

Datalog lekérdezőnyelv

Relációs algebrai műveletek kifejezése Datalog lekérdezésekkel (szabályokkal)

$R \cap S$ metszetnek megfelelő szabály:

Válasz(x_1, \dots, x_n) $\leftarrow R(x_1, \dots, x_n)$ AND $S(x_1, \dots, x_n)$

$R - S$ különbségnek megfelelő szabály:

Válasz(x_1, \dots, x_n) $\leftarrow R(x_1, \dots, x_n)$ AND NOT $S(x_1, \dots, x_n)$

$R \cup S$ uniónak megfelelő **két szabály**:

Válasz(x_1, \dots, x_n) $\leftarrow R(x_1, \dots, x_n)$

Válasz(x_1, \dots, x_n) $\leftarrow S(x_1, \dots, x_n)$

$\sigma_{X_i=X_j}(R)$ kiválasztásnak megfelelő szabály:

Válasz(x_1, \dots, x_n) $\leftarrow R(x_1, \dots, x_n)$ AND $X_i = X_j$

$\pi_{X_1, X_2}(R)$ vetítésnek megfelelő szabály:

Válasz(x_1, x_2) $\leftarrow R(x_1, x_2, \dots, x_n)$

$R \times S$ szorzatnak megfelelő szabály:

Válasz($x_1, \dots, x_n, y_1, \dots, y_m$) $\leftarrow R(x_1, x_2, \dots, x_n)$ AND $S(y_1, \dots, y_m)$

$R \bowtie S$ -nak megfelelő szabály $R(x_1, \dots, x_n, z_1, \dots, z_k)$ és $S(y_1, \dots, y_m, z_1, \dots, z_k)$ esetén

Válasz($x_1, \dots, x_n, y_1, \dots, y_m, z_1, \dots, z_k$) $\leftarrow R(x_1, \dots, x_n, z_1, \dots, z_k)$ AND $S(y_1, \dots, y_m, z_1, \dots, z_k)$

A fentiek alapján, ha egy lekérdezést ki tudunk fejezni **relációs algebrai műveletekkel**, akkor ki tudjuk azokat fejezni **Datalog lekérdezéssel is**.

Írjunk fel néhány egyszerű lekérdezést Datalog program segítségével a **Szeret(név, gyümölcs)** relációra:

Kik szeretik az almát?

Almaszereto(X) \leftarrow Szeret(X, 'alma') (vagy Almaszereto(X) \leftarrow Szeret(X, Y) AND Y='alma')

Kik azok, akik nem szeretik az almát? (de valami mást igen)

Almat_NemSz(X) \leftarrow Szeret(X, Y) AND NOT Szeret(X, 'alma')

Kik szeretik az almát és a körtét?

AlmaKorte(X) \leftarrow Szeret(X, 'alma') AND Szeret(X, 'körte')

Kik szeretik az almát vagy a körtét?

AlmavagyKorte(X) \leftarrow Szeret(X, 'alma')

AlmavagyKorte(X) \leftarrow Szeret(X, 'körte')

Kik azok akik szeretik az almát, de a körtét nem szeretik?

Alma_KörteNSz(X) ← Szeret(X, 'alma') AND NOT Szeret(X, 'körte')

Kik szeretnek legalább kétféle gyümölcsöt?

KettotSzeret(X) ← Szeret(X, Y) AND Szeret(X, Z) AND Y <> Z

Kik szeretnek legalább háromféle gyümölcsöt?

HarmatSzeret(X) ← Szeret(X, Y) AND Szeret(X, Z) AND Szeret(X, W)
AND Y <> Z AND Z <> W AND Y <> W

Kik szeretnek legfeljebb kétféle gyümölcsöt?

HarmatSzeret(X) ← Szeret(X, Y) AND Szeret(X, Z) AND Szeret(X, W)
AND Y <> Z AND Z <> W AND Y <> W

LegfKettot(X) ← Szeret(X, Y) AND NOT HarmatSzeret(X)

A fenti Datalog lekérdezés kifejezése **SQL WITH utasítással:**

WITH

```
HarmatSzeret(N) AS (  
  SELECT DISTINCT sz1.nev FROM szeret sz1, szeret sz2, szeret sz3  
  WHERE sz1.nev=sz2.nev AND sz2.nev=sz3.nev  
  AND sz1.gyumolcs <> sz2.gyumolcs AND sz2.gyumolcs <> sz3.gyumolcs  
  AND sz1.gyumolcs <> sz3.gyumolcs)  
SELECT nev FROM szeret MINUS SELECT N FROM HarmatSzeret;
```

Kik szeretnek pontosan kétféle gyümölcsöt?

KettotSzeret(X) ← Szeret(X, Y) AND Szeret(X, Z) AND Y <> Z

HarmatSzeret(X) ← Szeret(X, Y) AND Szeret(X, Z) AND Szeret(X, W)
AND Y <> Z AND Z <> W AND Y <> W

PontKettot(X) ← KettotSzeret(X) AND NOT HarmatSzeret(X)

A fenti Datalog lekérdezés kifejezése **SQL WITH utasítással:**

WITH

```
KettotSzeret(N) AS (  
  SELECT DISTINCT sz1.nev FROM szeret sz1, szeret sz2  
  WHERE sz1.nev=sz2.nev AND sz1.gyumolcs <> sz2.gyumolcs),  
HarmatSzeret(N) AS (  
  SELECT DISTINCT sz1.nev FROM szeret sz1, szeret sz2, szeret sz3  
  WHERE sz1.nev=sz2.nev AND sz2.nev=sz3.nev  
  AND sz1.gyumolcs <> sz2.gyumolcs AND sz2.gyumolcs <> sz3.gyumolcs  
  AND sz1.gyumolcs <> sz3.gyumolcs)  
SELECT N FROM KettotSzeret MINUS SELECT N FROM HarmatSzeret;
```

Kik szeretnek minden gyümölcsöt?

