Резниченко Валерий Анатольевич

Лекция 2. Дедуктивные БД и рекурсивный SQL

Институт программных систем НАНУ

СОДЕРЖАНИЕ

- Дедуктивные БД
- Рекурсивный SQL

Экстенсиональные и интенсиональные понятия

- **Отец** (ребенок, родитель) – **экстенсиональное понятие**

Интенсиональные понятия:

- **Сын** – ребенок родителя
- **Брат** – сын моего отца, но не я
- **Дед** – отец моего отца
- **Внук** – сын моего сына
- **Дядя** – брат моего отца
- **Кузин** – сын моего дяди (двоюродн. брат, троюродн. брат, …)
- **Племяник** - брат моего брата

Язык Datalog (Database logic)

- **Термы** – константы и переменные (функции НЕ допускаются)

```
Sum(x,0,x)
Sum(x,next(y),next(z)) :- Sum(x,y,z)
```

- **Атомарные формулы** (атомарный предикат) – $p(t_1,\ldots,t_n)$ – интерпретируются явно заданными (экстенсиональными) отношен.

```
p(x,y,z),   p(5,y,z),   p(x,y,x)
```

- **Встроенные предикаты** (=, ≥, ≠, …)
- **Литералы** – атомарные формулы и их отрицания
- **Правила** – $q :- p_1 \,\&\, \ldots \,\&\, p_n$ $\quad (p_1 \,\&\, \ldots \,\&\, p_n \Rightarrow q)$
  - (голова правила, тело правила, подцели)

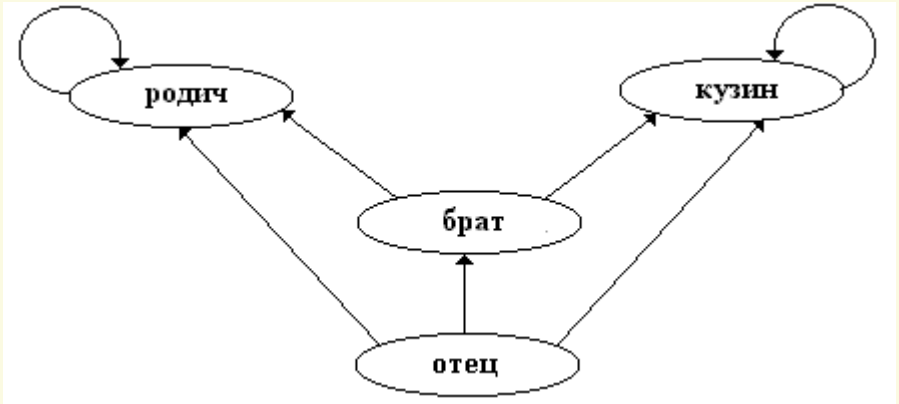
Пример:

- `брат (X,Y) :- отец(X,Z) & отец(Y,Z) & X≠Y;`

Граф зависимостей

Логическая программа:

```
брат (X,Y) :- отец(X,Z) & отец(Y,Z) & X ≠ Y
кузин(X,Y) :- отец (X,Xp) & отец (Y,Yp) & брат(Xp,Yp)
кузин(X,Y) :- отец (X,Xp) & отец (Y,Yp) & кузин(Xp,Yp)
родич(X,Y) :- брат (X,Y)
родич(X,Y) :- родич(X, Z) & отец(Y, Z)
родич(X,Y) :- родич (Z,Y) & отец(X, Z)
```



Циклы (петли), рекурсивный предикат, рекурсивная логическая программа, экстенсиональные и интенсиональные предикаты.

Безопасные правила

Надо, чтобы любое правило интерпретировалось конечным отношением (без обращения к домену)

- Правило называется ***безопасным***, если все его переменные ограничены.
- Переменная ограничена:
  - - если она входит в экстенсиональный предикат

    `p(x,y,z)`
  - - если она приравнивается к другой ограниченной переменной

    `p(x,y,z) & u=z`
  - - если она приравнивается к константе (`u=5`)
- Пример безопасного правила:

```
q(X,Y) :- p(X,Z) & W=a & Y=W
```

$T(X) = \pi_{s_2}(\sigma_{s_1=a}(P))$

Три вида правил

- Простые
- Рекурсивные
- С отрицаниями

Утверждение – простые правила выразимы через реляционную алгебру

Пример: `q(X,Y) :- p(a,X) & r(X,Z,X) & s(Y,Z)`

$$T(X) = \pi_{s_2}(\sigma_{s_1=a}(P))$$

$$U(X,Z) = \pi_{s_1,s_2}(\sigma_{s_1=s_3}(R)).$$

$$T(X) * U(X,Z) * S(Y,Z)$$

Рекурсивные правила

- ***Рекурсия*** – предикат определяется через самого себя

  `родич(X,Y) :- родич(X, Z) & отец(Y, Z)`

- Неподвижная точка
- Минимальная неподвижная точка (модель)
- Монотонные операции
- Все операции РА за исключением разности являются монотонными

