CPSC 304
Introduction to Database Systems

Datalog & Deductive Databases

Textbook Reference
Database Management Systems: Sections 24.1 – 24.4

Hassan Khosravi
Borrowing many slides from Rachel Pottinger

Databases: The Continuing Saga

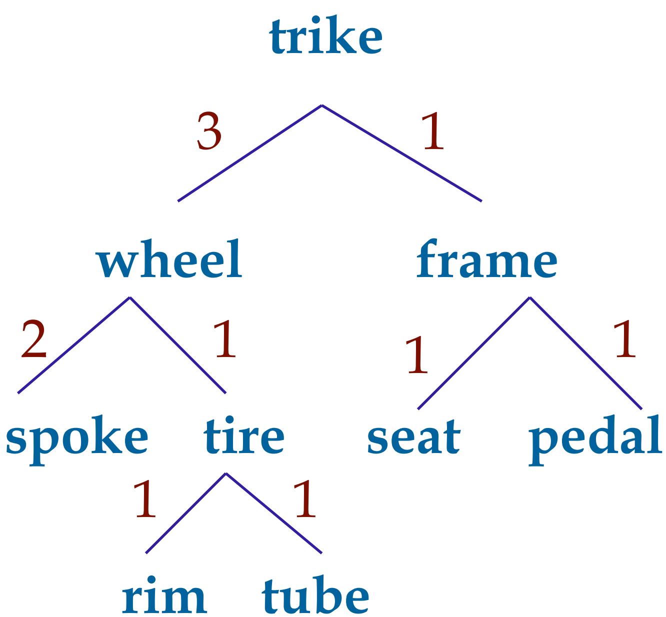
When last we left databases…

- We had decided they were great things
- We knew how to conceptually model them in ER diagrams
- We knew how to logically model them in the relational model
- We knew how to normalize them
- We could query them using SQL – the relational approach
- Are there any other good ways to query?

Learning Goals

- Given a set of tuples (an input relation) and rules, compute the output relation for a Datalog program.
- Write Datalog programs to query an input relation.
- Explain why we want to extend RA or SQL with recursive queries. Provide good examples of such queries.
- Explain the importance of safe queries, and what makes a Datalog query safe.

Motivation (Surprisingly difficult)



part	subpart	number
trike	wheel	3
trike	frame	1
frame	seat	1
frame	pedal	1
wheel	spoke	2
wheel	tire	1
tire	rim	1
tire	tube	1

Write an SQL query to find all of the components required for a trike

Datalog

- Our final relational query language
- Based on logic notation (Prolog)
- Can express queries that are not expressible in relational algebra or standard SQL (recursion).
- Uses sets (like RA, unlike SQL)
- No grouping and aggregation, order by.
- Cleaner → convenient for analysis

A nice and easy example to start

```
Likes(dee,jan).
Likes(jan, jamie).
Likes(dee,wally).
Likes(wally,jean).
Loves(A,C) :- Likes(A,B), Likes(B,C).
```

Loves(dee, jamie)
Loves(dee, jean)

- Based on some facts on some rules, you can deduct new facts.

Predicates and Atoms

- Relations are represented by predicates
- Tuples are represented by atoms.

  Likes(dee,jan)

Arithmetic comparison atoms:

