

Adatbázisok 1

Rekurzió a Datalogban és SQL-99

Expressive Power of Datalog

- ▶ **Without recursion**, Datalog can express all and only the queries of core relational algebra.
 - ▶ The same as SQL select-from-where, without aggregation and grouping.
- ▶ **But with recursion**, Datalog can express more than these languages.

Intuitive introduction

- ▶ **Example 1: Ancestors**

ParentOf(parent,child)

- ▶ Find all of Mary's ancestors

- ▶ **Example 2: Company hierarchy**

Employee(ID,salary)

Manager(mID,eID)

Project(name,mgrID)

- ▶ Find total salary cost of project 'X'

- ▶ **Example 3: Airline flights**

Flight(orig,dest,airline,cost)

- ▶ Find cheapest way to fly from 'A' to 'B'

Another Recursive Example

- ▶ EDB: $\text{Par}(c,p) = p$ is a parent of c .
- ▶ Generalized cousins: people with common ancestors one or more generations back:

$\text{Sib}(x,y) \leftarrow \text{Par}(x,p) \text{ AND } \text{Par}(y,p) \text{ AND } x \neq y$

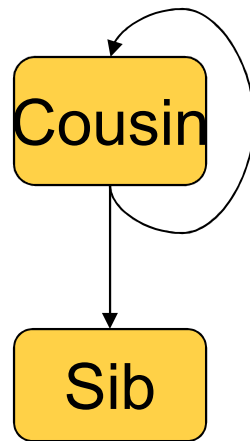
$\text{Cousin}(x,y) \leftarrow \text{Sib}(x,y)$

$\text{Cousin}(x,y) \leftarrow \text{Par}(x,xp) \text{ AND } \text{Par}(y,yp)$
 $\text{AND } \text{Cousin}(xp,yp)$

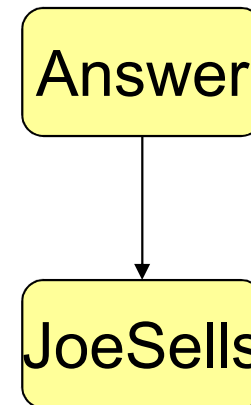
Definition of Recursion

- ▶ Form a **dependency graph** whose
- ▶ Nodes = IDB predicates.
- ▶ Arc $X \rightarrow Y$ if and only if there is a rule with X in the head and Y in the body.
- ▶ Cycle = **recursion**; no cycle = no recursion.

