CMU SCS

Carnegie Mellon Univ.
Dept. of Computer Science
15-415 - Database Applications

15-415/Fall '2010

Lecture#7: *Relational calculus*

(slides from Christos Faloutsos)

CMU SCS

General Overview - rel. model

- history
- concepts
- Formal query languages
  - relational algebra
  - **rel. tuple calculus**
  - rel. domain calculus

CMU SCS

Overview - detailed

- rel. tuple calculus
  - why?
  - details
  - examples
  - equivalence with rel. algebra
  - more examples; 'safety' of expressions
- re. domain calculus + QBE

CMU SCS

Motivation

- Q: weakness of rel. algebra?
- A: procedural
  - describes the steps (ie., 'how')
  - (still useful, for query optimization)

CMU SCS

Solution: rel. calculus

- describes what we want
- two equivalent flavors: 'tuple' and 'domain' calculus
- basis for SQL and QBE, resp.

CMU SCS

Rel. tuple calculus (RTC)

- first order logic

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

'Give me tuples 't', satisfying predicate P - eg:

$$\{t \mid t \in STUDENT\}$$

CMU SCS

Details

- symbols allowed:

  $\wedge, \quad \vee, \quad \neg, \quad \Rightarrow$

  $>, \quad <, \quad =, \quad \neq, \quad \leq, \quad \geq,$

  $(, \quad), \quad \in$

- quantifiers $\quad \forall, \quad \exists$

Faloutsos CMU SCS 15-415 #7

CMU SCS

Specifically

- Atom

  $t \in TABLE$

  $t.attr \leq\geq const$

  $t.attr \leq\geq s.attr'$

Faloutsos CMU SCS 15-415 #8

CMU SCS

Specifically

- Formula:
  - atom
  - if P1, P2 are formulas, so are $P1 \wedge P2; P1 \vee P2 \ldots$
  - if P(s) is a formula, so are $\exists s(P(s))$

    $\forall s(P(s))$

Faloutsos CMU SCS 15-415 #9

CMU SCS

Specifically

- Reminders:
  - DeMorgan $\quad P1 \wedge P2 \equiv \neg(\neg P1 \vee \neg P2)$
  - implication: $\quad P1 \Rightarrow P2 \equiv \neg P1 \vee P2$
  - double negation:

$$\forall s \in TABLE \ \ (P(s)) \quad \equiv \quad \neg \exists s \in TABLE \ \ (\neg P(s))$$

'every human is mortal : no human is immortal'

Faloutsos CMU SCS 15-415 #10

CMU SCS

Reminder: our Mini-U db

STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

CLASS		
c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

TAKES		
SSN	c-id	grade
123	15-413	A
234	15-413	B

Faloutsos CMU SCS 15-415 #11

CMU SCS

Examples

- find all student records

$$\{t \mid t \in STUDENT\}$$

output tuple

of type 'STUDENT'

Faloutsos CMU SCS 15-415 #12

CMU SCS

Examples

- (selection) find student record with ssn=123

Faloutsos CMU SCS 15-415 #13

CMU SCS

Examples

- (selection) find student record with ssn=123

$$\{t \mid t \in STUDENT \wedge t.ssn = 123\}$$

Faloutsos CMU SCS 15-415 #14

CMU SCS

Examples

- (projection) find **name** of student with ssn=123

$$\cancel{\{t \mid t \in STUDENT \wedge t.ssn = 123\}}$$

Faloutsos CMU SCS 15-415 #15

CMU SCS

Examples

- (projection) find name of student with ssn=123

$$\{t \mid \exists s \in STUDENT(s.ssn = 123 \wedge t.name = s.name)\}$$

't' has only one column

Faloutsos CMU SCS 15-415 #16

CMU SCS

'Tracing'

$$\{t \mid \exists s \in STUDENT(s.ssn = 123 \wedge t.name = s.name)\}$$

t

Name
aaaa
....
jones
...
zzzz

s

STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

Faloutsos CMU SCS 15-415 #17

CMU SCS

Examples cont'd

- (union) get records of both PT and FT students

Faloutsos CMU SCS 15-415 #18

CMU SCS

Examples cont'd

- (union) get records of both PT and FT students

$$\{t \mid t \in FT\_STUDENT \quad \vee \\ t \in PT\_STUDENT\}$$

CMU SCS

Examples

- difference: find students that are not staff

(assuming that STUDENT and STAFF are union-compatible)