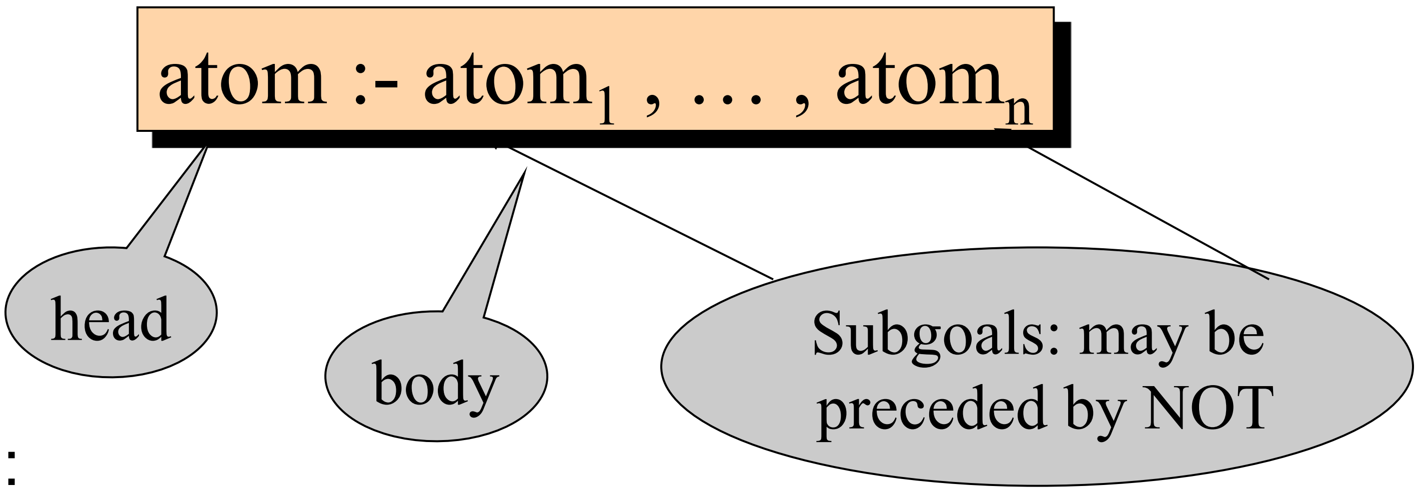
  X < 100,   X+Y+5 > Z/2, X <> 42

- Negated atoms:

  NOT Likes(dee,jean)

Datalog Definitions

- A Datalog rule:



- E.g.:
- Loves(A,C) :- Likes(A,B), Likes(B,C).
- A comma between the atoms means “and” (sometimes you’ll see this as “&”)
- Read the rule as “if we know body, then we know head”
- Datalog program = a collection of rules

A single rule can express exactly select-from-where queries.

The Meaning of Datalog Rules

```
Likes(dee,jan).
Likes(jan, jamie).
Likes(dee,wally).
Likes(wally,jean).
Loves(A,C) :- Likes(A,B), Likes(B,C).
```

Loves(dee, jamie)
Loves(dee, jean)

Consider every assignment from the variables in the body to the constants in the database. (same variable name means require the same value)

If each atom in the body is in the database, then the tuple for the head is in the result.

Running example

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

Projection

Product (<u>pid</u>, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (<u>cid</u>, name, stock price, country)
Person(<u>sin</u>, name, phone number, city)

Projection is performed by the variables in the head of the query:

- Find the name of all products:

RA: π_{name}(Product)

Datalog: Ans(N):-Product(P,N,PR,C,M)

Projection practice

Product (<u>pid</u>, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (<u>cid</u>, name, stock price, country)
Person(<u>sin</u>, name, phone number, city)

Find the countries of all the companies

Ans1(Co):- Company (C, N, S, Co)

– make sure C <> Co

Selection

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

Selection is performed by either using the same variable, a constant, or adding an arithmetic comparison:

- Find all purchases with the same buyer and seller:
  RA: $\sigma_{\text{buyer-sin} = \text{seller-sin}}(\text{Purchase})$
  Datalog: Ans1(B,B,S,P):-Purchase(B,B,S,P)

- Find all Canadian companies:
  RA: $\sigma_{\text{country='Canada'}}(\text{Company})$
  Datalog: Ans2(C,N,S, 'Canada'):-Company(C,N,S, 'Canada')

Selection practice

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

Find all products over $99.99:

RA: $\sigma_{price>99.99}$(Product)

Datalog: Ans(I,N,P,C,M) :- Product(I,N,P,C,M), P>99.99

Find all English companies with stock prices less than $100

Ans1(C,N,S, 'England'):-
Company(C, N, S, 'England'), S < 100

Selection & Projection

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

- Find the names of all products over $99.99:

  RA: $\pi_{name}(\sigma_{price>99.99}(\text{Product}))$

  Datalog: Ans(N) :- Product(I,N,P,C,M), P>99.99

Clicker Question

Given the following schema:

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

And the Datalog definition:

Ans(C,N) :- Product(I,N,P,C,M), P>99.99

What is the proper translation to RA?

