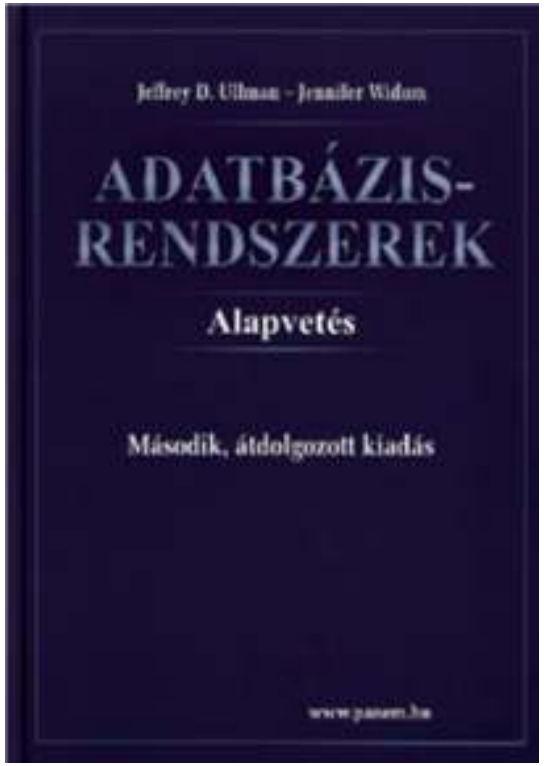
Rekurzió az SQL-ben

Tankönyv: Ullman-Widom:
Adatbázisrendszerek Alapvetés
Második, átdolgozott kiadás,
Panem, 2009



10.2. Rekurzió az SQL-ben, az „Eljut”-feladat

Az „Eljut-feladat”

Tankönyv 10.2. fejezet példája (az ELJUT feladat)

- Jaratok(legitarsasag, honnan, hova, koltseg, indulas, erkezes) táblában repülőjáratok adatait tároljuk.

  Mely (x,y) párokra lehet eljutni x városból y városba?

- Ezt egy relációs algebrai kifejezésként nem tudjuk megadni zárt alakban, klasszikus SQL SELECT utasítással sem tudjuk kifejezni, csak azt tudjuk, hogy átszállás nélkül, egy átszállással, két átszállással, stb... (lásd köv.oldalon)

Az „Eljut-feladat”

```
select distinct honnan, hova
    from jaratok
  union
  select j1.honnan, j2.hova
    from jaratok j1, jaratok j2
    where j1.hova=j2.honnan
  union
  select j1.honnan, j3.hova
    from jaratok j1, jaratok j2, jaratok j3
    where j1.hova=j2.honnan
    and j2.hova=j3.honnan
  -- union stb... Ezt így nem lehet felírni…
```

Milyen fontos rekurzív feladatok vannak?

I. Hierarchiák bejárása

- Leszármazottak-ősök ParentOf(parent,child)
  - Find all of Mary's ancestors
- Vállalati hierarchia felettes-beosztott

  Employee(ID,salary)

  Manager(mID,eID)

  Project(name,mgrID)
  - Find total salary cost of project 'X'
- Alkatrész struktúra (mely alkatrésznek mely alkatrész része)

Milyen fontos rekurzív feladatok vannak?

II. Gráf jellegű bejárások

- Repülőgép járatok, eljut-feladat

  Flight(orig,dest,airline,cost)
  - Find cheapest way to fly from ‘A’ to ‘B’
- Közösségi hálók

  Ki-kinek az ismerőse, Twitterben ki-kit követ

Kiegészítés a gráf adatbázisokról

- Gráfok könnyen megadhatók relációs táblával, a gráf lekérdezések egyre gyakoribb feladatok, ezek relációs megoldása hatékonysági kérdés. Vannak kimondottan gráf-adatbázisok.

Rekurzió az SQL-99 szabványban

- SQL-99 szabvány tartalmazza a lekérdezések rekurzív meghatározásának lehetőségét, például IBM DB2 ezeket az előírásokat implementálta
- SQL-99 szabvány lineáris rekurziót enged meg, vagyis az értékadás kifejezésben egyetlen rekurzív relációt használhatunk és még csak az ún. „monoton” rekurziót támogatja (vagyis nem használhatunk különbséget és csoportosítást)

Az „Eljut-feladat” Datalogban

Tankönyv 10.2. fejezet példája (az ELJUT feladat)

- Jaratok(legitarsasag, honnan, hova, koltseg, indulas, erkezes) táblában repülőjáratok adatait tároljuk.

  Mely (x,y) párokra lehet eljutni x városból y városba?

- Datalogban felírva (monoton és lineáris rekurzió)

  Eljut(x, y) <- Jaratok(l, x, y, k, i, e)
  Eljut(x, y) <- Eljut(x, z) AND Jaratok(l, z, y, k, i, e)

- Vagy másképp felírva Datalogban (ez nem lineáris)

  Eljut(x, y) <- Jaratok(_, x, y, _, _, _)
  Eljut(x, y) <- Eljut(x, z) AND Eljut(z, y)

Az „Eljut feladat” SQL-99 szabványban

- Lineáris és monoton rekurzió átírható SQL-be:
  Eljut(x, y) <- Jaratok(l, x, y, k, i, e)
  Eljut(x, y) <- Eljut(x, z) AND Jaratok(l, z, y, k, i, e)

- Hova, mely városokba tudunk eljutni Budapestről?

```
WITH RECURSIVE Eljut AS
        (SELECT honnan, hova FROM Jaratok
      UNION
          SELECT Eljut.honnan, Jaratok.hova
          FROM Eljut, Jaratok
          WHERE Eljut.hova = Jaratok.honnan)
 SELECT hova FROM Eljut WHERE honnan='Bp';
```

