CS 4092 Database Design and Development (DDD)

03: Formal Relational Query Language (Part 2)

Seokki Lee

Slides are adapted from:

Relational Calculus

- Two flavors
  - Tuple relational calculus (TRC)
  - Domain relational calculus (DRC)
- Calculus has variables, constants, comparison ops, logical connectivities, and quantifiers.
  - TRC: variables range over (i.e., get bound to) ***tuples***
  - DRC: variables range over ***domain*** elements
- Expressions in the calculus are called ***formulas***.
- Answer tuple is an assignment of constants to variables that make the formula evaluate to ***true***.

Topics

- Formal Relational Query Language
  - Relational Algebra
  - **Tuple Relational Calculus**
  - Domain Relational Calculus

Tuple Relational Calculus

- A ***non-procedural*** query language, where each query is of the form

  $$\{t \mid P(t)\}$$

- ***Answer*** is the set of all tuples t such that predicate (formula) P is true for *t.*
- Notations
  - t is a *tuple variable*
  - $t[A]$ (or t.A) denotes the value of tuple t on attribute A
  - $t \in r$ denotes that tuple t is in relation r

Predicate Calculus Formula

1. Set of attributes and constants
2. Set of comparison operators (op): (e.g., $<$, $\leq$, $=$, $\neq$, $>$, $\geq$)
3. Set of logical connectives: and ($\wedge$), or (v), not ($\neg$)
4. Implication ($\Rightarrow$): x $\Rightarrow$ y, if x is true, then y is true

$$x \Rightarrow y \equiv \neg x \vee y$$

5. Set of quantifiers:
   - $\exists\, t \in r\,(Q\,(t\,)) \equiv$ "there exists" a tuple t in relation r such that predicate $Q\,(t\,)$ is true
   - $\forall t \in r\,(Q\,(t\,)) \equiv Q$ is true "for all" tuples t in relation r

Predicate Calculus Formula

- Implication ($\Rightarrow$): $x \Rightarrow y$, if x is true, then y is true

$$x \Rightarrow y \equiv \neg x \vee y$$

  - If x is true and y is false, then the implication evaluates to false.
  - If x is not true, then we do not care about y.
    - The expression is always true.

Tuple Relational Calculus

- Atomic fomulas
  - $t \in r$ or $t[A] \in r$
  - $t[A]$ op $s[b]$
  - $t[A]$ op *constants*
- A formula can be
  - An atomic formula
  - $\neg p$, $p \wedge q$, $p \vee q$ where p and q are formulas
  - $\exists t\ (p(t))$
  - $\forall t\ (p(t))$
- Formula is recursively defined
  - Starting with simple **atomic formulas**
    - Get tuples from relations or make comparisons of values
  - Build bigger and better formulas using the **logical connectives**

Tuple Relational Calculus

- The use of quantifiers $\exists\, t$ and $\forall t$ in a formula is said to ***bind*** t in the formula.
  - A variable that is not bound is called ***free*** variable.
- Important restriction
  - The variable *t* that appears to the left of ‘ l ’ must be the ***only*** free variable in the formula.
  - All other tuple variables must be bound using a quantifier.

Example Queries

- Find the *ID, name, dept_name, salary* for instructors whose salary is greater than $80,000

$$\{t \mid P(t)\}$$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Example Queries

- Find the *ID, name, dept_name, salary* for instructors whose salary is greater than $80,000

selection

$$\{t \mid t \in instructor \wedge \boxed{t[salary] > 80000}\}$$

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Example Queries

- Find the *ID, name, dept_name, salary* for instructors whose salary is greater than $80,000

$$\{t \mid t \in \textit{instructor} \wedge \underbrace{\boxed{t[\textit{salary}] > 80000}}_{\text{selection}}\}$$

- As in the previous query, but output only the *ID* attribute value

$$\{t \mid \exists\, s \in \text{instructor}\ (\underbrace{\boxed{t[\textit{ID}] = s[\textit{ID}]}}_{\text{projection}} \wedge s[\textit{salary}] > 80000)\}$$

Example Queries

- Find the *ID, name, dept_name, salary* for instructors whose salary is greater than $80,000

$$\{t \mid t \in instructor \land \overset{\text{selection}}{\boxed{t[salary] > 80000}}\}$$

