Googling in PostgreSQL

FUDO SECURITY

Adam Wołk

a.wolk@fudosecurity.com

awolk@openbsd.org

https://blog.tintagel.pl

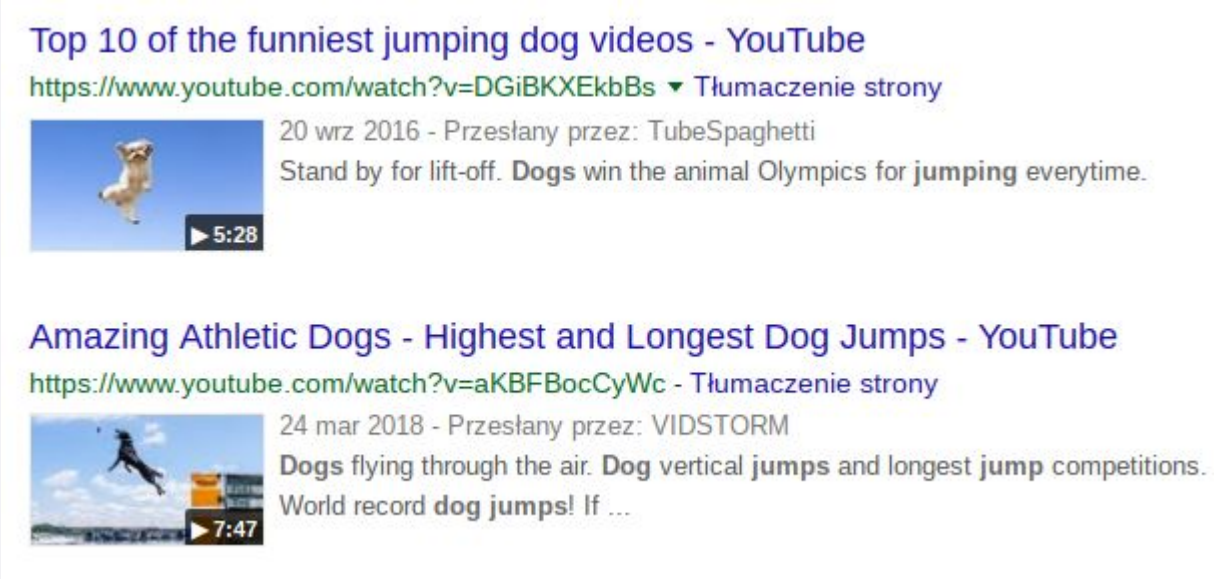
@mulander

Wyszukiwanie pełnotekstowe

Wyszukiwanie ~~pełnotekstowe~~ dokumentów

Wyszukiwanie ~~pełnotekstowe~~ dokumentów

System pozwalający na wyszukiwanie dokumentów, z uwzględnieniem jego całej zawartości.



Dokument?

Dokument

“A document is the unit of searching in a full text search system; for example, a magazine article or email message. The text search engine must be able to parse documents and store associations of lexemes (key words) with their parent document. Later, these associations are used to search for documents that contain query words.”

Source: PostgreSQL Documentation
https://www.postgresql.org/docs/9.3/textsearch-intro.html#TEXTSEARCH-DOCUMENT

Dokument

- Jest elementem ulotnym, musi istnieć w trakcie indeksowania ale nie musimy go przechowywać w całości - tylko wynik jego parsowania i odniesienie do elementów pozwalających go zlokalizować/odtworzyć. Przykładowo, indeksując maile możemy uwzględnić również treść stron z odnośników.
- Sekcje dokumentu można ważyć przypisując im osobne rangi od A do D. Wpływa to na kolejność w liście wyników wyszukiwania. Przykładowo dopasowanie w tytule maila jest ważniejsze niż wystąpienie słowa na podlinkowanej stronie.

Dokument - email

- Przykładowy email z listy misc@openbsd.org
- 60 linii nagłówków (headers)
  - 59 mało przydatne w wyszukiwaniu pełnotekstowym, lepiej stosować jako regularne kolumny np. **Date**
  - **Subject** jedyny użyteczny w dokumencie do indeksowania

Subject: awk regex bug

Hi misc,

I'm running a 5.7 release, and I'm wondering if anyone can confirm an awk bug I found.

Curly brackets are treated as literal characters instead of bounds as specified by re_format(7).

Reproduction:

```
echo aa | awk '/a{2}/'
```

produces no output instead of printing "aa" as expected.

```
echo 'a{2}' | awk '/a{2}/'
```

produces output when none is expected.