A. $\pi_{name,category}(\sigma_{price>99.99}(Product))$
B. $\pi_{name(}\pi_{category}(\sigma_{price>99.99}(Product)))$
C. $\pi_{category(}\pi_{name}(\sigma_{price>99.99}(Product)))$
D. $\pi_{category,name}(\sigma_{price>99.99}(Product))$
E. None of the above

Clicker Question

Given the following schema:

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

And the Datalog definition:

Ans(C,N) :- Product(I,N,P,C,M), P>99.99

What is the proper translation to RA?

A. $\pi_{name,category}(\sigma_{price>99.99}(Product))$
B. $\pi_{name(}\pi_{category}(\sigma_{price>99.99}(Product)))$
C. $\pi_{category(}\pi_{name}(\sigma_{price>99.99}(Product)))$
D. $\pi_{category,name}(\sigma_{price>99.99}(Product))$
E. None of the above

D is correct

A – name before category
B,C – can't project name from category & vice versa

Selection & Projection and Joins

Product (<u>pid</u>, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (<u>cid</u>, name, stock price, country)
Person(<u>sin</u>, name, phone number, city)

Joins are performed by using the same variable in different relations

- Find store names where Fred bought something:
  RA: $\pi_{store}\sigma_{name=\text{“Fred”}}(\text{Person}) \bowtie_{sin=buyer\text{-}sin} \text{Purchase}$

- Datalog: S(N) :- Person(S, “Fred”,T,C),
  Purchase(S,L,N,P)

Anonymous Variables

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

Find names of people who bought from "Gizmo Store"

- E.g.:

Ans4(N) :- Person(S, N, _, _), Purchase (S, _,"Gizmo Store", _)

Each _ means a fresh, new variable
Very useful: makes Datalog even easier to read

Exercise part 1

Product (<u>pid</u>, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (<u>cid</u>, name, stock price, country)
Person(<u>sin</u>, name, phone number, city)

Ex #1: Find SINs of people who bought products in the "computers" category.

Ans1(B):-Purchase(B,_,_,P), Product(P,_,_,'Computers',_)

Ex #2: Find the sin of people who bought Canadian products

Ans2(B):- Purchase(B,_,_,P), Product(P,_,_,_,C), Company(C, _, _, 'Canada')

Clicker exercise – basic Datalog

- Consider Unknown(A,B):
- Compute:
  Secret (A ,B):- Unknown(A,C), Unknown(C,B)
- Which of the following tuples are in Secret(A ,B)?

A. (a1,a1)
B. (a2,a3)
C. (a4,a5)
D. Both A & B
E. None of the above

o	d
a1	a2
a1	a3
a1	a4
a2	a3
a3	a4
a4	a5
a2	a1

Clicker exercise – basic Datalog

- Consider Unknown(A,B):
- Compute:
  Secret (A ,B):- Unknown(A,C), Unknown(C,B)
- Which of the following tuples are in Secret(A ,B)?

A. (a1,a1)
B. (a2,a3)
C. (a4,a5)
D. Both A & B
E. None of the above

One hop

Correct

o	d
a1	a2
a1	a3
a1	a4
a2	a3
a3	a4
a4	a5
a2	a1

Clicker exercise – A more meaningful version

- Consider Flight(orig,dest):
- Compute:
  Twohops (orig,final_dest):-
  Flight(orig,mid), Flight(mid,final_dest) (paths of length 2 again)
- Which of the following tuples are in Twohops(orig,final_dest)?

orig	dest
YVR	SEA
YVR	PIT
YVR	RDU
SEA	PIT
PIT	RDU
RDU	ITH
SEA	YVR

A. (YVR,YVR)

B. (SEA,PIT)

C. (RDU,ITH) One hop

D. Both A & B Correct

E. None of the above

Exercise part 2

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

Ex #3: Find names of people who bought Canadian products that cost under 50

Ans3(N):-Product(P, _, Pr, _, C), Company(C,_,_, 'Canada'),
Purchase(B, _, _, P), Person(B, N, _, _), Pr < 50

Clicker Question

Given the following schema:

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

And the following query:

“Find the phone numbers of all customers who bought a computer product from a Canadian company that cost $100”

What is the proper translation into Datalog?

A. Ans(PN):- Purchase(B,_,_,P), Product(P,_,_,‘computer’,C),
    Company(C, _, _, ‘Canada’), Person(B,_,PN,_,_),price=100.

