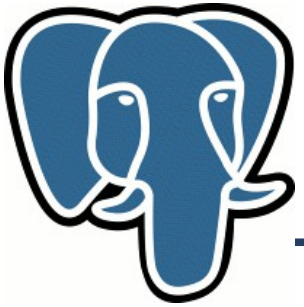
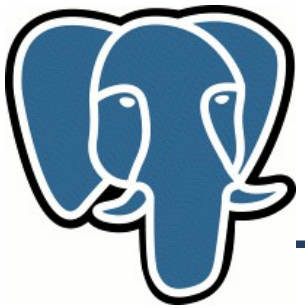


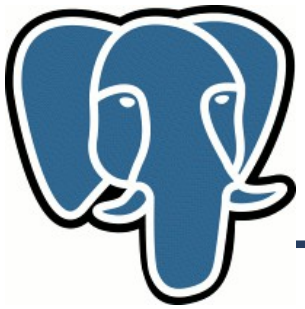
AnyArray

Фёдор Сигаев
Mail.Ru Group



Постреляционность

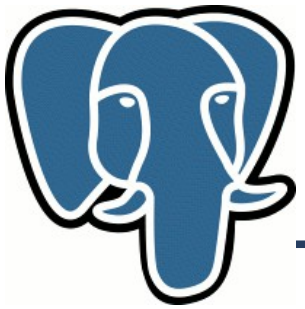
- Object-relational DBMS
- Массивы
- Табличные типы
- Наследование
- UD*
  - T
  - F
  - O
  - ...
- расширяемость



Массивы „из коробки“

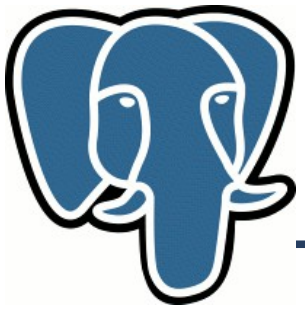
Для любых типов (и пользовательских тоже)

- '{1,2,3}', '{{1,2,3}, {3,4,5}}'
- select ('{1,2,3}'::int4[])[2];
- Select ('{1,2,3}'::int4[])[1:2];
- Select ARRAY(select ….) …
- …



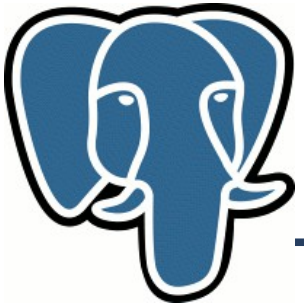
Массивы „из коробки“

- '{1}' || '{2}' , '{1}' || 2
- 10 = ANY '{1,10}'
- 10 = ALL '{1,10}'
- '{1,2}' && '{2,3}'
- '{2,3}' <@ '{1,2,3}'
- '{1,2,3}' @> '{2,3}'
- Array_*() ...



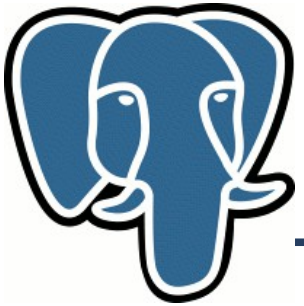
Массивы „из коробки“

- BTREE
  <, <=, =, >=, >
- GIN
  &&, <@, =, @>



intarray

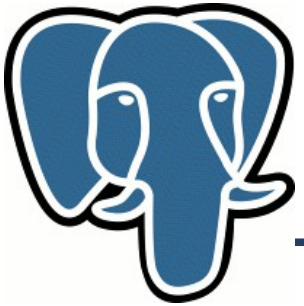
- 12 Jan 2001, v.7.1
- Одномерные, NULL-free, int4
- Полнотекстовый поиск! OpenFTS
- Спасибо, Рамблер!



Intarray

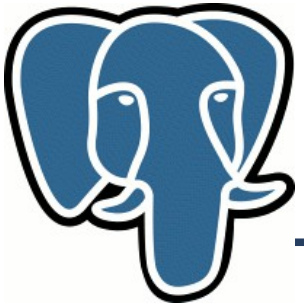
- Sort, sort_desc, sort_asc
- Uniq
- Idx
- \- вычитание массивов
- \+ union массивов

Большое количество операций и функций перенесено в ядро



Intarray

- TSQUERY like:
  '{1,2,3}'::int4[] @@ '1 & (2 | !4)'
- GIN
- GiST
  - gist__int_ops (default, up to 100)
  - gist__intbig_ops

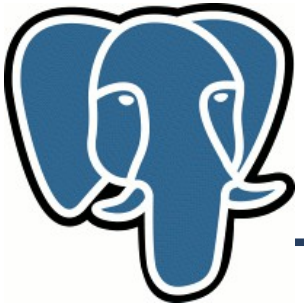


Anyarray

Повторяем все, что есть для int4.