CMU SCS

Examples

- difference: find students that are not staff

$$\{t \mid t \in STUDENT \wedge \\ t \notin STAFF\}$$

CMU SCS

Cartesian product

- eg., dog-breeding: MALE x FEMALE
- gives all possible couples

MALE
name
spike
spot

x

FEMALE
name
lassie
shiba

=

M.name	F.name
spike	lassie
spike	shiba
spot	lassie
spot	shiba

CMU SCS

Cartesian product

- find all the pairs of (male, female)

$$\{t \mid \exists m \in MALE \wedge \\ \exists f \in FEMALE \\ t.m-name = m.name \wedge \\ t.f-name = f.name\}$$

CMU SCS

'Proof' of equivalence

- rel. algebra <-> rel. tuple calculus

CMU SCS

Overview - detailed

- rel. tuple calculus
  - why?
  - details
  - examples
  - equivalence with rel. algebra
  - **more examples**; ‘safety’ of expressions
- re. domain calculus + QBE

Faloutsos CMU SCS 15-415 #25

CMU SCS

More examples

- join: find names of students taking 15-415

Faloutsos CMU SCS 15-415 #26

CMU SCS

Reminder: our Mini-U db

STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

CLASS		
c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

TAKES		
SSN	c-id	grade
123	15-413	A
234	15-413	B

Faloutsos CMU SCS 15-415 #27

CMU SCS

More examples

- join: find names of students taking 15-415

$$\{t \mid \exists s \in STUDENT$$
$$\wedge \exists e \in TAKES\ (s.ssn = e.ssn \wedge$$
$$t.name = s.name \wedge$$
$$e.c-id = 15-415)\}$$

Faloutsos CMU SCS 15-415 #28

CMU SCS

More examples

- join: find names of students taking 15-415

$$\{t \mid \exists s \in STUDENT$$
$$\wedge \exists e \in TAKES\ (s.ssn = e.ssn \wedge$$ join
$$t.name = s.name \wedge$$ projection
$$e.c-id = 15-415)\}$$ selection

Faloutsos CMU SCS 15-415 #29

CMU SCS

More examples

- 3-way join: find names of students taking a 2-unit course

Faloutsos CMU SCS 15-415 #30

Reminder: our Mini-U db

STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

CLASS		
c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

TAKES		
SSN	c-id	grade
123	15-413	A
234	15-413	B

More examples

- 3-way join: find names of students taking a 2-unit course

$$\{t \mid \exists s \in STUDENT \wedge \exists e \in TAKES$$
$$\exists c \in CLASS(s.ssn = e.ssn \wedge$$
$$e.c-id = c.c-id \wedge$$
$$t.name = s.name \wedge$$
$$c.units = 2)\}$$

join

projection

selection

More examples

- 3-way join: find names of students taking a 2-unit course - in rel. algebra??

$$\pi_{name}(\sigma_{units=2}(STUDENT \bowtie TAKES \bowtie CLASS))$$

Even more examples:

- self -joins: find Tom's grandparent(s)

PC	
p-id	c-id
Mary	Tom
Peter	Mary
John	Tom

PC	
p-id	c-id
Mary	Tom
Peter	Mary
John	Tom

Even more examples:

- self -joins: find Tom's grandparent(s)

$$\{t \mid \exists p \in PC \wedge \exists q \in PC$$
$$(p.c-id = q.p-id \wedge$$
$$p.p-id = t.p-id \wedge$$
$$q.c-id = "Tom")\}$$