B. Ans(PN):- Purchase(B,_,_,P), Product(P,_,100, ‘computer’,C),
    Company(C, _, _, ‘Canada’), Person(B,_,PN,_,_).

C. Ans(PN):- Purchase(B,_,_,P), Product(P,_,100, ‘computer’,C),
    Company(C, _, _, country), Person(B,_,_,_,_),country=‘Canada’.

D. All are correct

E. None of the above

Clicker Question

Given the following schema:

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

A – price not in any atoms
C – PN not in any atoms

And the following query:

"Find the phone numbers of all customers who bought a computer product from a Canadian company that cost $100"

What is the proper translation into Datalog?

B is correct

A. Ans(PN):- Purchase(B,_,_,P), Product(P,_,_,'computer',C), Company(C, _, _, 'Canada'), Person(B,_,PN,_,_),price=100.

B. Ans(PN):- Purchase(B,_,_,P), Product(P,_,100, 'computer',C), Company(C, _, _, 'Canada'), Person(B,_,PN,_,_).

C. Ans(PN):- Purchase(B,_,_,P), Product(P,_,100, 'computer',C), Company(C, _, _, country), Person(B,_,_,_,_),country='Canada'.

D. All are correct

E. None of the above

Clicker Question

Consider Unknown(A,B):

Compute:

Secret (A ,B):- Unknown(B,A), Unknown(C,A), C != B.

Which of the following tuples are in Secret(A ,B)?

A. (a2,a3)
B. (a1,a2)
C. (a2,a1)
D. All of the above
E. None of the above

o	d
a1	a2
a1	a3
a1	a4
a2	a3
a3	a4
a4	a2
a2	a1

Answer A explained

Consider Unknown(A,B):

Compute:

Secret (A ,B):- Unknown(B,A), Unknown(C,A), C != B.

Which of the following tuples are in Secret(A ,B)?

A. (a2,a3)
B. (a1,a2)
C. (a2,a1)
D. All of the above
E. None of the above

o	d
a1	a2
a1	a3
a1	a4
a2	a3
a3	a4
a4	a2
a2	a1

A=a2, B=a3

Unknown(a3,a2) is not in the table.

Answer B explained

Consider Unknown(A,B):

Compute:

Secret (A ,B):- Unknown(B,A), Unknown(C,A), C != B.

Which of the following tuples are in Secret(A ,B)?

A. ~~(a2,a3)~~
B. (a1,a2)
C. (a2,a1)
D. All of the above
E. None of the above

o	d
a1	a2
a1	a3
a1	a4
a2	a3
a3	a4
a4	a2
a2	a1

A=a1, B=a2

Unknown(a2,a1) ok

Unknown(C,a1), C !=B does not exist

Answer C explained

Consider Unknown(A,B):

Compute:

Secret (A ,B):- Unknown(B,A), Unknown(C,A), C != B.

Which of the following tuples are in Secret(A ,B)?

A. ~~(a2,a3)~~
B. ~~(a1,a2)~~
C. (a2,a1)
D. All of the above
E. None of the above

o	d
a1	a2
a1	a3
a1	a4
a2	a3
a3	a4
a4	a2
a2	a1

A=a2, B=a1

Unknown(a1,a2) ok

Unknown(a4,a2) where C=a4 ok

C is correct

Multiple Datalog Rules

Product (<u>pid</u>, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (<u>cid</u>, name, stock price, country)
Person(<u>sin</u>, name, phone number, city)

- Find names of people that are either buyers or sellers:

  A(N) :- Person(S,N,A,B), Purchase(S,C,D,E)
  A(N) :- Person(S,N,A,B), Purchase(C,S,D,E)

- Multiple rules correspond to union

Exercise part 3

Product (<u>pid</u>, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (<u>cid</u>, name, stock price, country)
Person(<u>sin</u>, name, phone number, city)