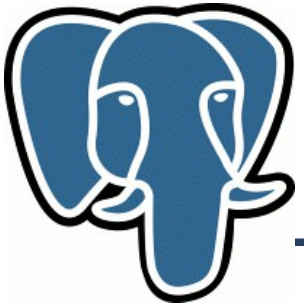
Но (элементы):

- hash_ops (GiST)
- btree_ops (GIN, sort)
- Hash/btree ops (uniq, matching etc)



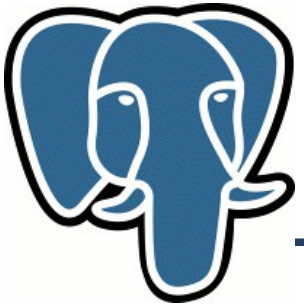
Anyarray

SMLAR



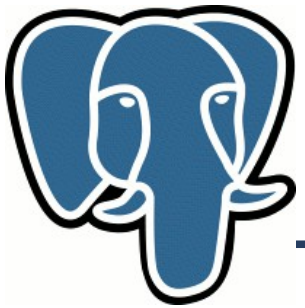
Anyarray: smlar

- Тексты в том или ином виде
- Блоги, стенки
- Товары
- Картинки/картины/фотки
- Музыка, кино, книги



Anyarray: smlar

- Экспертная оценка
  - Тяжело формализовать
- Свойства/атрибуты объектов
  - Набор атрибутов
- Пользовательский интерес/внимание (collaboration filtering, CF)
  - Рейтинг, лайки



Anyarray: smlar

- Текст –
  - Фрагменты - {fingerprints}, {lexems},{n-grams}
  - {tags},{authors},{languages},...

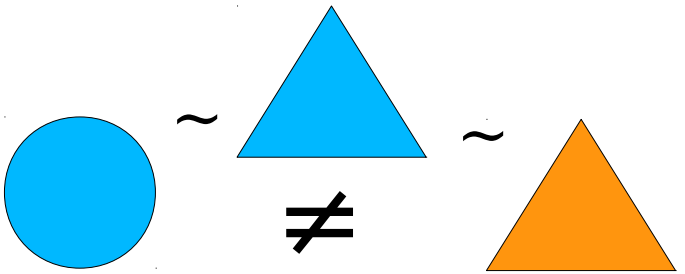
Similarity (S) – численная величина пересечения {lexems} && {lexems}

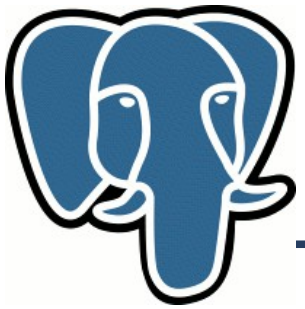
Anyarray: smlar

- Similarity $0 \le S \le 1$
  - $S \equiv 1$ – абсолютно похожие объекты (но не обязательно идентичные)
  - $S \equiv 0$ вообще ничего общего
- $S(A,B) = S(B,A)$ - симметрия
- Два объекта похожи если

$$S(A,B) \ge S_{threshold}$$

- $A \sim B$ и $A \sim C \neq B \sim C$





Обозначения

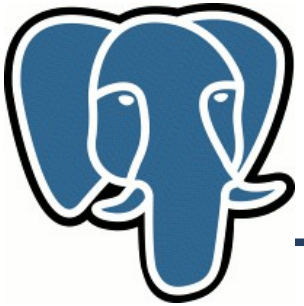
N_a, N_b - кол-во уникальных элементов в массиве

N_u – кол-во уникальных элементов в объединении

$$N_{a \text{ union } b}$$

N_i – кол-во уникальных элементов в пересечении

$$N_{a \text{ intersection } b}$$

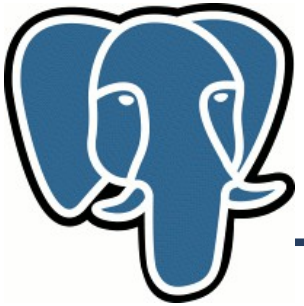


Метрика

Cosine (Ochiai):

$S(A, B) = N_i / sqrt(N_a * N_b)$

- ~ N*log(N)
- Для больших N



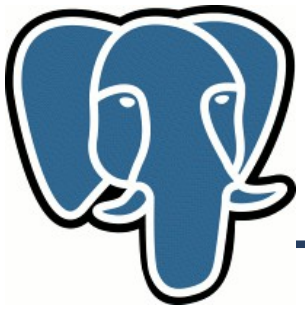
Anyarray: smlar

Функции и операторы:

- `float4 smlar(anyarray, anyarray)`
- `anyarray % anyarray`

Конфигурационные параметры:

- `anyarray.threshold = float4`
- `anyarray.type = (overlap, cosine)`

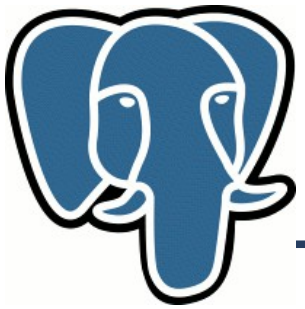


Anyarray: smlar

```
=# select smlar('{0,1,2,3,4,5,6,7,8,9}'::int[],'{0,1}'::int[]);
  smlar
----------
 0.447214
(1 row)
```

2/SQRT(10*2)=0.447214

```
SET smlar.threshold=0.6;
# select '{0,1,2,3,4,5,6,7,8,9}'::int[] % '{0,1}'::int[];
 ?column?
----------
 f
(1 row)
```

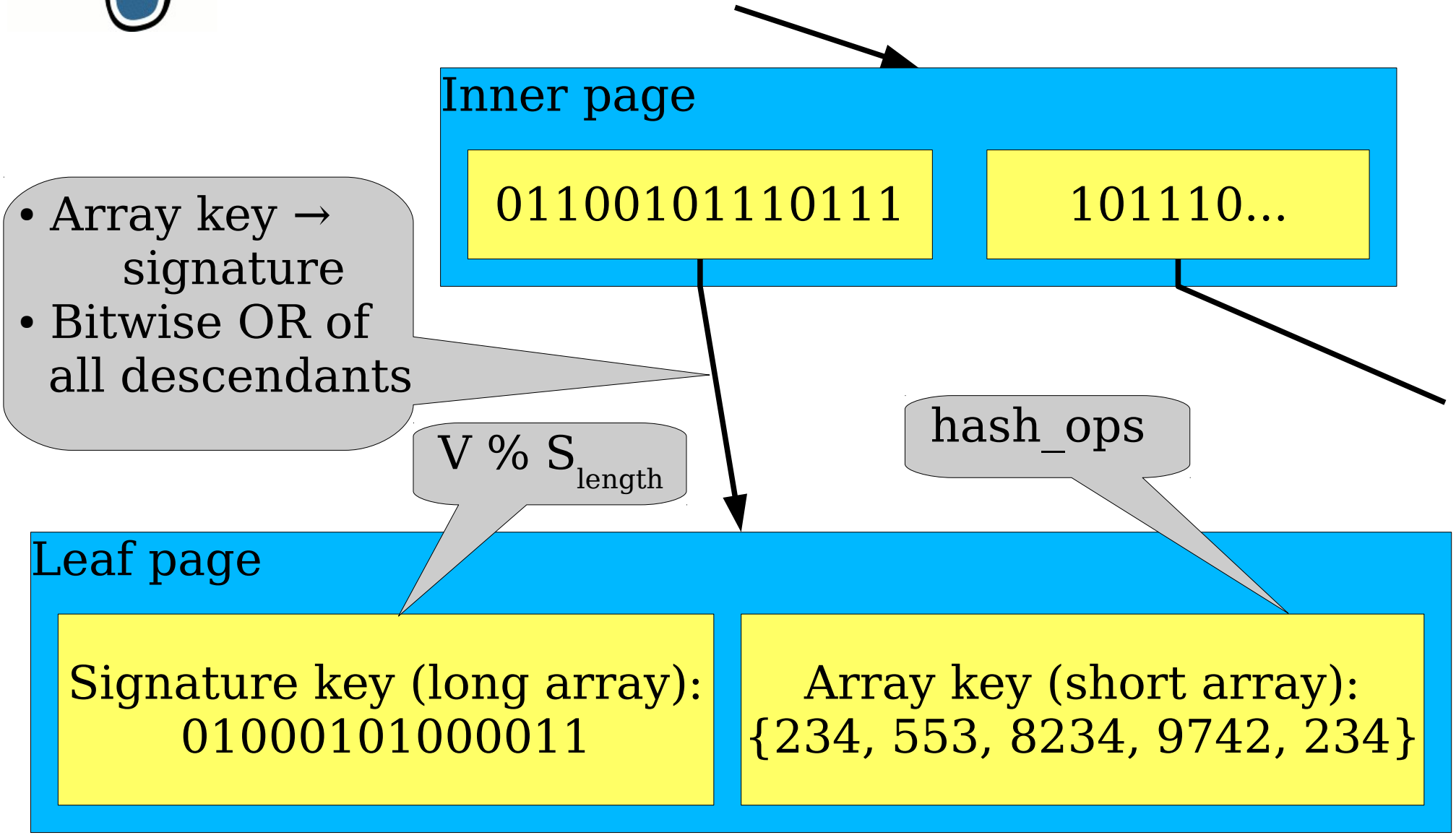


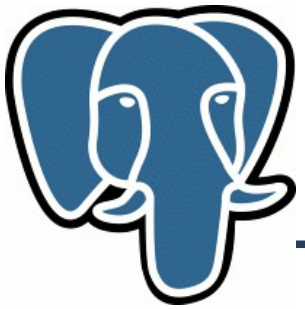
Индексная поддержка

```
Speedup anyarray % anyarray
```

- Btree, hash – not applicable
- GiST – Generalized Search Tree
- GIN - Generalized Inverted Index

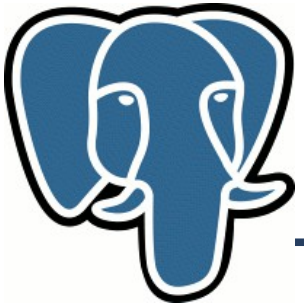
GiST index


Inner page
01100101110111
101110...
• Array key → signature
• Bitwise OR of all descendants
V % S length
hash_ops
Leaf page
Signature key (long array): 0100010100011
Array key (short array): {234, 553, 8234, 9742, 234}



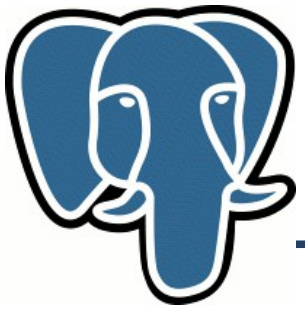
Anyarray: smlar

- float4 smlar(anyarray, anyarray, text Formula)
- text[] tsvector2textarray(tsvector)



Рекомендационная система

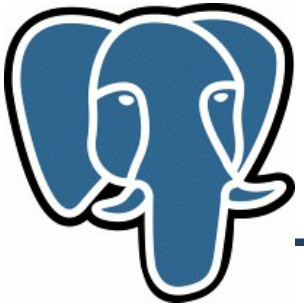