Example: Dependency Graphs



Recursive

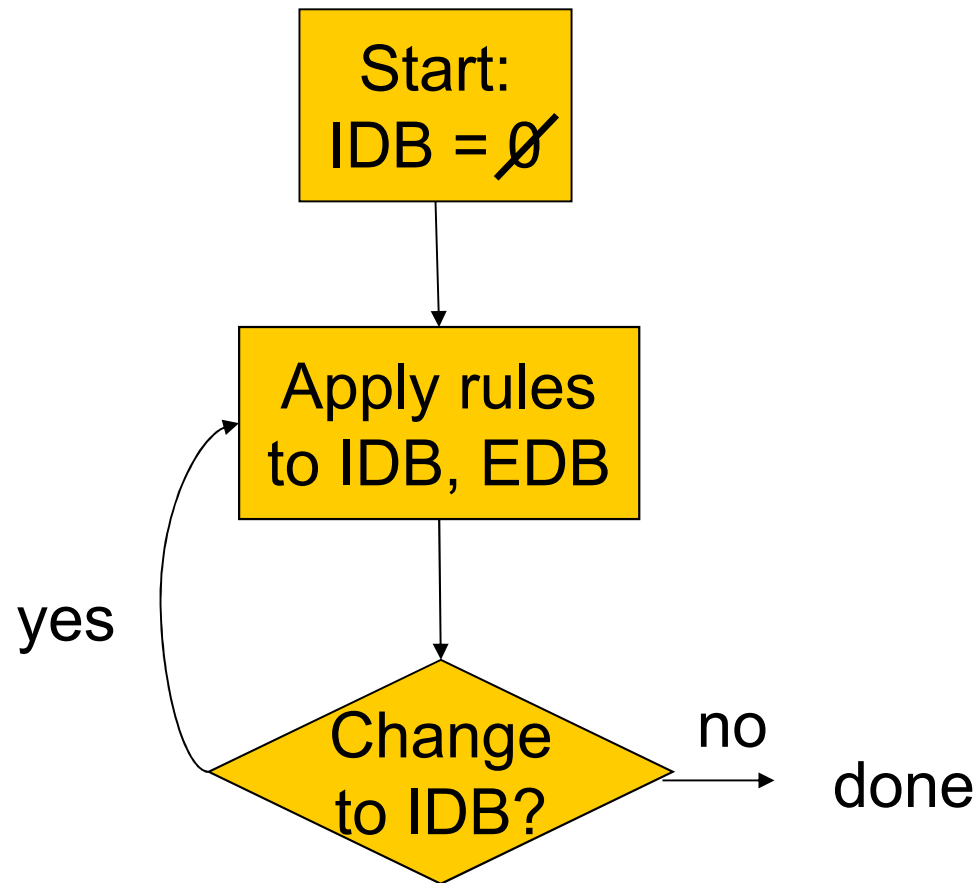


Nonrecursive

Evaluating Recursive Rules

- ▶ The following works when there is no negation:
 1. Start by assuming all IDB relations are empty.
 2. Repeatedly evaluate the rules using the EDB and the previous IDB, to get a new IDB.
 3. End when no change to IDB.

The “Naïve” Evaluation Algorithm



Seminaive Evaluation

- ▶ Since the EDB never changes, on each round we only get new IDB tuples if we use at least one IDB tuple that was obtained on the previous round.
- ▶ Saves work; lets us avoid rediscovering *most* known facts.
 - ▶ A fact could still be derived in a second way.

Example: Evaluation of Cousin

- ▶ We'll proceed in rounds to infer Sib facts and Cousin facts.

Remember the rules:

$Sib(x,y) \leftarrow Par(x,p) \text{ AND } Par(y,p) \text{ AND } x \neq y$

$Cousin(x,y) \leftarrow Sib(x,y)$

$Cousin(x,y) \leftarrow Par(x,xp) \text{ AND } Par(y,yp)$
 $\text{AND } Cousin(xp,yp)$

Par Data: Parent Above Child

$Sib(x,y) \leftarrow Par(x,p) \text{ AND } Par(y,p) \text{ AND } x \neq y$

$Cousin(x,y) \leftarrow Par(x,xp) \text{ AND } Par(y,yp) \text{ AND } Cousin(xp,yp)$