Hard examples: DIVISION

- find suppliers that shipped all the ABOMB parts

SHIPMENT	
s#	p#
s1	p1
s2	p1
s1	p2
s3	p1
s5	p3

$\div$

ABOMB
p#
p1
p2

$=$

BAD_S
s#
s1

Hard examples: DIVISION

- find suppliers that shipped all the ABOMB parts

$$
\begin{aligned}
&\{t \mid \forall p(p \in ABOMB \Rightarrow (\\
&\quad \exists s \in SHIPMENT(\\
&\qquad t.s\# = s.s\# \wedge \\
&\qquad s.p\# = p.p\#)))\}
\end{aligned}
$$

General pattern

- three equivalent versions:
  - 1) if it's bad, he shipped it

$$\{t \mid \forall p(p \in ABOMB \Rightarrow (P(t))\}$$

  - 2)either it was good, or he shipped it

$$\{t \mid \forall p(p \notin ABOMB \vee (P(t))\}$$

  - 3) there is no bad shipment that he missed

$$\{t \mid \neg\exists p(p \in ABOMB \wedge (\neg P(t))\}$$

a ⇒ b is the same as ¬a ∨ b

a \ b	T	F
T	T	F
F	T	T

- If a is true, b must be true for the implication to be true. If a is true and b is false, the implication evaluates to false.
- If a is not true, we don't care about b, the expression is always true.

More on division

- find (SSNs of) students that take all the courses that ssn=123 does (and maybe even more)

find students 's' so that

if 123 takes a course => so does 's'

More on division

- find students that take all the courses that ssn=123 does (and maybe even more)

$$
\begin{aligned}
&\{o \mid \forall t((t \in TAKES \wedge t.ssn = 123) \Rightarrow \\
&\quad \exists t1 \in TAKES(\\
&\qquad t1.c-id = t.c-id \wedge \\
&\qquad t1.ssn = o.ssn) \\
&\quad)\}
\end{aligned}
$$

Safety of expressions

- FORBIDDEN: $\cancel{\{t \mid t \notin STUDENT\}}$

It has infinite output!!

- Instead, always use

$$\{t \mid \ldots t \in SOME-TABLE\}$$

CMU SCS

Overview - conclusions

- rel. tuple calculus: DECLARATIVE
  - dfn
  - details
  - equivalence to rel. algebra
- **rel. domain calculus + QBE**

CMU SCS

General Overview

- relational model
- Formal query languages
  - relational algebra
  - rel. tuple calculus
  - **rel. domain calculus**

CMU SCS

Overview - detailed

- rel. tuple calculus
  - dfn
  - details
  - equivalence to rel. algebra
- **rel. domain calculus + QBE**

CMU SCS

Rel. domain calculus (RDC)

- Q: why?
- A: slightly easier than RTC, although equivalent - basis for QBE.
- idea: domain variables (w/ F.O.L.) - eg:
- 'find STUDENT record with ssn=123'

CMU SCS

Rel. Dom. Calculus

- find STUDENT record with ssn=123'

$$\{<s,n,a> | <s,n,a> \in STUDENT \wedge s = 123\}$$

CMU SCS

Details

- Like R.T.C - symbols allowed:

  $\wedge, \quad \vee, \quad \neg, \quad \Rightarrow$

  $>, \quad <, \quad =, \quad \neq, \quad \leq, \quad \geq,$

  $(, \quad), \quad \in$

- quantifiers $\forall, \quad \exists$

Details

- but: domain (= column) variables, as opposed to tuple variables, eg:

$$< s, n, a > \in STUDENT$$

ssn **name** **address**

Reminder: our Mini-U db

STUDENT

Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

CLASS

c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

TAKES

SSN	c-id	grade
123	15-413	A
234	15-413	B

Examples

- find all student records

$$\{< s, n, a > | < s, n, a > \in STUDENT\}$$

RTC: $\{t \mid t \in STUDENT\}$

Examples

- (selection) find student record with ssn=123

Examples

- (selection) find student record with ssn=123

$$\{< 123, n, a > | < 123, n, a > \in STUDENT\}$$

or

$$\{< s, n, a > | < s, n, a > \in STUDENT \wedge s = 123\}$$

RTC: $\{t \mid t \in STUDENT \wedge t.ssn = 123\}$

Examples

- (projection) find name of student with ssn=123

$$\{< n > | \quad < 123, n, a > \in STUDENT \}$$

CMU SCS

Examples

- (projection) find name of student with ssn=123

$$\{<n> \mid \exists a(<123,n,a> \in STUDENT)\}$$

need to 'restrict' "a"