- item-item CF (стабильность и независимость)
  - Метрика: cosine
- Данные MovieLens
  - 1mln rates: 6000 users on 4000 movies
  - 10 mln rates: 72000 users on 10,000 movies



Рекомендационная система

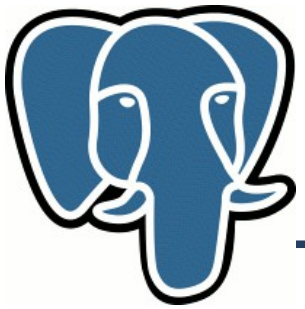
- Входные данные:
  - movies(mid,title,genre,description)
  - rates(uid,mid,rate)
- Шаг 1: Делаем простые лайки из оценки
  u: r=1 if r>avg(rate)
  rates(uid,mid,like)
- Результирующая табличка

  ihu(itemid,{users}, {rates})



Рекомендационная система

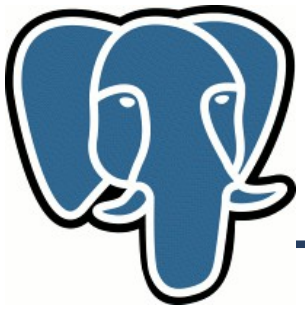
- Шаг 2. item-item матрица
- Предвычисляем item-item матрицу ii(itemid1,itemid2, sml) from ihu table
- Шаг 3. Собственно, результат
  - Q1: для заданного кино найти наиболее похожие
  - Q2: для заданного пользователя найти рекомендации



Рекомендационная система

- Шаг 1.
  - Produce table ihu (itemid,{users})
  - Create index to accelerate % operation

```
CREATE INDEX ihu_users_itemid_idx ON ihu
USING gist (users _int4_sml_ops, itemid);
```



Шаг 2. Item-Item

```
SELECT
  r1.itemid as itemid1,
  r2.itemid as itemid2,
  smlar(r1.users,r2.users) as sml
INTO ii
FROM
  ihu AS r1,
  ihu AS r2
WHERE
  r1.users % r2.users AND
  r1.itemid > r2.itemid;
```

