fu
- Variabl
- An inte iff all of th

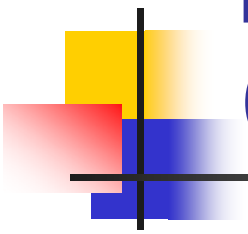
In order to have an interpretation, we need to describe (we need to write down) all of th tuples where a given predicate is true.

Think about a relational database. What does it store for us? It (exactly) stores the tuples that belong in a particular table. A relational database is (exactly) an interpretation of a very simple 1st order theory with no functions and no constant symbols.

Relational DB as an interpretation

"In [domain calculus] a relational database is considered to be an interpretation of a first-order theory. The domain of the interpretation is a superset of the active domain of the database and the relations in the database are the extensions of the relation symbols of the database schema. A query in the domain relational calculus is essentially checking [i.e., finding] whether [i.e., where] the database is a model of the first-order formula in the query." P. 108, textbook

Proof-Theoretic View of a Relational DB (by Raymond Reiter)



He set out to describe the entire DB content in the theory. (This then requires no interpretation. Everything is done using proofs.)

He introduced constant symbols for every constant in the domains of the database.

He introduced one axiom for every tuple in the relational database. (Each tuple is a “fact.”)

Then he figured out the additional axioms needed to make it work out.

1. Unique name assumption (Identical things are equal, everything else is not equal.)
2. Domain closure axioms (Every value must be equal to one of the constant symbols in the theory.)
3. Completion axioms (Every atomic predicate must be equal to one of the axioms that represent the tuples in the DB. This is essentially the closed world assumption. (If it is not in the DB, then assume it is false.))

Proof-theoretic view of a relational DB

The entire DB is described in a 1st order theory (and all results are proofs).

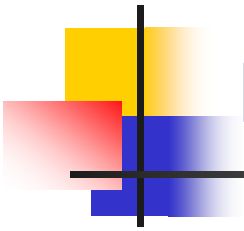
A query (which is a wff) is a theorem to be proved (rather than a wff to be evaluated in an interpretation).

Since the DB is in the theory (with “facts” as axioms) we can add any 1st order statements, as axioms, to our database.

For example we could add:

Age(John, 21) v Age(John 22)

without requiring that the tuple for John hold either of these ages.



Reference

"Towards a Logical Reconstruction of Relational Database Theory" by Raymond Reiter, a chapter in *On Conceptual Modeling*, edited by Michael Brodie, John Mylopoulos, and Joachim Schmidt, Springer-Verlag, 1984, pp. 191-233.

Reiter was one of many database researchers exploring the relationship between logic and databases. This work (in the 1980s) led to the definition and exploration of Datalog.

Results – 1st order theory

- Every instance of a tautology is a theorem that uses only axioms 1, 2, and 3, and MP.
- Completeness: Every theorem of a 1st order predicate calculus is logically valid.
  If you can prove a theorem…then it's logically valid (true in all models).
- Any logically valid wff is a theorem.
- In any predicate calculus, theorems = valid wffs.