RTC: $\{t \mid \exists s \in STUDENT(s.ssn = 123 \wedge t.name = s.name)\}$

CMU SCS

Examples cont'd

- (union) get records of both PT and FT students

RTC: $\{t \mid t \in FT\_STUDENT \vee t \in PT\_STUDENT\}$

CMU SCS

Examples cont'd

- (union) get records of both PT and FT students

$$\{<s,n,a> \mid <s,n,a> \in FT\_STUDENT \vee <s,n,a> \in PT\_STUDENT\}$$

CMU SCS

Examples

- difference: find students that are not staff

RTC: $\{t \mid t \in STUDENT \wedge t \notin STAFF\}$

CMU SCS

Examples

- difference: find students that are not staff

$$\{<s,n,a> \mid <s,n,a> \in STUDENT \wedge <s,n,a> \notin STAFF\}$$

CMU SCS

Cartesian product

- eg., dog-breeding: MALE x FEMALE
- gives all possible couples

MALE

name
spike
spot

x

FEMALE

name
lassie
shiba

=

M.name	F.name
spike	lassie
spike	shiba
spot	lassie
spot	shiba

CMU SCS

Cartesian product

- find all the pairs of (male, female) - RTC:

$$\{t \mid \exists m \in MALE \wedge \\ \exists f \in FEMALE \\ t.m-name = m.name \wedge \\ t.f-name = f.name\}$$

CMU SCS

Cartesian product

- find all the pairs of (male, female) - RDC:

$$\{<m, f> \mid <m> \in MALE \wedge \\ <f> \in FEMALE\}$$

CMU SCS

'Proof' of equivalence

- rel. algebra <-> rel. domain calculus <-> rel. tuple calculus

CMU SCS

Overview - detailed

- rel. domain calculus
  - why?
  - details
  - examples
  - equivalence with rel. algebra
  - **more examples**; 'safety' of expressions

CMU SCS

More examples

- join: find names of students taking 15-415

CMU SCS

Reminder: our Mini-U db

STUDENT

Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

CLASS

c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

TAKES

SSN	c-id	grade
123	15-413	A
234	15-413	B

CMU SCS

More examples

- join: find names of students taking 15-415 - in RTC

$$\{t \mid \exists s \in STUDENT$$
$$\wedge \exists e \in TAKES\ (\ s.ssn = e.ssn \wedge$$
$$t.name = s.name \wedge$$
$$e.c-id\ = 15-415)\}$$

CMU SCS

More examples

- join: find names of students taking 15-415 - in RDC

$$\{< n > \mid \exists s\ \exists a\ \exists g (< s, n, a > \in STUDENT$$
$$\wedge < s, 15-415, g > \in TAKES)\}$$

CMU SCS

Sneak preview of QBE:

$$\{< n > \mid \exists s\ \exists a\ \exists g (< s, n, a > \in STUDENT$$
$$\wedge < s, 15-415, g > \in TAKES)\}$$

STUDENT		
Ssn	Name	Address
_x	P.	

TAKES		
SSN	c-id	grade
_x	15-415	

CMU SCS

Sneak preview of QBE:

- very user friendly
- heavily based on RDC
- very similar to MS Access interface

STUDENT		
Ssn	Name	Address
_x	P.	