Smlar.threshold=0.2
SELECT 209657

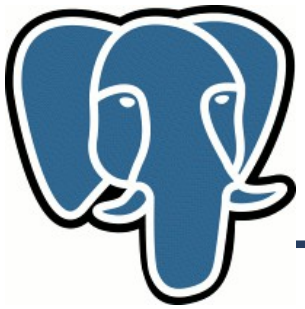
Index	no-index
526195 ms	1436433

Speedup 2.7

Smlar.threshold=0.4
SELECT 8955

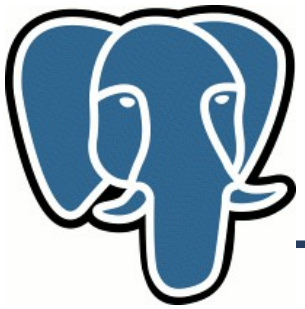
Index	no-index
253378 ms	1172432

Speedup 4.6



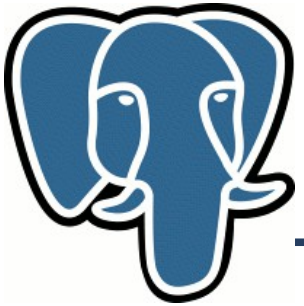
Шаг 2. Item-Item

```
CREATE INDEX ii_itemid1_idx on ii(itemid1);
CREATE INDEX ii_itemid2_idx on ii(itemid2);

CREATE OR REPLACE VIEW  ii_view AS
SELECT itemid1, itemid2, sml FROM ii
UNION ALL
SELECT itemid2, itemid1, sml  FROM ii;
```



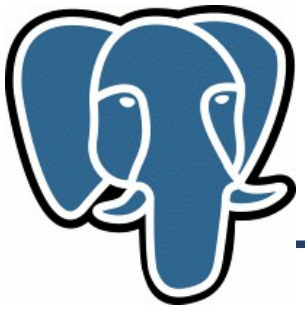
Шаг 3. Результат

```
CREATE OR REPLACE FUNCTION smlmovies(
   movie_id integer,num_movies integer,
   itemid OUT integer, sml OUT float, title OUT text)
RETURNS SETOF RECORD AS $$
SELECT s.itemid,s.sml::float, m.title
FROM movies m,
     ( SELECT itemid2 AS itemid, sml FROM ii_view
       WHERE itemid1 = movie_id
       UNION ALL
       SELECT movie_id, 1 -- just to illustration
     ) AS s
WHERE
      m.mid=s.itemid
GROUP BY s.itemid, rates, s.sml, m.title
ORDER BY s.sml DESC
LIMIT num_movies;
$$ LANGUAGE SQL IMMUTABLE;
```



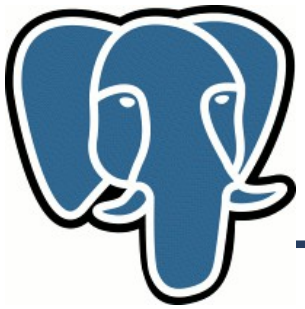
Шаг 3. Результат

```
=# select itemid, sml,title from smlmovies(1104,10);
 itemid |        sml         |                  title
--------+--------------------+-----------------------------------
   1104 |                  1 | Streetcar Named Desire, A (1951)
   1945 | 0.436752468347549  | On the Waterfront (1954)
   1952 | 0.397110104560852  | Midnight Cowboy (1969)
   1207 | 0.392107665538788  | To Kill a Mockingbird (1962)
   1247 | 0.387987941503525  | Graduate, The (1967)
   2132 | 0.384177327156067  | Who's Afraid of Virginia Woolf? (1966)
    923 | 0.381125450134277  | Citizen Kane (1941)
    926 | 0.377328515052795  | All About Eve (1950)
   1103 | 0.363485038280487  | Rebel Without a Cause (1955)
   1084 | 0.356647849082947  | Bonnie and Clyde (1967)
(10 rows)

Time: 5.780 ms
```



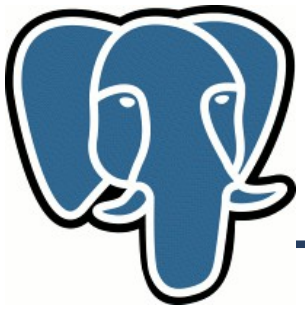
Шаг 3. Результат

```
# select itemid, sml,title from smlmovies(364,10);
 itemid |        sml         |                  title
--------+--------------------+-----------------------------------
    364 |                  1 | Lion King, The (1994)
    595 | 0.556357622146606 | Beauty and the Beast (1991)
    588 | 0.547775387763977 | Aladdin (1992)
      1 | 0.472894549369812 | Toy Story (1995)
   2081 |   0.4552321434021 | Little Mermaid, The (1989)
   1907 | 0.442262977361679 | Mulan (1998)
   1022 |  0.415279328823 09 | Cinderella (1950)
    594 | 0.407131761312485 | Snow White and the Seven Dwarfs (1937)
   2355 | 0.405456274747849 | Bug's Life, A (1998)
   2078 | 0.389742106199265 | Jungle Book, The (1967)
(10 rows)
```



