KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI
www.inf.upol.cz

Jan Laštovička
jan.lastovicka@upol.cz
www.inf.upol.cz/lide/jan-lastovicka
17. listopadu 12, 771 46 Olomouc

Databázové systémy

3. Projekce a spojení

1 Projekce

Připomeňme si, že n-tice je množina komponent. Vezmeme-li libovolnou její podmnožinu, dostaneme opět n-tici. Uvažujme například n-tici t vyjádřenou tabulkou:

name	age	street
Anna	3	Kosinova

Atribut `street` je typu `varchar(10)` a udává v jaké ulici dítě bydlí. Vidíme, že n-tice t je tříprvková množina. Můžeme vzít například její dvouprvkovou podmnožinu obsahující komponenty s atributy `name` a `street` a dostaneme n-tici:

name	street
Anna	Kosinova

Obecně uvažujme n-tici t nad $A_1, \ldots, A_n$ a m různých atributů $B_1, \ldots, B_m$ takových, že $\{B_1, \ldots, B_m\} \subseteq \{A_1, \ldots, A_n\}$. *Projektcí n-tice t na $B_1, \ldots, B_m$* rozumíme m-tici t', která obsahuje právě komponenty n-tice t s atributy $B_1, \ldots, B_m$. Tedy dvojice

name	street
Anna	Kosinova

je projekcí trojce

name	age	street
Anna	3	Kosinova

na `name` a `street`.

Pomocí projekce n-tice již můžeme snadno definovat projekci relace. Pokud r je relace nad $A_1, \ldots, A_n$ a $B_1, \ldots, B_m$ je m různých atributů takových, že $\{B_1, \ldots, B_m\} \subseteq \{A_1, \ldots, A_n\}$, pak *projekcí r na $B_1, \ldots, B_m$* rozumíme relaci r' nad $B_1, \ldots, B_m$, jejíž tělo je tvořeno projekcemi všech n-tic t v těle relace r na $B_1, \ldots, B_m$.

Například projekce relace

name	age	street
Anna	3	Kosinova
Bert	4	Mahlerova
Cyril	4	Kosinova

na `name` a `street` je relace

name	street
Anna	Kosinova
Bert	Mahlerova
Cyril	Kosinova

Pokud v je výraz, jehož hodnota je relace r nad $A_1, \ldots, A_n$, a $B_1, \ldots, B_m$ je m různých atributů takových, že $m \geq 1$, $\{B_1, \ldots, B_m\} \subseteq \{A_1, \ldots, A_n\}$ a R je jméno relace, pak

```
( SELECT DISTINCT B1, ..., Bm
  FROM  v AS R )
```

je relační výraz, jehož hodnota je projekce r na $B_1, \ldots, B_m$. Jméno R pojmenovává hodnotu výrazu v a zatím nehraje žádnou roli.

Předpokládáme, že `child` je relační proměnná obsahující relaci z ukázky projekce:

```
# TABLE child;

 name   | age |  street
--------+-----+-----------
 Bert   |   4 | Mahlerova
 Cyril  |   4 | Kosinova
 Anna   |   3 | Kosinova
(3 rows)
```

Pak projekci relace `child` na `name` a `street` dostaneme následovně:

```
# SELECT DISTINCT name, street
  FROM ( TABLE child ) AS t;


 name   |  street
--------+-----------
 Cyril | Kosinova
 Anna   | Kosinova
 Bert   | Mahlerova
(3 rows)
```

Počet n-tic v těle relace se nazývá *kardinalita relace*. Například relace `child` má kardinalitu tři. Kardinalita projekce relace r nemůže být větší než kardinalita r. Může však být menší. Například:

```
# SELECT DISTINCT age
  FROM ( TABLE child ) AS t;

 age
-----
  3
  4
(2 rows)
```

Co je hodnotou následujícího výrazu?

```
SELECT DISTINCT name
FROM ( SELECT DISTINCT name, street
       FROM ( TABLE child) AS t ) AS t
```

Uvažujme následující dvě relace:

child1

name	age	street
Anna	3	Kosinova
Bert	4	Mahlerova
Cyril	4	Kosinova

child2

name	age
Bert	4
Cyril	4
Daniela	5

Relace `child1` a `child2` nemůžeme přímo sjednotit, protože mají různé záhlaví. Můžeme ale sjednotit projekci relace `child1` s relací `child2`:

```
# ( SELECT DISTINCT name, age
    FROM ( TABLE child1 ) AS t )
    UNION
  ( TABLE child2 );

  name   | age
---------+-----
 Cyril   |   4
 Bert    |   4
 Anna    |   3
 Daniela |   5
(4 rows)
```

2 Spojení

Uvažujme dvě následující relační proměnné `parent` a `child`.