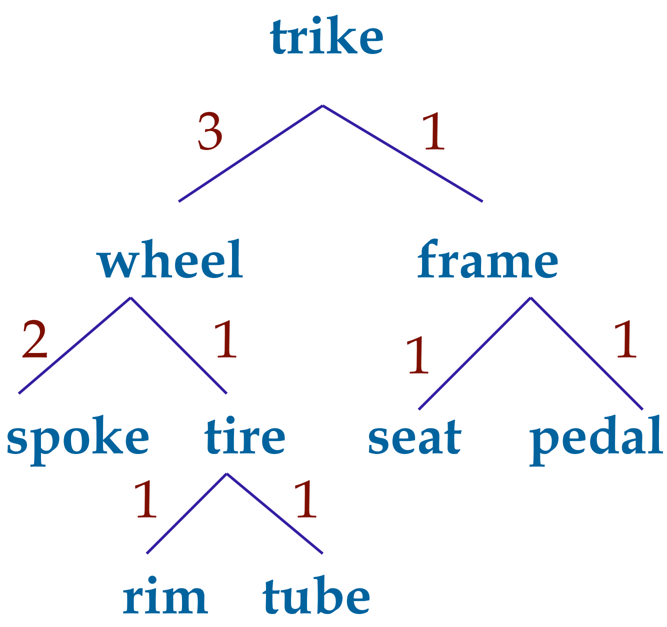
Ex #4: Find sins of people who bought stuff from a person named Joe or bought products from a company whose stock prices is more than $50.

Ans(Sin):-Purchase(Sin,Ssin,_,_), Person(Ssin,"Joe",_,_)
Ans(Sin):-Purchase(Sin,_,_,Pid), Product(Pid,_,_,_,Cid),
Company(Cid,_,Sp,_), Sp > 50

find all of the components required for trike

Components (Part , Subpart) :- Assembly (Part , Subpart, Qty)

Components (Part , Subpart) :- Assembly (Part , Part2, Qty) ,
Components(Part2, Subpart)



Component(trike, wheel)
Component(trike, frame)

Component(trike, spoke)
Component(trike, tire)

Component(trike, seat)
Component(trike, pedal)

....

When should we stop??

The Fixpoint Operator

- A fixpoint of a function f is a value v
  - f(v) = v
- Example
  - double → doubles every element of a list
    - double{1,2,5} = {2,4,10)
  - double+ = identity U double
    - double+{1,2,5} = {1,2,4,5,10)
  - double+{even integers} = {even integers} (fixpoint)

Fixpoint over set of tuples

f=

Components (Part , Subpart) :- Assembly (Part , Subpart, Qty)

Components (Part , Subpart) :- Assembly (Part , Part2, Qty) ,
Components(Part2, Subpart)

v=

Component(trike, wheel)
Component(trike, frame)
Component(trike, spoke)
Component(trike, tire)
Component(trike, seat)
Component(trike, pedal)
Component(trike, rim)
Component(trike, tube)

f(v) = v

v is a Fixpoint for f

I cannot use any of my rules to deduct any new facts

Rule safety

- Every variable in the head of a rule must also appear in the body.

PriceYarts (Part, Price) :- Assembly(Part, Subpart, Qty) , Qty> 2.

Can generate infinite new facts

- Every variable must appear in a relation

Ans(Id) :- Product(Id,Name,Price,Category,Cid), Id < Stock_price

What is the value of stock_price?

- If a relation appears in the body of a rule preceded by NOT, we call this a negated occurrence. A program is range-restricted if every variable in the head of the rule appears in some positive relation occurrence in the body

Ans(Sin):- NOT Person(Sin, ‘Joe’, Ph, City)

Sin is unsafe

Exercise part 4

Product (pid, name, price, category, maker-cid)
Purchase (buyer-sin, seller-sin, store, pid)
Company (cid, name, stock price, country)
Person(sin, name, phone number, city)

Ex #5: Find the sins of people who are not named ‘Joe’

Ans(Sin):- Person(Sin,Name,Ph,City), NOT Person(Sin, ‘Joe’, Ph, City)

Recursive Queries with Negation

Is trike considered a big part or a small part?

Big(Part):- Assembly(Part , Subpart, Qty) , Qty> 2, NOT Small(Part) .
Small (Part) :- Assembly (Part , Subpart, Qty), NOT Big(Part).