Egy másik példa rekurzióra

- EDB: Par(c,p) = *p* is a parent of *c*.
- Generalized cousins: people with common ancestors one or more generations back:

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

Cousin(x,y) ← Sib(x,y)

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

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>

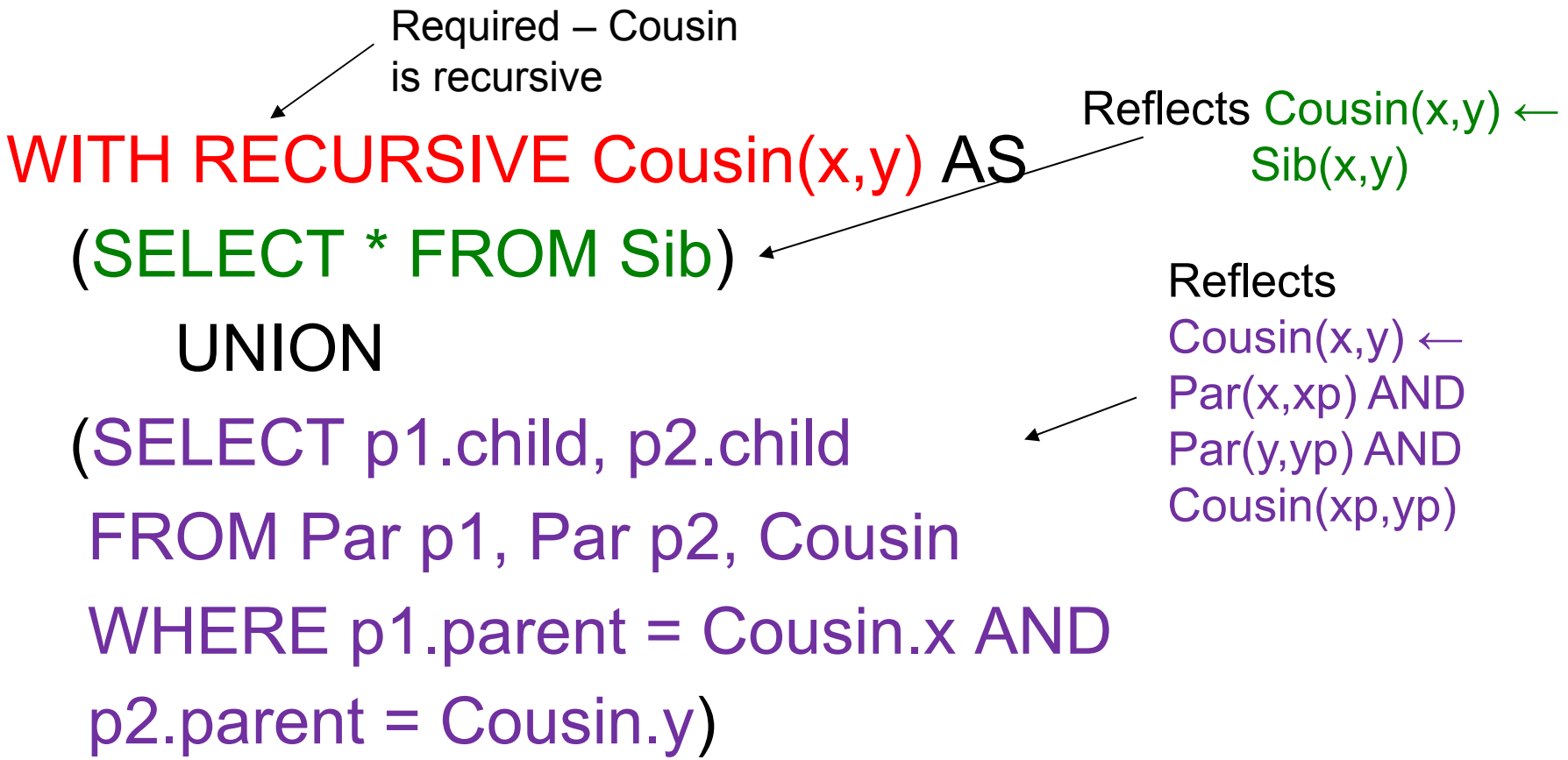
Példa: SQL Rekurzió ---1

- 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;
--- Folyt.köv.oldalon
--- WITH RECURSIVE Cousin(x,y) …
```

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

Példa: SQL Recursion ---2




```
SELECT y FROM Cousin WHERE x = Sally';
```

Plan to Explain Legal SQL Recursion

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

Oracle megoldások: WITH utasítással

- Az Oracle SQL a WITH RECURSIVE utasítást nem támogatja, ott másképpen oldották meg WITH utasítással (Oracle 11gR2 verziótól)
- with eljut (honnan, hova) as
  (select honnan, hova from jaratok
  union all
  select jaratok.honnan, eljut.hova
  from jaratok, eljut
  where jaratok.hova=eljut.honnan
)
  SEARCH DEPTH FIRST BY honnan SET SORTING
  CYCLE honnan SET is_cycle TO 1 DEFAULT 0
  select distinct honnan, hova from eljut order by honnan;

Oracle megoldások: connect by

- SELECT DISTINCT hova FROM jaratok
  WHERE HOVA <> 'DAL'
  START WITH honnan = 'DAL'
  CONNECT BY NOCYCLE PRIOR hova = honnan;

- SELECT LPAD(' ', 4*level) || honnan, hova,
  level-1 Atszallasok,
  sys_connect_by_path(honnan||'->'||hova, '/'),
  connect_by_isleaf, connect_by_iscycle
  FROM jaratok
  START WITH honnan = 'SF'
  CONNECT BY NOCYCLE PRIOR hova = honnan;