- As in the previous query, but output only the *ID* attribute value

$$\{t \mid \exists\ s \in \text{instructor}\ (\overset{\text{projection}}{\boxed{t[ID] = s[ID]}} \land s[salary] > 80000)\}$$

- Notice that a relation on schema (*ID*) is implicitly defined by the query, because
  - t is not bound to any relation by the predicate
  - We implicitly state that t has an ID attribute (*t[ID] = s[ID]*)

Example Queries

- Find the names of all instructors whose department is in the Watson building

instructor

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

department

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Example Queries

- Find the names of all instructors whose department is in the Watson building

$$\{t \mid \exists s \in instructor\ (t[name] = s[name]$$
$$\wedge \exists u \in department\ (\boxed{u[dept\_name] = s[dept\_name]}$$
$$\wedge\ u[building] = \text{“Watson”}\ ))\}$$

join

instructor

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

department

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Example Queries

- Find the names of all instructors whose department is in the Watson building

"The parentheses control the scope of each quantifier's binding."

$$\{t \mid \exists s \in instructor\ (t[name] = s[name] \\ \land \boxed{\exists u \in department\ (u[dept\_name] = s[dept\_name] \\ \land\ u[building] = \text{"Watson"}\ )})\}$$

instructor

ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

department

dept_name	building	budget
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Example Queries

- Find the set of all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or both

section

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	B
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	B
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

Example Queries

- Find the set of all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or both

$$\{t \mid \exists s \in section\ (t[course\_id] = s[course\_id] \land s[semester] = \text{"Fall"} \land s[\text{year}] = 2009)$$
$$\lor \exists u \in section\ (t\ [course\_id] = u[course\_id] \land u[semester] = \text{"Spring"} \land u[\text{year}] = 2010)\}$$

section

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	B
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	B
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

Example Queries

- Can we do this?

$$\{t \mid \exists s \in section\ (t[course\_id] = s[course\_id] \wedge s[semester] = \text{“Fall”} \vee s[semester] = \text{“Spring”}) \wedge \exists u \in section\ (t[course\_id] = u[course\_id] \wedge u[year] = 2009 \vee u[year] = 2010)\}$$

section

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	B
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	B
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

Example Queries

- Find the set of all courses taught in the Fall 2009 semester, **and** in the Spring 2010 semester

$$\{t \mid \exists s \in section\ (t[course\_id] = s[course\_id] \land s[semester] = \text{“Fall”} \land s[\text{year}] = 2009) \land \exists u \in section\ (t[course\_id] = u[course\_id] \land u[semester] = \text{“Spring”} \land u[\text{year}] = 2010)\}$$

- Find the set of all courses taught in the Fall 2009 semester, **but not in** the Spring 2010 semester

$$\{t \mid \exists s \in section\ (t[course\_id] = s[course\_id] \land s[semester] = \text{“Fall”} \land s[\text{year}] = 2009) \land \neg \exists u \in section\ (t[course\_id] = u[course\_id] \land u[semester] = \text{“Spring”} \land u[\text{year}] = 2010)\}$$

Safety of Expressions

- It is possible to write tuple calculus expressions that generate infinite relations.
  - For example, $\{ t \mid \neg (t \in r) \}$ results in an infinite relation if the domain of any attribute of relation *r* is infinite
- To guard against the problem, we restrict the set of allowable expressions to safe expressions.
  - $\{ t \mid dom(\neg (t \in r)) \}$

Safety of Expressions

- An expression $\{t \mid P(t)\}$ in the tuple relational calculus is ***safe*** if every component of t appears in one of the relations, tuples, or constants that appear in P
  - NOTE: this is more than just a syntax condition.
    - E.g. $\{ t \mid t[A] = 5 \vee \textbf{true} \}$ is not safe

Safety of Expressions

- An expression $\{t \mid P(t)\}$ in the tuple relational calculus is ***safe*** if every component of t appears in one of the relations, tuples, or constants that appear in P
  - NOTE: this is more than just a syntax condition.
    - E.g. $\{ t \mid t[A] = 5 \vee$ **true** $\}$ is not safe --- it defines an infinite set with attribute values that do not appear in any relation or tuples or constants in P.