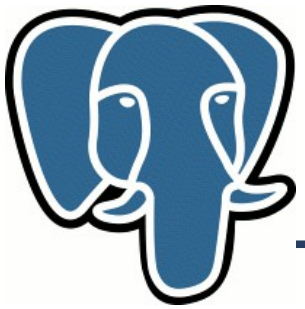
Шаг 3. Результат

```
=# select itemid, sml,title from smlmovies(919,10);
 itemid |        sml         |                              title
--------+--------------------+-----------------------------------
    919 |                  1 | Wizard of Oz, The (1939)
    260 | 0.495729923248291 | Star Wars: Episode IV - A New Hope (197
    912 | 0.48350244760513 | Casablanca (1942)
   1198 | 0.48167577385902 | Raiders of the Lost Ark (1981)
   1196 | 0.46829551458358 | Star Wars: Episode V - The Empire Strik
   1028 | 0.46054756641387 | Mary Poppins (1964)
   1097 | 0.45598563551902 | E.T. the Extra-Terrestrial (1982)
   1247 | 0.44949394464492 | Graduate, The (1967)
    858 | 0.44678425788879 | Godfather, The (1972)
    594 |  0.4467646181583 | Snow White and the Seven Dwarfs (1937)
(10 rows)

Time: 10.207 ms
```



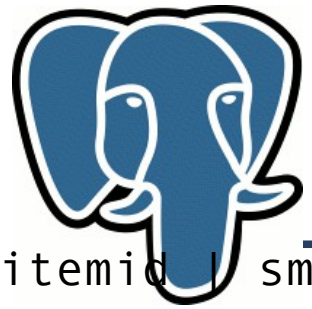
Мой профиль

```
CREATE TABLE myprofile (mid integer);
INSERT INTO myprofile VALUES
   (912),(1961),(1210),(1291),(3148),(356),(919),(2943),(362),(2116);

=# select p.mid, m.title from movies m, myprofile p where m.mid=p.mid;
 mid  |                          title
------+-----------------------------------------------
  912 | Casablanca (1942)
 1961 | Rain Man (1988)
 1210 | Star Wars: Episode VI - Return of the Jedi (1983)
 1291 | Indiana Jones and the Last Crusade (1989)
 3148 | Cider House Rules, The (1999)
  356 | Forrest Gump (1994)
  919 | Wizard of Oz, The (1939)
 2943 | Indochine (1992)
  362 | Jungle Book, The (1994)
 2116 | Lord of the Rings, The (1978)
(10 rows)
```



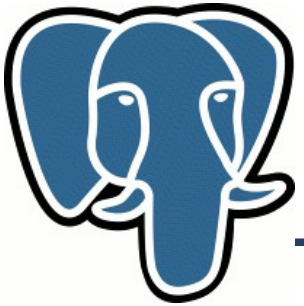
Порекумендуйте мне

```
SELECT t.itemid2 as itemid, t.sml::float, m.title
FROM movies m,
(
   WITH usermovies AS (
              SELECT mid  FROM myprofile
   ),
              mrec AS (
              SELECT itemid2, sml
              FROM ii_view ii, usermovies um
              WHERE
                    ii.itemid1=um.mid  AND
                    ii.itemid2 NOT IN ( SELECT *  FROM usermovies)
              ORDER BY itemid2 ASC
   )
   SELECT itemid2, sml,  rank()
   OVER (PARTITION BY itemid2 ORDER BY sml DESC) FROM mrec
) t
WHERE t.itemid2=m.mid AND t.rank = 1
ORDER BY t.sml DESC
LIMIT 10;
```