Two fixpoints

Big: trike

Small: frame, wheel, tire

Big:

Small: trike, frame, wheel, tire

trike	wheel	3
trike	frame	1
frame	seat	1
frame	pedal	1
wheel	spoke	2
wheel	tire	1
tire	rim	1
tire	tube	1

We would need to use Stratification to overcome this problem

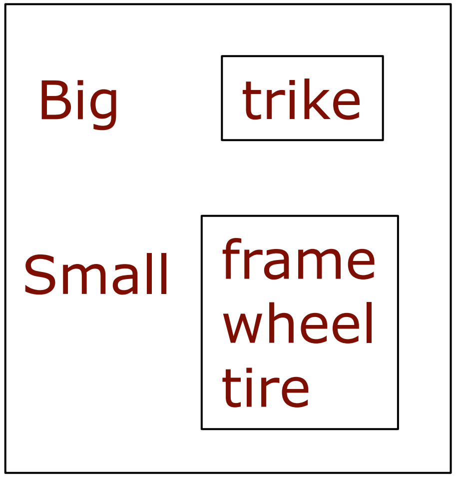
Stratification

- T <u>depends on</u> S if some rule with T in the head contains S or (recursively) some predicate that depends on S, in the body.
- <u>Stratified program:</u> If T depends on **not** S, then S cannot depend on T (or **not** T).
- If a program is stratified, the tables in the program can be partitioned into strata:
  - If predicate P is positively derived from a predicate Q then the strata number of P must be greater or equal to that of Q **[S(P) ≥ S(Q)]**
  - If predicate P is negatively derived from a predicate Q then the strata number of P must be greater than that of Q **[S(P) > S(Q)]**

Stratification

Big2(Part):- Assembly(Part , Subpart, Qty) , Qty> 2.
Small2 (Part) :- Assembly (Part , Subpart, Qty), NOT Big2(Part).

- First, compute the fixpoint of all tables in Stratum 1. (Stratum 0 tables are fixed.)
- Then, compute the fixpoint of tables in Stratum 2; then and so on, stratum-by-stratum.



Defining Views

VancouverView(Buyer,Seller,Product,Store) :-

Person(Buyer, “Vancouver”, Phone),

Purchase(Buyer, Seller, Product, Store),

not Purchase(Buyer, Seller, Product, “The Bay”)

Ans6(Buyer) :- VancouverView(Buyer, “Joe”, Prod, Store)

Ans6(Buyer) :- VancouverView(Buyer, Seller, Prod, Store),

Product(Prod, Price, Cat, Maker)

Company(Maker, Sp, Country), Sp > 50.

What is returned by Ans6?

Buyers from Vancouver that have never purchased anything from “The Bay” that have either bought from Joe or products that are from companies with SP> 50

Clicker exercise – Datalog with negation

- Consider the Unknown(A,B) relation, which is given on the right hand side

o	d
a1	a2
a1	a3
a1	a4
a2	a3
a3	a4
a4	a5

Secret (A, B):- Unknown (A,C), Unknown (C,B)
Nameless (A,B):-Secret (A,B), NOT unknown(A,B)

- Which of the following tuples are in Nameless(A,B)?

A. (a1,a4)
B. (a1,a5)
C. (a4,a5)
D. All of the above
E. None of the above

Clicker exercise – Datalog with negation

- Consider the Unknown(A,B) relation, which is given on the right hand side

o	d
a1	a2
a1	a3
a1	a4
a2	a3
a3	a4
a4	a5

Secret (A, B):- Unknown (A,C), Unknown (C,B)
Nameless (A,B):-Secret (A,B),   NOT unknown(A,B)

- Which of the following tuples are in Nameless(A,B)?

A. (a1,a4)   In Secret(a1,a3)(a3,a4) and Unknown(a1,a4)
B. (a1,a5)   In Nameless, so correct
C. (a4,a5)   In Unknown
D. All of the above
E. None of the above

Clicker exercise – A more meaningful version

- Consider Flight(orig,dest):
- Compute Indirect_only(orig,dest) defined by:
  Twohops (Orig,Final_dest):- Flight(Orig,Mid),
  Flight(Mid,Final_dest)
  Indirect_only(orig,dest):-Twohops(orig,dest),
  NOT Flight(orig,dest)
- Which of the following tuples are in Indirect_only(orig,dest)?