Утверждение – Для вычисления рекурсивных правил необходимо реляционные выражения заключать в циклы

Правила с отрицаниями

```
одинокий мужчина(X) :- мужчина(X) & ~женатый(X,Y)
```

Все	Мужчины	Женатые	Неженатые	Неженате мужчины	Одиноке мужчины
Иван Петр Ира	Иван Петр	Петр, Ира	Иван, Иван Иван, Петр Иван, Ира Петр, Иван Петр, Петр Ира, Иван Ира, Петр, Ира, Ира	Иван, Иван Иван, Петр Иван, Ира Петр, Иван Петр, Петр	Иван Петр

Следует запретить использование переменных в отрицаемых подцелях, если эти переменные не используются в подцелях без отрицания

```
женатый мужчина(X) :- женатый(X,Y)
одинокий мужчина(X):- мужчина(X) & ~женатый мужчина(X)
```

Привила с отрицаниями – множество миним. точек

```
p(X) :- r(X) & ~q(X)
q(x) :- r(X) & ~p(X)
```

- Пусть P, Q, R являются отношениями для ИБД-предикатов p, q и ЭБД-предиката r. Пусть ЭБД-отношение R состоит из еди нственного кортежа, например, $R = \{1\}$. Тогда имеется два ре шения приведенной выше дедуктивной программы:

$$S_1 = \{P = \varnothing,\ Q = \{1\}\};$$

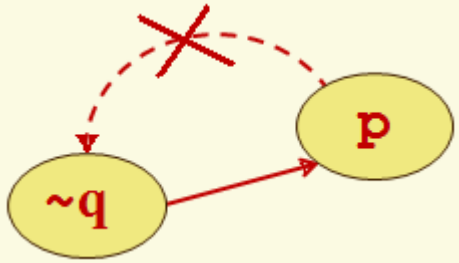
$$S_2 = \{P = \{1\},\ Q = \varnothing\}.$$

- S_1 и S_2 являются решениями уравнений:

$$P = R - Q \text{ i } Q = R - P$$

Страфицируемые отрицания

- Правило, голова которого содержит предикат p, а тело — отрицаемую подцель q, является страцифицируемым, если в графе зависимостей отсутствует путь от p к q.



- Наличие стратифицируемого отрицания дает возможность выбора среди многих минимальных точек такую, которая бу дет интерпретироваться как содержание (смысл) логической программы со стратифицируемым отрицанием

Стратификация правил

- Существует алгоритм проверки стратифицируемости логической программы, и если она стратифицируема, то и производит страти фикацию правил, то есть группирует предикаты в ***страты***, котор ые представляют собой такие максимальные множества предикат ов, что:

  1) если предикат p является головой правила с телом, содержа щим отрицаемый предикат q, то q располагается в страте бо лее низкого уровня;

  2) если предикат p является головой правила с телом, содержа щим неотрицаемый предикат q, то страта для p имеет по кра йней мере тот же уровень, что и страта для q.

- Этот алгоритм строит так называемую ***предпочтительную ми нимальную точку*** (среди множества всех возможных))

Рекурсивный SQL

- В стандарте SQL–99 было расширено определение запроса. Его синтаксис принял следующий вид:

```
[фраза_WITH] запрос
```

- фраза WITH, которая может располагаться в начале запроса, используется для достижения следующих целей:
  - описать подзапросы, которые многократно используются в самом запросе, с тем, чтобы к ним (подзапросам) можн о было обращаться в запросе по имени;
  - описать рекурсивное выполнение запроса.

Рекурсивное выполнение запросов

- Давайте рассмотрим следующий фрагмент запроса:

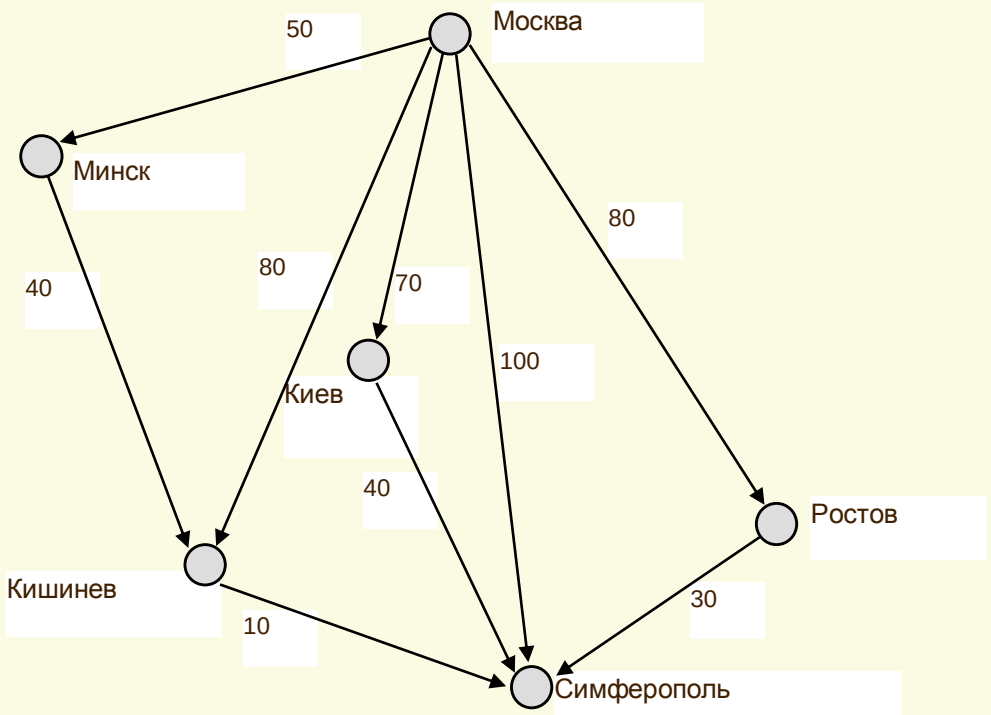
```
WITH B AS
      (SELECT ... FROM A ...
         UNION ALL
       SELECT ... FROM A,V ...)
   SELECT ....
```

- Это не рекурсивный запрос. Если теперь мы в нем заменим «V» на «B» то он по форме преобразуется в рекурсивный:

```
WITH RECURSIVE B AS
      (SELECT ... FROM A ...
          UNION ALL
       SELECT ... FROM A,B ...)
   SELECT....
```

Рекурсивные запросы: рейсовые полеты

```
CREATE TABLE FLIGHTS
     (FromCity varchar(15),
      ToCity   varchar(15),
      Cost     numeric(4));
```



Рекурсия с накоплением

- Найти самый дешевый перелет из Москвы в Симферополь

```
WITH RECURSIVE REACHABLE(FromCity, Destination, Total_Cost) AS
   (SELECT FromCity, ToCity, Cost
    FROM   FLIGHTS
    WHERE  FromCity = 'Москва'
        UNION
    SELECT inn.FromCity, out.ToCity, inn.Total_Cost + out.Cost
    FROM   REACHABLE inn, FLIGHTS out
    WHERE  inn.Destination = out.FromCity
   )
SELECT FromCity, Destination, MIN(Total_Cost)
FROM   REACHABLE
WHERE  Destination = 'Симферополь'
GROUP  BY FromCity, Destination;
```

Отрицание в рекурсии

- *Отрицание в рекурсии разрешимо, если оно применяется к таблицам, которые являются полностью известным (то есть уже вычислены).*
- **Запрос**. Города, достижимые из других городов кроме Москвы.

```
with RECURSIVE Reachable_From (FromCity, ToCity) AS
      (SELECT FromCity, ToCity
       FROM   Flights
          UNION
       SELECT inn.FromCity, out.ToCity
       FROM   Reachable_From inn, Flights out
       WHERE  inn.ToCity = out.FromCity
      )
   SELECT * FROM Reachable_From
   EXCEPT
   SELECT * FROM Reachable_From WHERE FromCity = 'Москва';
```

Рекурсия – другие возможности

- **Фиксация наличия циклов**
- **Направление поиска** (в глубину, в ширину)
- **Вкладывание** рекурсивных запросов в друга
- **Прямая** (рекурсия отношения через самого себя) и **взаимная рекурсия** (рекурсия R через S и S через R)
- **Линейная и нелинейная рекурсия** (Рекурсивный запрос является ***лине йным***, если таблица, определяемая рекурсивно, соединяется в запросе т олько один раз (то есть она перечисляется во фразе FROM только один раз). Кроме того, линейная рекурсия предполагает, что рекурсивно опре деляемая таблица не может одновременно использоваться во фразах FR OM как самого рекурсивного запроса, так и во всех его подзапросах.

Литература

- Пасічник В.В., Резніченко В.А. Організація баз даних та знань. – К.: Видавнича група BHV, 2006. – 384 с.
- Ф. Андон, В. Резниченко - Язык запросов SQL. Учеб ный курс. - СПб.: Питер, Киев: BHV, 2006. — 416 с.
- В.А. Резниченко. Рекурсивный SQL. - Інженерія програмного забезпечення. - № 4 2010 – с. – 63-81