`parent`

parent_name	child_name
Pavel	Anna
Monika	Bert
Petr	Bert
Marie	Daniela

`child`

child_name	child_age
Anna	3
Bert	4
Cyril	4

Relace `parent` vyjadřuje kdo je rodič jakého dítěte. Například Monika je rodič Berta. Máme-li k dispozici tyto dvě relace můžeme jistě zodpovědět otázku, který rodič má čtyřleté dítě. K zodpovězení otázky potřebuje informace z obou tabulek. Výhodné je uvažovat relaci r:

parent_name	child_name	child_age
Pavel	Anna	3
Monika	Bert	4
Petr	Bert	4

Záhlaví relace r je sjednocením záhlaví relací parent a child. Tělo relace r obsahuje všechny n-tice t nad `parent_name`, `child_name` a `child_age` takové, že projekce t na `parent_name` a `child_name` je v těle relace parent a projekce t na `child_name` a `child_age` je v těle relace child.

Obecně uvažujme relaci r_1 nad $A_1, \ldots, A_n$ a relaci r_2 nad $B_1, \ldots, B_m$ a předpokládejme, že $\{A_{(n-k)+1}, \ldots, A_n\} = \{B_1, \ldots, B_k\} = \{A_1, \ldots, A_n\} \cap \{B_1, \ldots, B_m\}$. Tedy předpokládáme, že společné atributy číslujeme jako poslední mezi atributy A_i a první mezi atributy B_j. Pak *spojení relací* r_1 *a* r_2 rozumíme relaci r nad $A_1, \ldots, A_n, B_{k+1}, \ldots, B_m$, kde tělo relace r tvoří množina n-tice t taková, že

1. projekce t na $A_1, \ldots, A_n$ náleží do těla r_1,
2. projekce t na $B_1, \ldots, B_m$ náleží do těla r_2.

Relace nad `parent_name`, `child_name` a `child_age` z předchozího příkladu je spojení relací `parent` a `child`.

Pokud v_1 a v_2 jsou relační výrazy a R_1, R_2 dvě různá jména relací, pak

```
( SELECT  *
  FROM    v1 AS R1
  NATURAL JOIN v2 AS R2 )
```

je relační výraz, jehož hodnota je spojení hodnot výrazů v_1 a v_2. Jak už jsme si zvykli, tak R_1 a R_2 pojmenovávají hodnoty výrazů v_1 a v_2 a zatím nehrají žádnou roli.

Například:

```
# SELECT  *
  FROM  ( TABLE parent ) AS t1
  NATURAL JOIN ( TABLE child) AS t2;

 child_name | parent_name | child_age
------------+-------------+-----------
 Anna       | Pavel       |         3
 Bert       | Monika      |         4
 Bert       | Petr        |         4
(3 rows)
```

Tady je odpověď na otázku, kteří rodičové mají čtyřleté dítě:

```
# SELECT DISTINCT parent_name
  FROM ( SELECT *
         FROM ( SELECT  *
                FROM  ( TABLE parent ) AS t1
                NATURAL JOIN ( TABLE child ) AS t2 ) AS t
         WHERE  child_age = 4 ) AS t;

 parent_name
-------------
 Petr
 Monika
(2 rows)
```

Spojení relací přináší dva okrajové případy. První případ je, že záhlaví jedné relace je disjunktní se záhlavím druhé relace. Uvažujme dvě relace r_1, r_2, kde záhlaví r_1 je disjunktní se záhlavím r_2. V této situaci pro každý pár n-tici t_1 a t_2 z těl relací r_1 a r_2 existuje n-tice t z těla spojení r_1 a r_2 taková, že t_1 a t_2 jsou příslušnými projekcemi t. Spojení r_1 a r_2 se v této situaci příznačně nazývá *kartézským součinem*.

Například uvažujme relační proměnné `toy` (hračka) a `child` (dítě):

```
# TABLE toy;

 toy_name
----------
 balon
 lopatka
(2 rows)

# TABLE child;

 child_name
------------
 Anna
 Bert
 Cyril
(3 rows)
```

Takto dostaneme všechny možné kombinace hraček a dětí:

```
# SELECT  *
  FROM  ( TABLE toy) AS t1
  NATURAL JOIN ( TABLE child ) AS t2;

 toy_name | child_name
----------+------------
 balon    | Anna
 balon    | Bert
 balon    | Cyril
 lopatka  | Anna
 lopatka  | Bert
 lopatka  | Cyril
(6 rows)
```

Druhým okrajovým případem je, když spojujeme relace stejného typu. Uvažujme dvě relace r_1 a r_2 nad $A_1, \ldots, A_n$. Pak spojením r_1 a r_2 bude relace r opět nad $A_1, \ldots, A_n$. Tělo relace r bude obsahovat n-tice t takové, že projekce t na $A_1, \ldots, A_n$ bude v těle r_1 a současně projekce t na $A_1, \ldots, A_n$ bude v těle r_2. Projekce t na

$A_1, \ldots, A_n$ je ale přímo rovna t. Tedy tělo r bude obsahovat n-tice t nad $A_1, \ldots, A_n$, které jsou současně v těle r_1 i v těle r_2. Neboli tělo r bude průnikem těl r_1 a r_2. V této situaci je spojení relací rovno jejich průniku.