orig	dest
YVR	SEA
YVR	PIT
YVR	RDU
SEA	PIT
PIT	RDU
RDU	ITH

A. (YVR,RDU) In Twohops(YVR,PIT)(PIT,RDU) and Flight(YVR,RDU)

B. (YVR,ITH) In Indirect_only, so correct

C. (RDU,ITH) In Flight

D. All of the above

E. None of the above

Taking it to the next level

Say you're planning a beach vacation

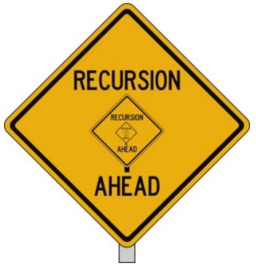
And you wanted to find if it's possible to get from YVR to OGG (that's on Maui)

Your available information:

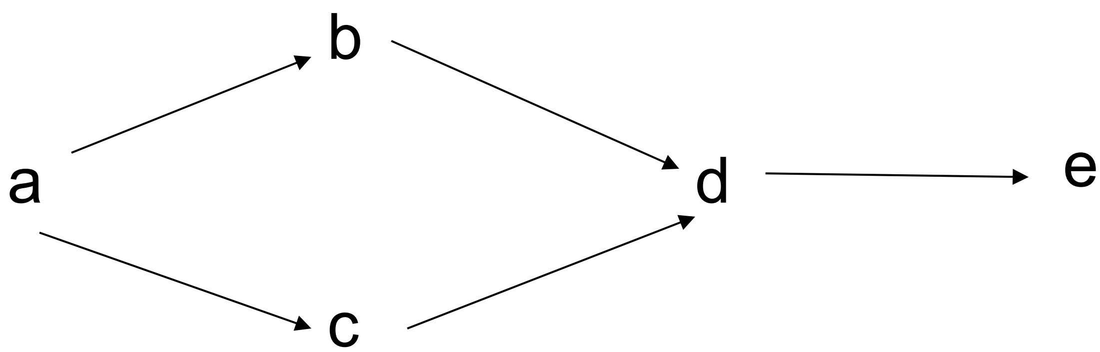
Flight(airline,num,origin,destination)

Now what?

A more general Example: Transitive Closure



Suppose we represent a graph w/ relation *Edge(X,Y):*
Edge(a,b), Edge (a,c), Edge(b,d), Edge(c,d), Edge(d,e)



How can I express the query: *Find all paths*

Path(X, Y)   :-   Edge(X, Y).
Path(X, Y)   :-   Path(X, Z), Path(Z, Y).

Evaluating Recursive Queries

Path(X, Y)   :-   Edge(X, Y).
Path(X, Y)   :-   Path(X, Z), Path(Z, Y).

Semantics: evaluate the rules until a *fixed point:*

Iteration #0: Edge: {(a,b), (a,c), (b,d), (c,d), (d,e)}
Path: {}

Iteration #1: Path: {(a,b), (a,c), (b,d), (c,d), (d,e)}

Iteration #2: Path gets the new tuples: (a,d), (b,e), (c,e)
Path: {(a,b), (a,c), (b,d), (c,d), (d,e), (a, d), (b,e), (c, e)}

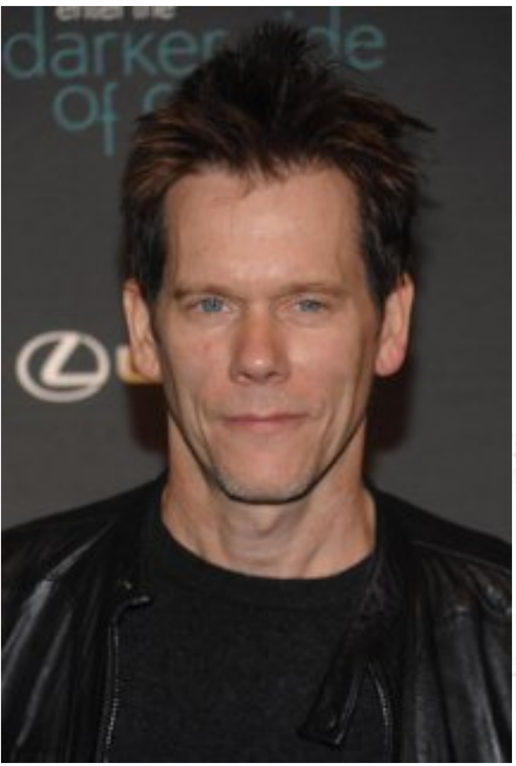
Iteration #3: Path gets the new tuple: (a,e)
Path: {(a,b), (a,c), (b,d), (c,d), (d,e), (a, d), (b,e), (c, e), (a,e)}

Iteration #4: Nothing changes → Stop.