Пожалуйста!

```
 itemid | sml  |                          title
--------+------+-------------------------------------------------------
   1196 | 0.71 | Star Wars: Episode V - The Empire Strikes Back (1980)
    260 | 0.67 | Star Wars: Episode IV - A New Hope (1977)
   1198 | 0.67 | Raiders of the Lost Ark (1981)
   1036 | 0.58 | Die Hard (1988)
   2571 | 0.57 | Matrix, The (1999)
   1240 | 0.56 | Terminator, The (1984)
   2115 | 0.56 | Indiana Jones and the Temple of Doom (1984)
    589 | 0.54 | Terminator 2: Judgment Day (1991)
    592 | 0.54 | Batman (1989)
    923 | 0.53 | Citizen Kane (1941)
   1270 | 0.53 | Back to the Future (1985)
   1197 | 0.52 | Princess Bride, The (1987)
    480 | 0.51 | Jurassic Park (1993)
   1200 | 0.51 | Aliens (1986)
    457 | 0.51 | Fugitive, The (1993)
   1374 | 0.50 | Star Trek: The Wrath of Khan (1982)
   2000 | 0.50 | Lethal Weapon (1987)
   2628 | 0.50 | Star Wars: Episode I - The Phantom Menace (1999)
   2028 | 0.49 | Saving Private Ryan (1998)
   1610 | 0.49 | Hunt for Red October, The (1990)
(20 rows)

Time: 207.065 ms
```

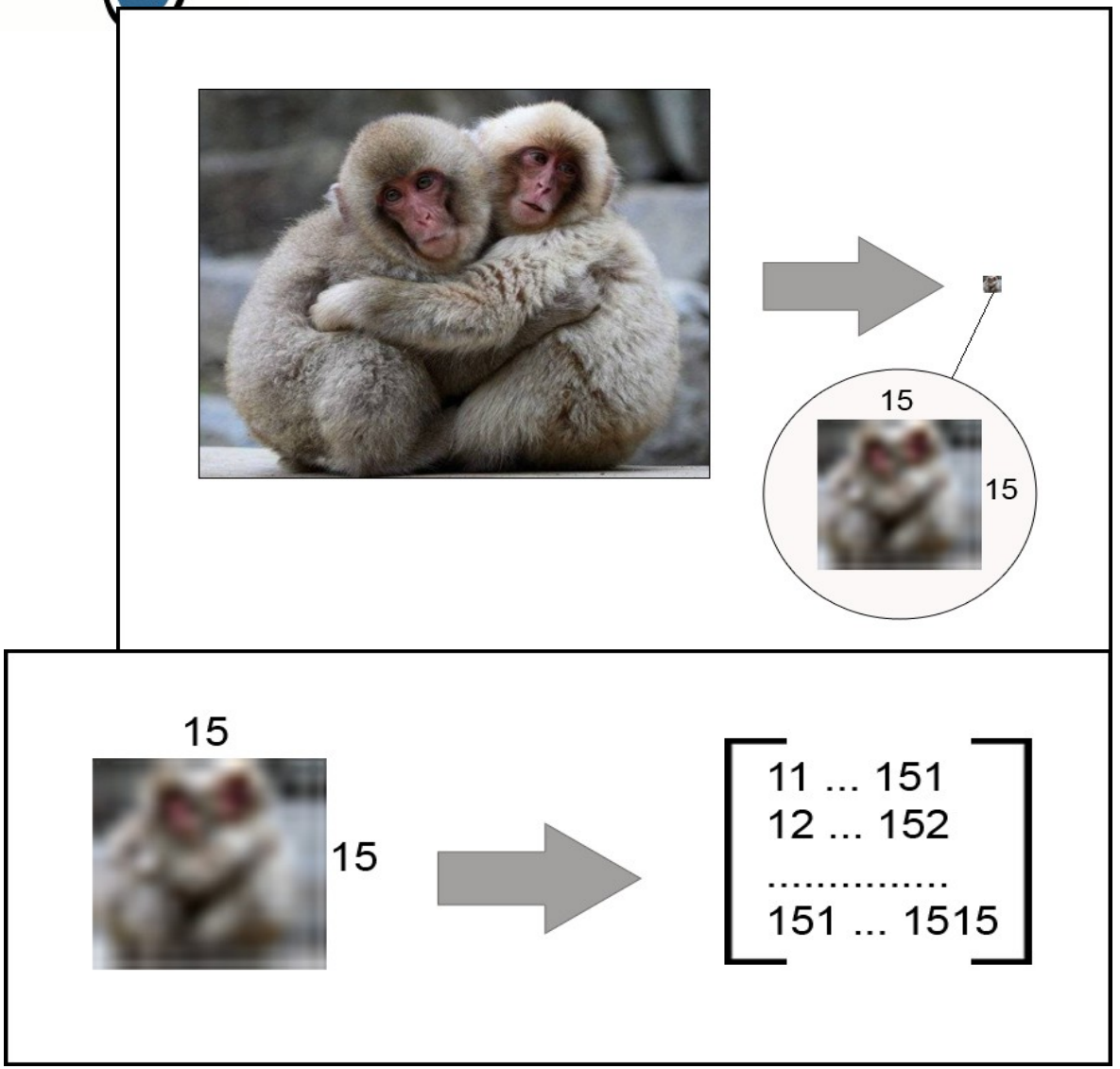


Рекомендационная система

- Очень простая система!
- Работает!
- Периодический пересчет

  (10 mln лайков <10 минут на МакБуке)
- Нужно чистить от спама, эскперты в кино не являются экпертами в музыке.