3 Přejmenování atributů

Vrátíme se k relačním proměnným `parent` a `child` z příkladu spojení relací. Uvažujme ale záhlaví relace `child` s původními názvy atributů:

`parent`

parent_name	child_name
Pavel	Anna
Monika	Bert
Petr	Bert
Marie	Daniela

`child`

name	age
Anna	3
Bert	4
Cyril	4

Spojení relací `parent` a `child` nyní nepřinese kýžený výsledek, protože záhlaví obou relací jsou disjunktní. Potřebujeme přejmenovat atribut `name` v záhlaví relace `child` na `child_name`.

Vezměme si záhlaví složené z atributů $A_1, \ldots, A_n$ a atributy $B_1, \ldots, B_m$ takové, že $m \leq n$ a atribut A_i je stejného typu jako atribut B_i pro každé $1 \leq i \leq m$ a atributy $B_1, \ldots B_m, A_{m+1}, \ldots, A_n$ mají po dvou různá jména. Dále uvažujme n-tici t nad $A_1, \ldots, A_n$.

Přejmenováním atributů $A_1, \ldots, A_m$ *n-tice* t *na* $B_1, \ldots, B_m$ obdržíme n-tici t' nad $B_1, \ldots, B_m, A_{m+1}, \ldots, A_n$ takovou, že

1. atributu B_i přiřazuje stejnou hodnotu jako n-tice t atributu A_i pro každé $1 \leq i \leq m$,
2. atributu A_i přiřazuje stejnou hodnotu jako n-tice t atributu A_i pro každé $m < i \leq n$.

Například přejmenováním atributu `name` na `child_name` v n-tici

name	age
Anna	3

nad `name` a `age` vznikne n-tice nad `child_name` a `age`:

child_name	age
Anna	3

Nyní se můžeme pustit do přejmenování atributů v záhlaví relace. Uvažujme relaci r nad $A_1, \ldots, A_n$. *Přejmenováním atributů* $A_1, \ldots, A_m$ *v záhlaví relace* r *na* $B_1, \ldots, B_m$ obdržíme relaci r' nad $B_1, \ldots, B_m, A_{m+1}, \ldots, A_n$. Tělo relace r' obsahuje právě ty n-tice, které vzniknou přejmenováním atributů $A_1, \ldots, A_n$ na $B_1, \ldots, B_m$ nějaké n-tice v těle r.

Například přejmenováním atributu `name` na `child_name` v záhlaví relace `child` obdržíme relaci:

child_name	age
Anna	3
Bert	4
Cyril	4

Pokud v je relační výraz, jehož hodnota je relace nad $A_1, \ldots, A_n$, a R je jméno relace, pak

```
( SELECT A1 AS B1, ..., Am AS Bm, Am+1, ..., An
  FROM   v AS R )
```

je *relační výraz*, jehož hodnota je relace vzniklá přejmenováním atributů $A_1, \ldots, A_m$ na $B_1, \ldots, B_m$ v záhlaví relace r. Jak jsme zvyklí, tak R dává jméno hodnotě výrazu v a nehraje zatím žádnou roli.

Například:

```
# SELECT name AS child_name, age
  FROM ( TABLE child ) AS t;

 child_name | age
------------+-----
 Anna       |   3
 Bert       |   4
 Cyril      |   4
(3 rows)
```

Nyní již můžeme spojit relaci `parent` s předchozí relací:

```
# SELECT  *
  FROM  ( TABLE parent) AS t1
  NATURAL JOIN ( SELECT name AS child_name, age
                 FROM ( TABLE child ) AS t ) AS t2;

 child_name | parent_name | age
------------+-------------+-----
 Anna       | Pavel       |   3
 Bert       | Monika      |   4
 Bert       | Petr        |   4
(3 rows)
```