NemSzeret(N,G) ← Szeret(N,X) AND Szeret(Y,G) AND NOT Szeret(N,G)

MindentSzeret(N) ← Szeret(N,X) AND NOT NemSzeret(N,Y)

A fenti Datalog lekérdezés kifejezése **SQL WITH utasítással:**

```

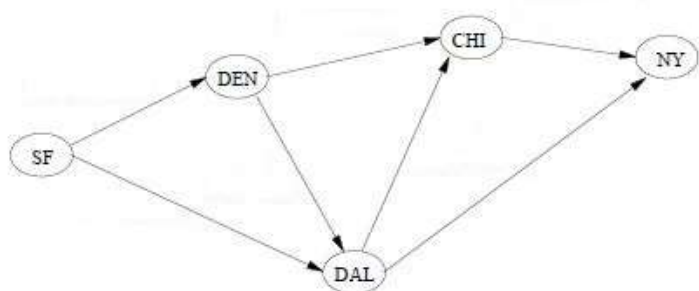
WITH
Nemszeret(N,G) AS (
  SELECT sz1.nev, sz2.gyumolcs FROM szeret sz1, szeret sz2
  MINUS
  SELECT nev, gyumolcs FROM szeret)
SELECT nev FROM szeret MINUS SELECT N FROM Nemszeret;

```

Rekurzív Datalog

A Datalog programok segítségével rekurzív lekérdezéseket is ki tudunk fejezni. Tekintsük az alábbi táblát, ami repülőgép járatok adatait tartalmazza:

JARATOK(honnan, hova)



Adjuk meg azokat a város párokat, amelyek esetén az elsőből el lehet jutni a másodikba.

```

Eljut(X,Y) ← Jaratok(X,Y)
Eljut(X,Y) ← Jaratok(X,Z) AND Eljut(Z,Y) AND X <> Y

```

A fenti rekurzív Datalog programot az SQL szabvány szerint az alábbi WITH utasítással fejezhetjük ki, amelyben a rekurzívan megadott ELJUT reláció is szerepel. Egyes adatbáziskezelők ezt a szintaxist támogatják, az Oracle egy kicsit eltérőt, lásd lejjebb.

```

WITH RECURSIVE eljut(honnan, hova) AS
(SELECT honnan, hova FROM jaratok
 UNION
 SELECT jaratok.honnan, eljut.hova FROM jaratok, eljut
 WHERE jaratok.hova = eljut.honnan AND jaratok.honnan <> eljut.hova
 )
SELECT honnan, hova FROM eljut order by 1;

```

A fenti típusú lekérdezést **Oracle-ben egy kicsit eltérő szintaxissal**, de hasonló szerkezetű lekérdezéssel valósíthatjuk meg:

```

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 AND jaratok.honnan <> eljut.hova
 )
CYCLE honnan SET van_kor TO 'I' DEFAULT 'N'
SELECT distinct honnan, hova FROM eljut order by 1;

```

Példaadatok a táblához:

```
CREATE TABLE jaratok(legitarsasag VARCHAR2(10), honnan VARCHAR2(15),
                      hova VARCHAR2(15), koltseg NUMBER);

INSERT INTO jaratok VALUES('Lufthansa', 'San Francisco', 'Denver', 1000);
INSERT INTO jaratok VALUES('Lufthansa', 'San Francisco', 'Dallas', 10000);
INSERT INTO jaratok VALUES('Lufthansa', 'Denver', 'Dallas', 500);
INSERT INTO jaratok VALUES('Lufthansa', 'Denver', 'Chicago', 2000);
INSERT INTO jaratok VALUES('Lufthansa', 'Dallas', 'Chicago', 600);
INSERT INTO jaratok VALUES('Lufthansa', 'Dallas', 'New York', 2000);
INSERT INTO jaratok VALUES('Lufthansa', 'Chicago', 'New York', 3000);
INSERT INTO jaratok VALUES('Lufthansa', 'Chicago', 'Denver', 2000);
```

Egészítsük ki az ELJUT eredményt a **költségekkel**, vagyis most az is érdekel minket, hogy honnan hova milyen költséggel tudunk eljutni. ELJUT(honnan, hova, költség)

$Eljut(X,Y,K) \leftarrow Jaratok(X,Y,K)$

$Eljut(X,Y,K) \leftarrow Jaratok(X,Z,Q) \text{ AND } Eljut(Z,Y,R) \text{ AND } K=Q+R \text{ AND } X \neq Y$

A fenti rekurzív Datalog programnak megfelelő WITH utasítás Oracle-ben.

```
WITH eljut(honnan, hova, koltseg) AS
(
  SELECT honnan, hova, koltseg FROM jaratok
  UNION ALL
  SELECT jaratok.honnan, eljut.hova, jaratok.koltseg + eljut.koltseg
  FROM jaratok, eljut
  WHERE jaratok.hova = eljut.honnan AND jaratok.honnan <> eljut.hova
)
CYCLE honnan SET van_kor TO 'I' DEFAULT 'N'
SELECT distinct honnan, hova, koltseg, van_kor
FROM eljut ORDER BY honnan, hova;
```

A van_kor oszlopot a lekérdezés végrehajtás közben tölti fel, és ez azt jelzi, hogy az útvonal tartalmaz-e kört.