CMU SCS

Carnegie Mellon Univ. Dept. of Computer Science 15-415/615 – DB Applications

C. Faloutsos & A. Pavlo

Lecture#5: *Relational calculus*

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
- rel. domain calculus + QBE

CMU SCS

Motivation

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

Solution: rel. calculus

- describes what we want
- two equivalent flavors: ‘tuple’ and ‘domain’ calculus
- basis for SQL and QBE, resp.
- Useful for proofs (see query optimization, later)

Rel. tuple calculus (RTC)

- first order logic

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

‘Give me tuples ‘t’, satisfying predicate P - eg:

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

Details

- symbols allowed:

$$\wedge, \quad \vee, \quad \neg, \quad \Rightarrow$$
$$>, \quad <, \quad =, \quad \neq, \quad \leq, \quad \geq,$$
$$(, \quad), \quad \in$$

- quantifiers $\forall, \quad \exists$

Specifically

- Atom

$$t \in TABLE$$
$$t.attr \leq\geq const$$
$$t.attr \leq\geq s.attr'$$

CMU SCS

Specifically

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

Faloutsos - Pavlo CMU SCS 15-415/615 #9

CMU SCS

Specifically

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

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

'every human is mortal : no human is immortal'

Faloutsos - Pavlo CMU SCS 15-415/615 #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 - Pavlo CMU SCS 15-415/615 #11

CMU SCS

Examples

- find all student records

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

output tuple

of type 'STUDENT'

Faloutsos - Pavlo CMU SCS 15-415/615 #12

CMU SCS

(Goal: evidence that RTC = RA)

- Full proof: complicated
- We'll just show examples of the 5 RA fundamental operators, and how RTC can handle them
- (Quiz: which are the 5 fundamental op's?)

CMU SCS

FUNDAMENTAL Relational operators

- selection $\sigma_{condition} \quad (R)$
- projection $\pi_{att-list}(R)$
- cartesian product MALE x FEMALE
- set union R U S
- set difference R - S

CMU SCS

Examples

- (selection) find student record with ssn=123

CMU SCS

Examples

- ✔ σ selection
- π projection
- X cartesian product
- U set union
- \- set difference

- (selection) find student record with ssn=123

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

CMU SCS

Examples

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #17

CMU SCS

- ✔ σ selection
- ✔ π projection
- X cartesian product
- U set union
- \- set difference

Examples

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

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

't' has only one column

Faloutsos - Pavlo CMU SCS 15-415/615 #18

CMU SCS

'Tracing'

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

t

Name
aaaa
....
jones
...
zzzz

s

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

Faloutsos - Pavlo CMU SCS 15-415/615 #19

CMU SCS

- ✔ σ selection
- ✔ π projection
- X cartesian product
- ✔ U set union
- \- set difference

Examples cont'd

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

Faloutsos - Pavlo CMU SCS 15-415/615 #20

CMU SCS

Examples cont'd

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

$$\{t \mid t \in FT\_STUDENT \lor t \in PT\_STUDENT\}$$

CMU SCS

Examples

- ✔ σ selection
- ✔ π projection
- X cartesian product
- ✔ U set union
- ✔ - set difference

- 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

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

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

- σ selection
- π projection
- X cartesian product
- U set union
- \- set difference

- find all the pairs of (male, female)

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

Faloutsos - Pavlo CMU SCS 15-415/615 #25

CMU SCS

'Proof' of equivalence

- rel. algebra <-> rel. tuple calculus

- σ selection
- π projection
- X cartesian product
- U set union
- \- set difference

Faloutsos - Pavlo CMU SCS 15-415/615 #26

CMU SCS

Overview - detailed

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

Faloutsos - Pavlo CMU SCS 15-415/615 #27

CMU SCS

More examples

- join: find names of students taking 15-415

Faloutsos - Pavlo CMU SCS 15-415/615 #28

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

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

CMU SCS

More examples

- join: find names of students taking 15-415

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

(Remember: 'SPJ')

CMU SCS

More examples

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

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??

More examples

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

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

CMU SCS

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

CMU SCS

Even more examples:

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

$$
\begin{aligned}
\{t \mid & \exists p \in PC \wedge \exists q \in PC \\
& (\, p.c-id = q.p-id \wedge \\
& \quad p.p-id = t.p-id \wedge \\
& \quad q.c-id = "Tom")\}
\end{aligned}
$$

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

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

CMU SCS

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

CMU SCS

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

¬ a v b

CMU SCS

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

$\forall x(P(x)) \equiv \neg \exists x(\neg P(x))$

CMU SCS

a ⇒ b is the same as ¬a v 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.

CMU SCS

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'

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

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

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 $\forall, \quad \exists$

CMU SCS

Details

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

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

ssn name address

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

Examples

- find all student records

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

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

CMU SCS

Examples

- (selection) find student record with ssn=123

CMU SCS

('Proof' of RDC = RA)

- Again, we show examples of the 5 fundamental operators, in RDC

CMU SCS

Examples

- (selection) find student record with ssn=123

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

CMU SCS

Examples

- (selection) find student record with ssn=123

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

or

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

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

CMU SCS

Examples

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

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

CMU SCS

Examples

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

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

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

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

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #65

CMU SCS

Cartesian product

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

MALE
name
spike
spot

x $\bowtie$

FEMALE
name
lassie
shiba

=

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

Faloutsos - Pavlo CMU SCS 15-415/615 #66

CMU SCS

Cartesian product

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #67

CMU SCS

Cartesian product

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

Faloutsos - Pavlo CMU SCS 15-415/615 #68

Cartesian product

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

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

'Proof' of equivalence

- rel. algebra <-> rel. domain calculus

<-> rel. tuple calculus

Overview - detailed

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

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

Glimpse of QBE:

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

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

CMU SCS

More examples

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

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

CMU SCS

More examples

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #81

CMU SCS

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

Faloutsos - Pavlo CMU SCS 15-415/615 #82

CMU SCS

Even more examples:

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #83

CMU SCS

Even more examples:

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

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #84

CMU SCS

Even more examples:

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #85

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

Faloutsos - Pavlo CMU SCS 15-415/615 #86

CMU SCS

Hard examples: DIVISION

- find suppliers that shipped all the ABOMB parts

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

Faloutsos - Pavlo CMU SCS 15-415/615 #87

CMU SCS

Hard examples: DIVISION

- find suppliers that shipped all the ABOMB parts

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #88

CMU SCS

More on division

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #89

CMU SCS

More on division

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

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

Faloutsos - Pavlo CMU SCS 15-415/615 #90

CMU SCS

Safety of expressions

- similar to RTC
- FORBIDDEN:

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

Faloutsos - Pavlo CMU SCS 15-415/615 #91

CMU SCS

Overview - detailed

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

Faloutsos - Pavlo CMU SCS 15-415/615 #92

CMU SCS

Fun Drill:Your turn …

- Schema:

  Movie(title, year, studioName)

  ActsIn(movieTitle, starName)

  Star(name, gender, birthdate, salary)

CMU SCS

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 any number of films**

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

CMU SCS

Answers …

- Find all movies by Paramount studio

$$\{M \mid M \in Movie \land M.studioName = \text{'Paramount'}\}$$

CMU SCS

Answers …

- Movies starring Kevin Bacon

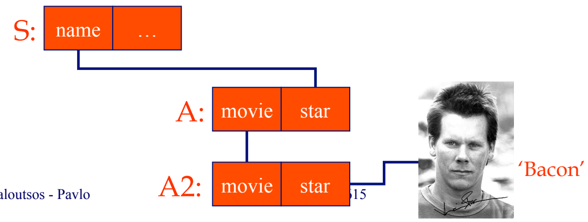
$$\{M \mid M \in Movie \land \exists A \in ActsIn(A.movieTitle = M.title \land A.starName = \text{'Bacon'})\}$$

CMU SCS

Answers …

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

$$\{S \mid S \in Star \wedge \\ \exists A \in ActsIn(A.starName = S.name \wedge \\ \exists A2 \in ActsIn(A2.movieTitle = A.movieTitle \wedge \\ A2.starName = \text{'Bacon'}))\}$$



CMU SCS

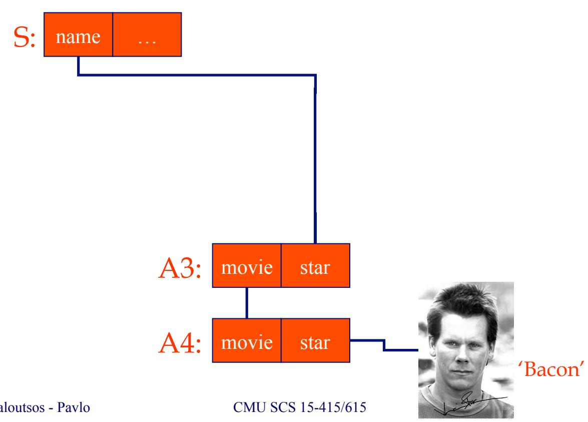
Answers …

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

$$\{S \mid S \in Star \wedge \\ \exists A \in ActsIn(A.starName = S.name \wedge \\ \exists A2 \in ActsIn(A2.movieTitle = A.movieTitle \wedge \\ \exists A3 \in ActsIn(A3.starName = A2.starName \wedge \\ \exists A4 \in ActsIn(A4.movieTitle = A3.movieTitle \wedge \\ A4.starName = \text{'Bacon'}))\}$$

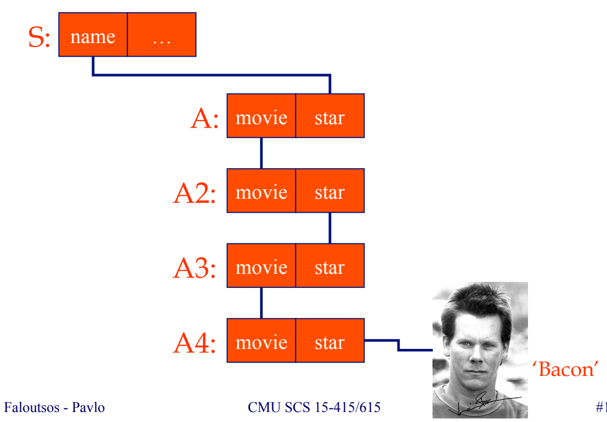
CMU SCS

Two degrees:



CMU SCS

Two degrees:



CMU SCS

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?
  - No – RA, RTC, RDC are equivalent

Faloutsos - Pavlo CMU SCS 15-415/615 #101

CMU SCS

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 RDC / RTC; 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…)

Faloutsos - Pavlo CMU SCS 15-415/615 #102

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.

Faloutsos - Pavlo CMU SCS 15-415/615 #103

CMU SCS

Summary - cnt'd

- Several ways of expressing a given query
  - a *query optimizer* chooses best plan.
- Algebra and safe calculus: same *expressive power*

  => *relational completeness*.

Faloutsos - Pavlo CMU SCS 15-415/615 #104