Note: # of iterations depends on the data. Cannot be anticipated by only looking at the query!

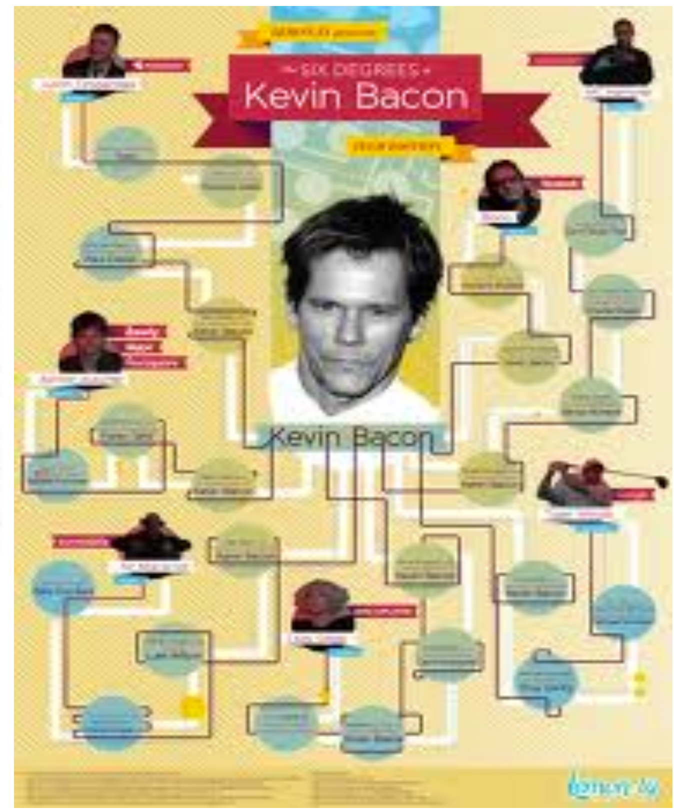
A fun Example

Kevin Bacon



6 degrees of separation

6 degrees of Kevin Bacon



More examples

- Given:
  Movie(id, title)
  Actor(id, name)
  Role(movie-id, actor-id, character)
- Find names of actors who have “Bacon numbers” (assume there’s only one “Kevin Bacon”)

CoStars(Aid,Bid):-Role(Mid,Aid,_), Role(Mid,Bid,_)
CoStars(Aid,Bid):- CoStars(Aid,Cid), CoStars(Cid,Bid)
Bacon_N(B):-Actor(Aid, “Kevin Bacon”), CoStars(Aid,Bid), Actor(Bid,B)

Not tested on exam

Recursive SQL? Sometimes…

Given: Assembly(Part, Subpart, Quantity)
Find: all of the components required for a trike

Datalog:
Comp(Part, Subpt) :- Assembly(Part, Subpt, Qty).
Comp(Part, Subpt) :- Assembly(Part, Part2, Qty), Comp(Part2, Subpt).

SQL:

```
WITH RECURSIVE Comp(Part, Subpt) AS
   (SELECT A1.Part, A1.Subpt FROM Assembly A1)
  UNION
   (SELECT A2.Part, C1.Subpt
   FROM Assembly A2, Comp C1
   WHERE A2.Subpt=C1.Part)

SELECT Subpart FROM Comp C2
WHERE Part = 'trike'
```

Skip the stuff on Magic Sets

- That's Datalog
- It's simple
- It's based on logic
- It's easy to see the join patterns (especially with anonymous variables)

Learning Goals Revisited

- Given a set of tuples (an input relation) and rules, compute the output relation for a Datalog program.
- Write Datalog programs to query an input relation.
- Explain why we want to extend RA or SQL with recursive queries. Provide good examples of such queries.
- Explain the importance of safe queries, and what makes a Datalog query safe.