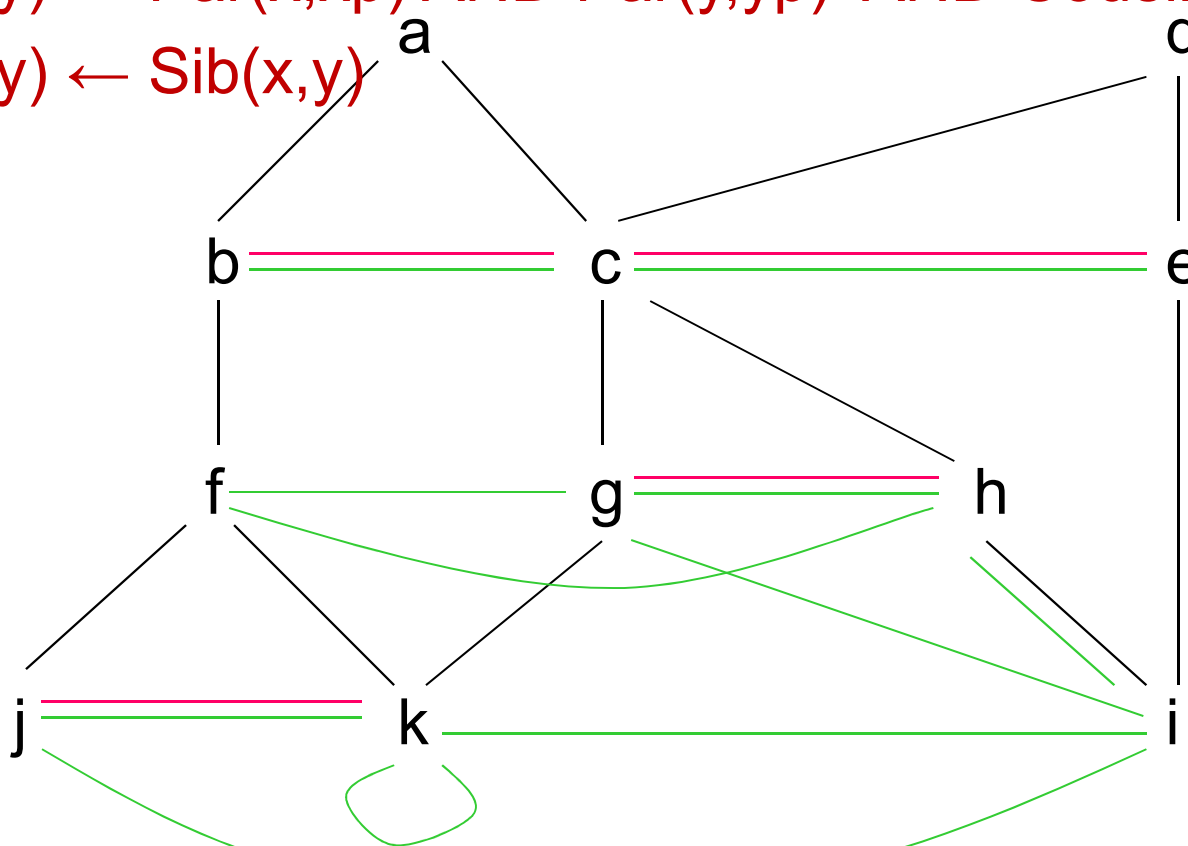
$Cousin(x,y) \leftarrow Sib(x,y)$

Round 1

Round 2

Round 3

Round 4



SQL-99 Recursion

- ▶ Datalog recursion has inspired the addition of recursion to the SQL-99 standard.
- ▶ Tricky, because SQL allows negation grouping-and-aggregation, which interact with recursion in strange ways.

Form of SQL Recursive Queries

WITH

<stuff that looks like Datalog rules>

<a SQL query about EDB, IDB>

“Datalog rule” =

[RECURSIVE] <name>(<arguments>)

AS <query>

Example: SQL Recursion ---(1)

- ▶ Find Sally's cousins, using SQL like the recursive Datalog example.
- ▶ **Par(child,parent)** is the EDB.

WITH Sib(x,y) AS


SELECT p1.child, p2.child

FROM Par p1, Par p2

WHERE p1.parent = p2.parent AND

p1.child <> p2.child;

Like Sib(x,y) ←
Par(x,p) AND
Par(y,p) AND
x <> y



Example: SQL Recursion --- (2)

Required – Cousin
is recursive

WITH RECURSIVE Cousin(x,y) AS

(SELECT * FROM Sib)

UNION

(SELECT p1.child, p2.child
FROM Par p1, Par p2, Cousin
WHERE p1.parent = Cousin.x AND
p2.parent = Cousin.y)

Reflects Cousin(x,y) ←
Sib(x,y)

Reflects
Cousin(x,y) ←
Par(x, xp) AND
Par(y, yp) AND
Cousin(xp, yp)

SELECT y FROM Cousin WHERE x = 'Sally' ;

Plan to Explain Legal SQL Recursion

1. Define “monotone” recursions.
2. Define a “stratum graph” to represent the connections among subqueries.
3. Define proper SQL recursions in terms of the stratum graph.

Recursion in the SQL-99 standard

- ▶ SQL-99 szabvány csak az ún. „monoton” rekurziót támogatja (viszont Datalogban megengedett a negáció és rekurzió együtt, de igen bonyolulttá válik a lekérdezés értelmezése, mint például a rétegzés),
- ▶ SQL-99 szabvány lineáris rekurziót enged meg, vagyis az értékadás kifejezésben egyetlen rekurzív relációt használhatunk.

Tankönyv példája (10.2 fejezet)

- ▶ Jaratok(legitarsasag, honnan, hova, koltseg, indulas, erkezes) táblában repülőjáratok adatait tároljuk.
- ▶ Mely (x,y) párokra lehet eljutni x városból y-ba?
- ▶ `Eljut(x, y) <- Jaratok(_, x, y, _, _, _)`
`Eljut(x, y) <- Eljut(x, z) AND Jaratok(_, z, y, _, _, _)`
- ▶ WITH RECURSIVE Eljut AS
 (SELECT honnan, hova FROM Jaratok
 UNION
 (SELECT Eljut.honnan, Jaratok.hova
 FROM Eljut, Jaratok
 WHERE Eljut.hova = Jaratok.honnan)
SELECT * FROM eljut;