Corresponding Reading Materials

- Tuple Relational Calculus
  - Database System Concepts 6th Edition
    - Chapter 6.2

Topics

- Formal Relational Query Language
  - Relational Algebra
  - Tuple Relational Calculus
  - **Domain Relational Calculus**

Domain Relational Calculus

- A **non-procedural** query language equivalent in power to the tuple relational calculus
- Each query is an expression of the form:

$$\{ < x_1, x_2, \ldots, x_n > \mid P(x_1, x_2, \ldots, x_n) \}$$

  - $x_1, x_2, \ldots, x_n$ represent domain variables
    - Variables that range of **attribute values**
    - Variables appearing to the left of '|' must be the **only free variables**
  - P represents a formula similar to that of the predicate calculus
  - Tuples can be formed using < >
    - E.g., <'Einstein','Physics'>

Example Queries

- Find the *ID, name, dept_name, salary* for instructors whose salary is greater than \$80,000
  - $\{< i, n, d, s> \mid < i, n, d, s> \in instructor \land s > 80000\}$
- As in the previous query, but output only the *ID* attribute value
  - $\{< i> \mid \exists\ n,d,s\ (< i, n, d, s> \in instructor \land s > 80000)\}$

instructor

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

department

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Example Queries

- Find the names of all instructors whose department is in the Watson building

instructor

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

department

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Example Queries

- Find the names of all instructors whose department is in the Watson building
  - $\{< n > \mid \exists\, i, d, s\, (< i, n, d, s > \in instructor$
    $\wedge \exists\, \text{b, a}\, (< d, b, a> \in department\ \wedge\ b = \text{“Watson”}\,))\}$

instructor

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

department

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Example Queries

- Find the set of all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or both
  - $\{<c> \mid \exists\ a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section\ \wedge$
    $s = \text{“Fall”} \wedge y = 2009)$
    $\vee\ \exists\ a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section\ \wedge$
    $s = \text{“Spring”} \wedge y = 2010)\}$

section

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	B
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	B
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

Example Queries

- Find the set of all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or both
  - $\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge s = \text{"Fall"} \wedge y = 2009)$
    $\vee \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge s = \text{"Spring"} \wedge y = 2010)\}$
  - $\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge ((s = \text{"Fall"} \wedge y = 2009) \vee (s = \text{"Spring"} \wedge y = 2010))\}$

Example Queries

- Find the set of all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or both
  - $\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge s = \text{“Fall”} \wedge y = 2009) \vee \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge s = \text{“Spring”} \wedge y = 2010)\}$
  - $\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge ((s = \text{“Fall”} \wedge y = 2009) \vee (s = \text{“Spring”} \wedge y = 2010))\}$

- Find the set of all courses taught in the Fall 2009 semester, and in the Spring 2010 semester
  - $\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge s = \text{“Fall”} \wedge y = 2009) \wedge \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, t> \in section \wedge s = \text{“Spring”} \wedge y = 2010)\}$

Safety of Expressions

The expression:

$$\{ < x_1, x_2, \ldots, x_n > \mid P(x_1, x_2, \ldots, x_n)\}$$

is **safe** if all of the following hold:

1. All values that appear in tuples of the expression are values from *dom* (P) (that is, the values appear either as constants in P or in a tuple of a relation mentioned in P).
2. For every "there exists" subformula of the form $\exists\, x\,(P_1(x))$, the subformula is true if and only if there is a value of x in *dom* (P_1) such that $P_1(x)$ is true.
3. For every "for all" subformula of the form $\forall_x\,(P_1\,(x))$, the subformula is true if and only if $P_1(x)$ is true for all values x from *dom* (P_1).

Relationship between Relational Algebra and Tuple (Domain) Calculus

- **Codd's theorem**
  - Relational algebra and tuple calculus are equivalent in terms of expressiveness
- Every query expressible in relational algebra can also be expressed in tuple calculus and vice versa
- Since domain calculus is as expressive as tuple calculus, the same holds for the domain calculus

Why Relational Calculus Important?

- SQL without aggregation = Relational calculus expressions
- Calculus is an **alternative way** for expressing the same queries of RA
  - Relational algebra (RA) is more operational.
    - Useful as internal representation for query evaluation plans
  - Relational calculus is non-operational.
    - Users define queries in terms of what they want, not in terms of how to compute it
- Many equivalent algebra “implementations” possible for given calculus expression
- Not much value on practical but it offers theoratically grounded alternatives

Corresponding Reading Materials

- Domain Relational Calculus
  - Database System Concepts 6th Edition
    - Chapter 6.3