TAKES		
SSN	c-id	grade
_x	15-415	

CMU SCS

More examples

- 3-way join: find names of students taking a 2-unit course - in RTC:

$$\{t \mid \exists s \in STUDENT \wedge \exists e \in TAKES$$
$$\exists c \in CLASS(\ s.ssn = e.ssn \wedge$$
$$e.c-id\ = c.c-id \wedge$$
$$t.name = s.name \wedge$$
$$c.units\ = 2)\}$$

join

projection

selection

CMU SCS

Reminder: our Mini-U db

_x .P

STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

_y 2

CLASS		
c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

TAKES		
SSN	c-id	grade
123	15-413	A
234	15-413	B

_x _y

More examples

- 3-way join: find names of students taking a 2-unit course

$$\{<n>| \ldots\ldots\ldots..$$
$$<s,n,a> \in STUDENT \wedge$$
$$<s,c,g> \in TAKES \wedge$$
$$<c,cn,2> \in CLASS\}$$

More examples

- 3-way join: find names of students taking a 2-unit course

$$\{<n>| \exists s,a,c,g,cn($$
$$<s,n,a> \in STUDENT \wedge$$
$$<s,c,g> \in TAKES \wedge$$
$$<c,cn,2> \in CLASS$$
$$)\}$$

Even more examples:

- self -joins: find Tom's grandparent(s)

PC	
p-id	c-id
Mary	Tom
Peter	Mary
John	Tom

PC	
p-id	c-id
Mary	Tom
Peter	Mary
John	Tom

Even more examples:

- self -joins: find Tom's grandparent(s)

$$\{t \mid \exists p \in PC \wedge \exists q \in PC$$
$$(\ p.c-id = q.p-id \ \wedge$$
$$p.p-id = t.p-id \wedge$$
$$q.c-id = "Tom")\}$$

Even more examples:

- self -joins: find Tom's grandparent(s)

$$\{t \mid \exists p \in PC \wedge \exists q \in PC$$
$$(\ p.c-id = q.p-id \ \wedge$$
$$p.p-id = t.p-id \wedge$$
$$q.c-id = "Tom")\}$$

$$\{<g>| \exists p(<g,p> \in PC \wedge$$
$$<p,"Tom"> \in PC)\}$$

Even more examples:

- self -joins: find Tom's grandparent(s)

$$\{<g>| \exists p(<g,p> \in PC \wedge$$
$$<p,"Tom"> \in PC)\}$$

CMU SCS

Hard examples: DIVISION

- find suppliers that shipped all the ABOMB parts

SHIPMENT

s#	p#
s1	p1
s2	p1
s1	p2
s3	p1
s5	p3

÷

ABOMB

p#
p1
p2

=

BAD_S

s#
s1

CMU SCS

Hard examples: DIVISION

- find suppliers that shipped all the ABOMB parts

$$\{t \mid \forall p(p \in ABOMB \Rightarrow (\\ \exists s \in SHIPMENT(\\ t.s\# = s.s\# \wedge \\ s.p\# = p.p\#)))\}$$

CMU SCS

Hard examples: DIVISION

- find suppliers that shipped all the ABOMB parts

$$\{t \mid \forall p(p \in ABOMB \Rightarrow (\\ \exists s \in SHIPMENT(\\ t.s\# = s.s\# \wedge \\ s.p\# = p.p\#)))\}$$

$$\{<s> \mid \forall p(<p> \in ABOMB \Rightarrow \\ <s,p> \in SHIPMENT)\}$$

CMU SCS

More on division

- find students that take all the courses that ssn=123 does (and maybe even more)

$$\{o \mid \forall t((t \in TAKES \wedge t.ssn = 123) \Rightarrow \\ \exists t1 \in TAKES(\\ t1.c-id = t.c-id \wedge \\ t1.ssn = o.ssn) \\ )\}$$

CMU SCS

More on division

- find students that take all the courses that ssn=123 does (and maybe even more)

$$\{<s> \mid \forall c(\exists g(<123, c, g> \in TAKES) \Rightarrow \\ \exists g'(<s, c, g'>) \in TAKES))\}$$

CMU SCS

Safety of expressions

- similar to RTC
- FORBIDDEN:

$$\{<s,n,a> \mid <s,n,a> \notin STUDENT\}$$

Overview - detailed

- **rel. domain calculus + QBE**
  - dfn
  - details
  - equivalence to rel. algebra

Fun Drill:Your turn …

- Schema:

  Movie(<u>title</u>, year, studioName)

  ActsIn(<u>movieTitle</u>, <u>starName</u>)

  Star(<u>name</u>, gender, birthdate, salary)

Your turn …

- Queries to write in TRC:
  - Find all movies by Paramount studio
  - … movies starring Kevin Bacon
  - Find stars who have been in a film w/Kevin Bacon
  - Stars within six degrees of Kevin Bacon*
  - Stars connected to K. Bacon via <u>any number</u> of films**

* Try *two* degrees for starters   ** Good luck with this one!

