In the Lecture Series Introduction to Database Systems

Calculus

Presented by Stéphane Bressan

Relational Calculi

There are two calculi:

- Domain relational calculus (DRC);
- T-uple relational calculus (TRC).

- DRC and TRC are query languages
- They are both based on logic

Learning Objectives

- Write and understand queries in Domain Relational Calculus
- Write and understand queries in T-uple Relational Calculus

Propositional Logic

"Aristotle is Greek"

"Aristotle is Greek and Alexander is Persian"

"Aristotle is not Persian"

"Alexander is Macedonian or Persian"

Propositional Logic

"Roxane is Bactrian or not Bactrian"

"Olympias is Greek and is not Greek"

"Olympias is Greek implies Alexander is Greek"

"Roxane is Bactrian implies Roxane is Bactrian"

Propositional Logic

"Olympias is Greek implies Alexander is Greek"

"Alexander is not Greek implies Olympias is not Greek"

Semantics of Propositional Logic

The semantic of propositional logic is defined by truth tables

A	B	$(A \vee B)$	$(A \wedge B)$	$(A \Rightarrow B)$	$\neg (A)$
T	T	T	T	T	F
F	T	T	F	T	T
T	F	T	F	F	F
F	F	F	F	T	T

$(\neg A \vee B)$

First Order Logic: Predicates

greek(aristotle)

greek(X)

mother(olympias, alexander)

mother(X, Y)

First Order Logic

$\exists$ X greek(X)

$\exists$ X mother(olympias, X)

$\exists$ X $\exists$ Y mother(Y, X)

$\exists$ Y $\exists$ X mother(Y, X)

First Order Logic

$\forall$ X greek(X)

$\forall$ Y $\exists$ X mother(X, Y)

$\exists$ X $\forall$ Y mother(X, Y)

First Order Logic

$\forall$ X $\forall$ Y ((mother(X, Y) $\wedge$ greek(X))
$\Rightarrow$ greek(Y))

Syntax of First Order Logic

First order logic consists of formulae built from predicates, constants *(lower case)* and variables *(upper case)*, and connectives:*

- (F $\wedge$ G)
- (F $\vee$ G)
- $\neg$ (F)
- (F $\Rightarrow$ G)

And **quantifiers:** $\forall$ and $\exists$

Variables can be quantified (bound) or free

Semantics of Predicate Logic

To avoid confusion we agree that:

A variable is quantified once at most.

If a variable is quantified in a formula, it cannot appear outside of the scope of its quantifier.

$\forall X\ (\underbrace{\exists X\ (\overbrace{\text{mother}(X, Y)}))}$

Different scopes

$\exists Y\ (\overbrace{\text{mother}(X, Y))} \wedge \text{father}\ (Z, Y)$

Semantics of Predicate Logic

$\neg \forall X\ F$ is equivalent to $\exists X \neg F$

$\neg \exists X\ F$ is equivalent to $\forall X \neg F$

*(*Here F represents a formula)*

Calculus

- A Calculus defines formulae and their meaning
- T-uple Relational Calculus: variables range over t-uples (TRC)
- Domain Relational Calculus: variables range over values (DRC)

Calculus

How to represent the set of integers 2, 3, and 4?

In extension:

$\{2, 3, 4\}$

In Intention (set-builder notation, comprehension, abstraction):

$\{ X \mid X \in N \wedge 1 < X \wedge X < 5\}$

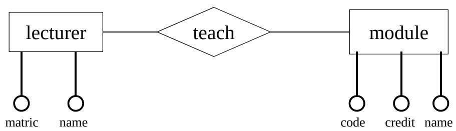
Calculus: Where is the Truth?

- The truth is in the database
- If a relation Mother in the database has a t-uple mother(olympias, alexander) then Olympias is the mother of Alexander
- Otherwise it is not *(closed world assumption)*

T-uple relational Calculus

Example

- lecturer(matric, name)
- module(code, name, credit),
- teach(matric, code)



Example

- $\{T \mid T \in lecturer\}$
- $\{T \mid \exists T1\ (T1 \in lecturer \wedge T = T1)\}$
- $\{T \mid \exists T1\ (T1 \in Lecturer$
  $\wedge$ T.matric = T1.matric
  $\wedge$ T.name = T1.name$)\}$
  by CONVENTION!