This bug seems awk specific since the equivalents using grep

```
echo aa | grep -E 'a{2}'

echo 'a{2}' | grep -E 'a{2}'
```

work as expected.

Parsowanie dokumentu

Parsowanie

- Podczas parsowania, zachowujemy jedynie wysoce selektywne elementy - przykładowo w książce o PostgreSQL sama nazwa bazy jest mało selektywna ponieważ najprawdopodobniej występuje na każdej stronie zatem wyszukiwania z jej użyciem są mało przydatne. PostgreSQL stosuje w tym celu stop listy (**stop words**) - np. dla języka Angielskiego pomijamy **a** oraz **the**.
- Normalizujemy tokenizowany tekst, lingwistycznie (**stemming**), sprowadzając liczby do jednolitego formatu/precyzji itd.

```
select * from to_tsquery('english', EMAIL);
'/a':45,60
'2':46,58,61,83,86,90
'5.7':10
'7':40
'aa':43,53,79
'anyon':17
'awk':1,21,44,59,71
'bound':34
'bracket':26
'bug':3,22,69
'charact':31
'confirm':19
'cur':25
'e':81,88
'echo':42,56,78,84
'equival':75
'expect':55,67,93
'format':39
'found':24
'grep':77,80,87 'hi':4 'instead':32,50
'liter':30 'm':7,14 'misc':5 'none':65
'output':49,63 'print':52 'produc':47,62 're':38
'regex':2 'releas':11 'reproduct':41 'run':8
'seem':70 'sinc'
:73 'specif':72 'specifi':36 'treat':28 'use':76
'wonder':15 'work':91
```

```
Subject: awk regex bug

Hi misc,

I'm running a 5.7 release, and I'm wondering if anyone can confirm
an awk bug I found.

Curly brackets are treated as literal characters instead of bounds
as specified by re_format(7).

Reproduction:

echo aa | awk '/a{2}/'

produces no output instead of printing "aa" as expected.

echo 'a{2}' | awk '/a{2}/'

produces output when none is expected.

This bug seems awk specific since the equivalents using grep

echo aa | grep -E 'a{2}'

echo 'a{2}' | grep -E 'a{2}'

work as expected.
```

```
select * from ts_debug('english',EMAIL);
   alias   |    description    |    token     |  dictionaries  |  dictionary  |   lexemes
-----------+-------------------+--------------+----------------+--------------+-------------
 asciiword | Word, all ASCII   | awk          | {english_stem} | english_stem | {awk}
 blank     | Space symbols     |              | {}             |              |
 asciiword | Word, all ASCII   | regex        | {english_stem} | english_stem | {regex}
 blank     | Space symbols     |              | {}             |              |
 asciiword | Word, all ASCII   | bug          | {english_stem} | english_stem | {bug}
 blank     | Space symbols     |             +| {}             |              |
           |                   |             +|                |              |
           |                   |              |                |              |
 asciiword | Word, all ASCII   | Hi           | {english_stem} | english_stem | {hi}
 blank     | Space symbols     |              | {}             |              |
 asciiword | Word, all ASCII   | misc         | {english_stem} | english_stem | {misc}
 blank     | Space symbols     | ,           +| {}             |              |
           |                   |             +|                |              |
           |                   |              |                |              |
 asciiword | Word, all ASCII   | I            | {english_stem} | english_stem | {}
 blank     | Space symbols     | '            | {}             |              |
 asciiword | Word, all ASCII   | m            | {english_stem} | english_stem | {m}
 blank     | Space symbols     |              | {}             |              |
 asciiword | Word, all ASCII   | running      | {english_stem} | english_stem | {run}
 blank     | Space symbols     |              | {}             |              |
 asciiword | Word, all ASCII   | a            | {english_stem} | english_stem | {}
 blank     | Space symbols     |              | {}             |              |
 float     | Decimal notation  | 5.7          | {simple}       | simple       | {5.7}
 blank     | Space symbols     |              | {}             |              |
 asciiword | Word, all ASCII   | release      | {english_stem} | english_stem | {releas}
```

alias	description	token	dictionaries	dictionary	lexemes
asciiword	Word, all ASCII	echo	{english_stem}	english_stem	{echo}
blank	Space symbols		{}		
asciiword	Word, all ASCII	aa	{english_stem}	english_stem	{aa}
blank	Space symbols	|	{}		
asciiword	Word, all ASCII	awk	{english_stem}	english_stem	{awk}
blank	Space symbols	'	{}		
file	**File or path name**	**/a**	**{simple}**	**simple**	**{/a}**
blank	Space symbols	{	{}		
uint	Unsigned integer	2	{simple}	simple	{2}
blank	Space symbols	}	{}		
blank	Space symbols	/'	{}		