Answers …

- Find all movies by Paramount studio

{M | M∈Movie ∧
  M.studioName = 'Paramount'}

Answers …

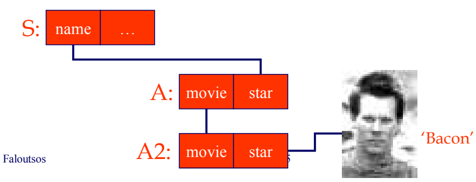
- Movies starring Kevin Bacon

{M | M∈Movie ∧
∃A∈ActsIn(A.movieTitle = M.title ∧
  A.starName = 'Bacon'))}

Answers …

- Stars who have been in a film w/Kevin Bacon

{S | S∈Star ∧
  ∃A∈ActsIn(A.starName = S.name ∧
    ∃A2∈ActsIn(A2.movieTitle = A.movieTitle ∧
      A2.starName = 'Bacon'))}

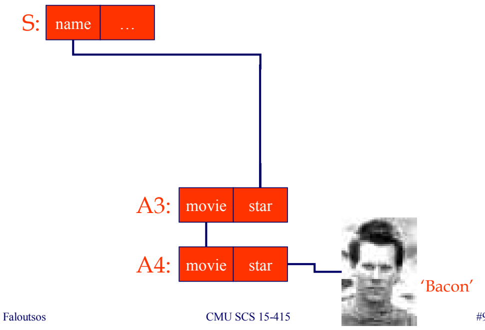


Answers …

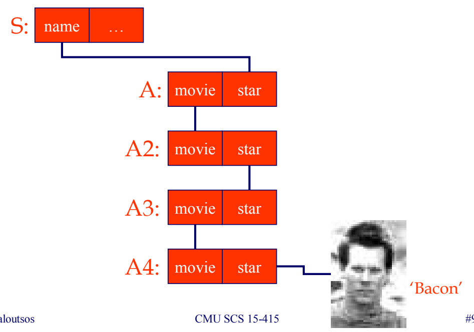
- Stars within ~~six~~ two degrees of Kevin Bacon

{S | S∈Star ∧
∃A∈ActsIn(A.starName = S.name ∧
∃A2∈ActsIn(A2.movieTitle = A.movieTitle ∧
∃A3∈ActsIn(A3.starName = A2.starName ∧
∃A4∈ActsIn(A4.movieTitle = A3.movieTitle ∧
A4.starName = 'Bacon'))}

Two degrees:



Two degrees:



Answers …

- Stars connected to K. Bacon via any number of films
- Sorry … that was a trick question
  - Not expressible in relational calculus!!
- What about in relational algebra?
  - We will be able to answer this question shortly …

Expressive Power

- Expressive Power (Theorem due to Codd):
  - Every query that can be expressed in relational algebra can be expressed as a safe query in DRC / TRC; the converse is also true.
- *Relational Completeness*:
  Query language (e.g., SQL) can express every query that is expressible in relational algebra/calculus.
  (actually, SQL is more powerful, as we will see…)

Question:

- Can we express previous query ('any # steps') in relational algebra?
- A: If we could, then by Codd's theorem we could also express it in relational calculus. However, we know the latter is not possible, so the answer is no.

CMU SCS

Summary

- The relational model has rigorously defined query languages — simple and powerful.
- Relational algebra is more operational/procedural
  - useful as internal representation for query evaluation plans
- Relational calculus is declarative
  - users define queries in terms of what they want, not in terms of how to compute it.

CMU SCS

Summary - cnt'd

- Several ways of expressing a given query
  - a *query optimizer* should choose the most efficient version.
- Algebra and safe calculus have same *expressive power*
  - leads to the notion of *relational completeness*.