SELECT *
FROM
LECTURER

SELECT
matric, name
FROM lecturer

Syntax of T-uple Relational Calculus

- Parenthesis can be omitted if non ambiguous

Example

- $\{T \mid \exists T1\ T1 \in lecturer$
  $\wedge$ T.name = T1.name$\}$

select only the name

Example

- $\{T \mid \exists T1\ T1 \in lecturer$
  $\wedge$ T1.name = "Smith"
  $\wedge$ T.matric = T1.matric$\}$

SELECT matric
FROM lecturer
WHERE name='Smith'

Example

- $\{T \mid \exists T1\ \exists T2\ \exists T3$
  $T1 \in lecturer$
  $\wedge\ T2 \in module$
  $\wedge\ T3 \in teach$
  $\wedge$ T1.matric = T3.matric
  $\wedge$ T2.code = T3.code
  $\wedge$ T2.credit < 3
  $\wedge$ T.lec_name = T1.name
  $\wedge$ T.mod_name = T2.name$\}$

FROM

WHERE

SELECT

Example

SELECT
  T1.name as lec_name,
  T2.name as mod_name
FROM lecturer T1, module T2,teach t3
WHERE T1.matric = T3.matric
  AND T2.code = T3.code
  AND T2.credit < 3

Example (incorrect)

$P \Rightarrow Q$

$P' \vee Q$

- {T | ∃ T1
  T1 ∈ lecturer
  ∧ ((∃ T2 T2 ∈ module) ⇒ (∃ T3
  T3 ∈ teach
  ∧ T1.matric = T3.matric
  ∧ T2.code = T3.code))
  ∧ T.name = T1.name}

WRONG (Scope)

all quantifiers must be on top

Example (correct)

De morgans

$(\exists T_2\ P(x))'$

$\forall T_2\ (P(x))'$

- {T | ∃ T1 ∀ T2 ∃ T3
  T1 ∈ lecturer
  ∧ (T2 ∈ module ⇒ (
  T3 ∈ teach
  ∧ T1.matric = T3.matric
  ∧ T2.code = T3.code))
  ∧ T.name = T1.name}

$\exists P\ P(x) \Rightarrow \exists Q\ Q(x)$

$(\exists P\ P(x))' \vee (\exists Q\ Q(x))$

$\forall P\ (P(x))' \vee \exists Q\ Q(X)$

$\forall P\ \exists Q\ (P(x))' \vee Q(x)$

Attention!

- ∃ T ∈ r (F)
  means ∃ T (T ∈ r ∧ F)

- ∃ T1 ∈ r ∃ T2 ∈ s (F)
  means ∃T1 ∃ T2 (T1 ∈ r ∧ T2 ∈ s ∧ F)

(*Here F represents a formula)

Attention!

- ∀ T ∈ r (F)
  means ∀ T (T ∈ r ⇒ F)

- ∀ T1 ∈ r ∀ T2 ∈ s F
  means
  ∀ T1 ∀ T2 ((T1 ∈ r ∧ T2 ∈ s) ⇒ F)

(*Here F represents a formula)

Semantics of T-uple Relational Calculus

- Example
  - ∀ T ∈ module (T.credit > 1)

  means
  - ∀ T (T∈ module ⇒ T.credit > 1)

  and does not mean
  - ∀ T (T ∈ module ∧ T.credit > 1)

Semantics of T-uple Relational Calculus

{T | F(T)}

- An **interpretation** I is a mapping of a formula to {true, false}
- An interpretation is defined by a mapping I of the free variable (T) of a formula (F(T)) to a t-uple t of constants
- $t \in R$ is true if and only if the t_uple t is in the instance of R in the database
- A **model** is an interpretation for which the formula is true

Safety of Queries in T-uple Relational Calculus

$\{T \mid T \notin \text{lecturer}\}$

(“mycat”, 22, “red”) is not a lecturer, any t-uple in the world maybe an answer if it is not already in the lecturer relation.

Safety of Queries in T-uple Relational Calculus

$\{T \mid \exists\, T1\ T1 \in \text{lecturer}$
$\quad \wedge\ T1.\text{matric} \neq \text{‘1234’}$
$\quad \wedge\ T.\text{name} = T1.\text{name}\}$

Safety of Queries in T-uple Relational Calculus

A query is **safe** if the set of t-uples in the answer is a subset of the set of t-uples that can be constructed from the constants explicitly referenced directly (they appear in the query) or indirectly (they appear in a relation mentioned in the query) in the query.

When you can map the relation to a set of tables

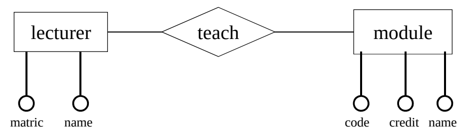
Safety

We consider only safe queries

Domain Relational Calculus

Example

- lecturer(matric, name)
- module(code, name, credit),
- teach(matric, code)



Example

Tuple constructor

- {<X> | ∃Y lecturer(X, Y)} → Only print Matric → SELECT Matric FROM lecturer

  matric / name

- {<X> | ∃Y lecturer(X, Y) ∧ Y = “john”} → SELECT matric FROM lecturer WHERE name='john'

  ⇕ → Same

- {<X> | lecturer(X, “john”)}

- How do you express
  SELECT * FROM Lecturer

Example

Find the names of lecturers teaching a module with less than 2 credits. Print the names of the lecturers and the names of the corresponding modules.