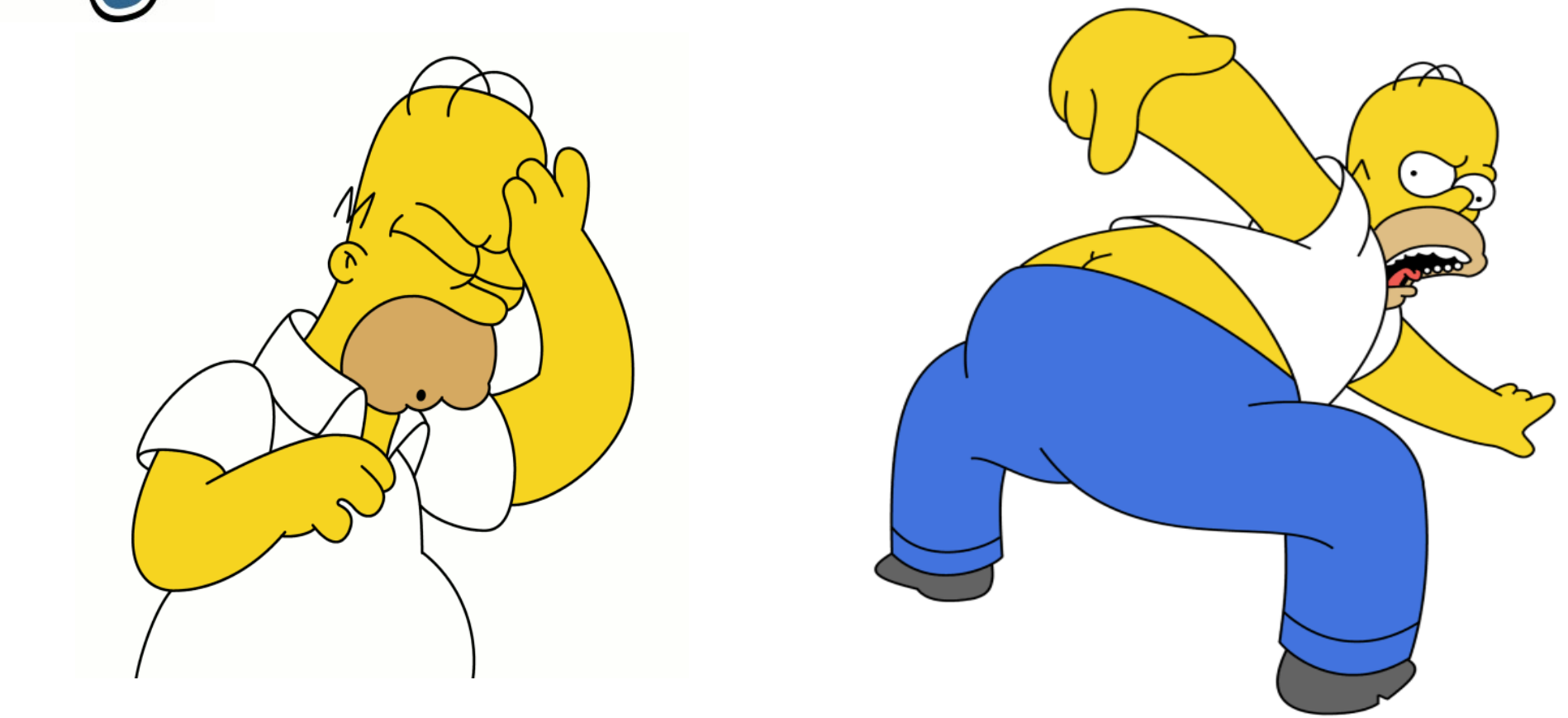
Content-based similarity



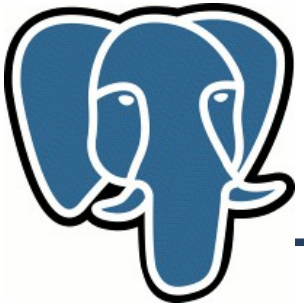
For each image
{
  1. Scale -> 15x15
  2. Array of intensities
}

smlar(arr1,arr2)

Content-based similarity



23.56% similarity



Anyarray

- Intarray
- git clone git://sigaev.ru/smlar.git
- Anyarray... in progress

Спасибо!