Ještě doplníme přejmenování atributu `age`, aby bylo jasné, že se jedná o věk dítěte a ne rodiče:

```
# SELECT  *
  FROM  ( TABLE parent) AS t1
  NATURAL JOIN ( SELECT name AS child_name, age AS child_age
                 FROM ( TABLE child ) AS t ) AS t2;

 child_name | parent_name | child_age
------------+-------------+-----------
 Anna       | Pavel       |         3
 Bert       | Monika      |         4
 Bert       | Petr        |         4
(3 rows)
```

4 Konstantní relace

Je dána neprázdná relace r nad $A_1, \ldots, A_n$. Tělo r je množina n-tic $\{t_1, \ldots, t_m\}$. Označme pro každé $1 \leq i \leq n$ a $1 \leq j \leq m$ hodnotu v_{ij}, kterou přiřadí n-tice t_j atributu A_i. Zvolme jméno relace R. Pak

```
( SELECT *
  FROM ( VALUES ( v11, ..., v1n ),
                      ⋮
                ( vm1, ..., vmn ) ) AS R ( A1, ..., An ) )
```

je relační výraz, jehož hodnota je relace r. Jméno relace R pojmenovává relaci r a nehraje zatím žádnou roli.

Například:

```
# SELECT *
  FROM ( VALUES ( 'Anna', 3 ),
                ( 'Bert', 4 ),
                ( 'Cyril', 4 ) ) AS child ( name, age );

 name  | age
-------+-----
 Anna  |   3
 Bert  |   4
 Cyril |   4
(3 rows)
```

Takto můžeme vytvořit relaci, kde záhlaví bude mít jen jeden atribut a tělo jen jednu n-tici:

```
# SELECT *
  FROM ( VALUES ( 1 ) ) AS t ( num );

 num
--------
      1
  (1 row)
```

Uvažujme relační výraz v, kde hodnota r je relace nad $A_1, \ldots, A_n$. Zvolme atribut A_{n+1}, který není v záhlaví relace r, a hodnotu v stejného typu jako je typ atributu A_{n+1}. Pomocí konstantní relace a spojení můžeme vytvořit relaci r' nad $A_1, \ldots, A_{n+1}$. Tělo relace r' obsahuje právě ty n-tice t, kde platí, že projekce t na $A_1, \ldots, A_n$ je v těle r a t přiřazuje hodnotu v atributu A_{n+1}. Relace r' bude hodnotou následujícího výrazu.

```
SELECT  *
FROM  ( v ) AS t1
NATURAL JOIN ( SELECT *
               FROM ( VALUES ( v ) ) AS t ( A_{n+1} ) ) AS t2
```

Například přidání atributu `cons` do záhlaví relace `child`:

```
# TABLE child;

name  | age
------+-----
 Anna  |   3
 Bert  |   4
 Cyril |   4
(3 rows)

# SELECT  *
  FROM ( TABLE child ) AS t1
  NATURAL JOIN ( SELECT *
                   FROM ( VALUES ( 4 ) ) AS t ( cons ) ) AS t2;

 name  | age | cons
-------+-----+------
 Anna  |   3 |    4
 Bert  |   4 |    4
 Cyril |   4 |    4
(3 rows)
```

Můžeme získat čtyřleté děti restrikcí vzhledem k podmínce porovnání hodnot dvou atributů:

```
# SELECT *
  FROM (
         SELECT  *
         FROM ( TABLE child ) AS t1
         NATURAL JOIN ( SELECT *
                          FROM ( VALUES ( 4 ) ) AS t ( cons ) ) AS t2
       ) AS t
  WHERE age = cons;

 name  | age | cons
-------+-----+------
 Bert  |   4 |    4
 Cyril |   4 |    4
(2 rows)
```

Pomocný atribut `cons` můžeme odstranit:

```
# SELECT DISTINCT name, age
  FROM (
         SELECT *
         FROM (
                SELECT  *
                FROM ( TABLE child ) AS t1
                NATURAL JOIN (
                               SELECT *
                               FROM ( VALUES ( 4 ) ) AS t ( cons )
                             ) AS t2
              ) AS t
         WHERE age = cons
       ) AS t;

 name  | age
-------+-----
 Bert  |   4
 Cyril |   4
(2 rows)
```

Touto technikou je možné provést restrikci vzhledem k podmínce rovnosti atributu a hodnoty (`age = 4`) pomocí konstantní relace, spojení, restrikce vzhledem k rovnosti dvou atributů a projekci.

Pojmenování atributu v konstantní relaci jako `age` učiní dotaz výrazně jednodušší:

```
# SELECT  * FROM ( TABLE child ) AS t1
  NATURAL JOIN ( SELECT * FROM ( VALUES ( 4 ) ) AS t ( age ) ) AS t2;

 age | name
-----+-------
   4 | Bert
   4 | Cyril
(2 rows)
```