Example

Cannot quantify the 2 variables here

{<LN, MN> | ∃ M ∃ C ∃ Cr
Lecturer(M, LN)
∧ Module(C, MN, Cr)
∧ Teach(M, C)
∧ Cr < 2}

COMPACT!!

Example SQL

Split   Split

- {<LN, MN> | ∃ M1 ∃ M2 ∃ C1 ∃ C2 ∃ Cr
  Lecturer(M1, LN)
  ∧ Module(C1, MN, Cr)
  ∧ Teach(M2, C2)
  ∧ C1 = C2 ∧ M1 = M2 ∧ Cr < 2}

→ Added on: Get equality

Example SQL

- SELECT
  Lecturer.lecName,
  Module.moduleName
  FROM Lecturer, Module, Teach
  WHERE Lecturer.matric=Teach.matric
  AND Module.code = Teach.code
  AND Module.credit < 2

Example

{<M1, M2, N> |
lecturer(M1, N)
∧ lecturer(M2, N)
∧ M1 <> M2}

Look for diff lecturers with the same name

Example

{<M1, M2, N1> | ∃N2
lecturer(M1, N1)
∧ lecturer(M2, N2)
∧ M1 <> M2
∧ N1 = N2}

Example

Find the names of the lecturers teaching all modules

{<N> | ∃ M ∀ C ∀ MN ∀ Cr
(lecturer(M, N) ∧
(Module(C, MN, Cr) ⇒ teach(M, C))) }

If there is a module, then the lecturer must teach it

Semantics of Domain Relational Calculus

$$\{\langle X_1, \ldots, X_n\rangle \mid F(X_1, \ldots, X_n)\}$$

- An **<u>interpretation</u>** **I** is a mapping of each formula to **{true, false}**
- An interpretation is defined by a mapping I of the free variables ($X_1, \ldots, X_n$) of a formula ($F(X_1, \ldots, X_n)$) to constants
- $R(c_1, \ldots, c_n)$ is true if and only if the t_uple $\langle c_1, \ldots, c_n\rangle$ is in the instance of R in the database
- A **<u>model</u>** is an interpretation for which the formula is true

Example

Find the names of the lecturers teaching all modules:

{<N> | ∃ M ∀ C ∀ MN ∀ Cr
(lecturer(M, N) ∧
(Module(C, MN, Cr) ⇒ teach(M, C))) }

Example

∃ M ∀ C ∀ MN ∀ Cr

We are looking for values of N such that the formula below is true for **<u>SOME</u>** value of M and, <u>for that value of M</u>, for **<u>ALL</u>** values of C, MN and Cr :

(lecturer(M, N) ∧
(Module(C, N, Cr) ⇒ teach(M, C))) }

Example

If <M, N> if a lecturer and

<C, N, Cr> is not a module the formula is true!!!!

(lecturer(M, N) $\wedge$

(Module(C, N, Cr) $\Rightarrow$ teach(M, C))) }

Example

If <M, N> if a lecturer and

<C, N, Cr> is a module, and M teaches C the formula is true

(lecturer(M, N) $\wedge$

(Module(C, N, Cr) $\Rightarrow$ teach(M, C))) }

Example

The formula is false only if

<M,N> is not a lecturer or if

<C, N, Cr> is a module, and M does not teach C

(lecturer(M, N) $\wedge$

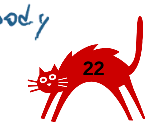
(Module(C, N, Cr) $\Rightarrow$ teach(M, C))) }

Safety of Queries in the Domain Calculus

{<M,N> | $\neg$ lecturer(M,N)}

* All variables here MUST NOT be quantified

* Varibles in body must be quantified



Safety of Queries in the Domain Calculus

- A query is **safe** if the set of t-uples in the answer is a subset of the set of t-uples that can be constructed from the constants explicitly referenced directly (they appear in the query) or indirectly (they appear in a relation mentioned in the query) in the query.

Safety

- We consider only safe queries

Credits

The content of this lecture is based on chapter 3 of the book "Introduction to database Systems" By S. Bressan and B. Catania, McGraw Hill publisher

Clipart and media are licensed from Microsoft Office Online Clipart and Media