Parsowanie

- Wprowadzamy obsługę synonimów, odpowiednich dla naszej dziedziny. Jeżeli ktoś, stara się znaleźć prezentację na temat bazy PostgreSQL to nasz parser dokumentów powinien posiadać synonimy: **postgresql, postgres, pgsql**
- Definiujemy **Thesaurus** w celu normalizacji leksemów z uwzględnieniem relacji między nimi. Przykładowo:
  - **Owczarek Australijski**
  - **Owczarek Australijski typ Amerykański**
  - **Aussie**

Synonimy?

- awk : gawk
- grep : egrep

Thesaurus?

- 5.7 release
- release 5.7
- OpenBSD 5.7
- 5.7 OpenBSD
- 5.7

```
Subject: awk regex bug

Hi misc,

I'm running a 5.7 release, and I'm wondering if anyone can confirm
an awk bug I found.

Curly brackets are treated as literal characters instead of bounds
as specified by re_format(7).

Reproduction:

echo aa | awk '/a{2}/'

produces no output instead of printing "aa" as expected.

echo 'a{2}' | awk '/a{2}/'

produces output when none is expected.

This bug seems awk specific since the equivalents using grep

echo aa | grep -E 'a{2}'

echo 'a{2}' | grep -E 'a{2}'

work as expected.
```

Wyszukiwanie email

Nasz korpus danych

misc@openbsd.org
198M Maildir
26,758 maili
307 multipart

Ignorujemy maile multipart

Zakładamy kodowanie UTF-8,
odrzucimy maile z innym (497).

Indeksujemy
- Tytuł
- Treść maila

Przechowujemy je dodatkowo w bazie tylko dla wygody

```
Subject: awk regex bug

Hi misc,

I'm running a 5.7 release, and I'm wondering if anyone can confirm
an awk bug I found.

Curly brackets are treated as literal characters instead of bounds
as specified by re_format(7).

Reproduction:

echo aa | awk '/a{2}/'

produces no output instead of printing "aa" as expected.

echo 'a{2}' | awk '/a{2}/'

produces output when none is expected.

This bug seems awk specific since the equivalents using grep

echo aa | grep -E 'a{2}'

echo 'a{2}' | grep -E 'a{2}'

work as expected.
```

Struktura w bazie

- file_path TEXT
- subject TEXT
- body TEXT
- document tsvector

Dokument

- **Subject** rank **A**
- **Body** rank **B**

Indeks

- **GIN** na tsvector

```
Subject: awk regex bug

Hi misc,

I'm running a 5.7 release, and I'm wondering if anyone can confirm
an awk bug I found.

Curly brackets are treated as literal characters instead of bounds
as specified by re_format(7).

Reproduction:

echo aa | awk '/a{2}/'

produces no output instead of printing "aa" as expected.

echo 'a{2}' | awk '/a{2}/'

produces output when none is expected.

This bug seems awk specific since the equivalents using grep

echo aa | grep -E 'a{2}'

echo 'a{2}' | grep -E 'a{2}'

work as expected.
```

Prawie minimalna konfiguracja:

- Tworzymy tabelę
- Używamy parsera ‘english’
- Przypisujemy wagi
  - A - tytuł
  - B - zawartość

table.sql:

```
CREATE TABLE emails(
        id bigserial,
        file_path text,
        subject text,
        body text,
        document tsvector
);
```

load.py: https://gist.github.com/mulander/512775783640a3ac0d3404c130836461

```
index_email = """
    INSERT INTO
         emails(file_path, subject   , body,
                document)
         VALUES(%(path)s, %(subject)s, %(body)s  ,
                setweight(to_tsvector('english', %(subject)s), 'A') ||
                setweight(to_tsvector('english', %(body)s), 'B'))
                ;
"""
```

```
postgres=# \dt+ public.emails
                     List of relations
 Schema |  Name  | Type  |  Owner   |  Size  | Description
--------+--------+-------+----------+--------+-------------
 public | emails | table | postgres | 101 MB |
(1 row)
```

`websearch_to_tsquery`

Zamienia wyszukiwanie z formatu web na query tsearch Postgres

WHERE document @@ query

```
postgres=# SELECT websearch_to_tsquery('english', 'awk regex bug');
  websearch_to_tsquery
-------------------------
 'awk' & 'regex' & 'bug'
(1 row)

postgres=# SELECT websearch_to_tsquery('english', 'awk "regex bug"');
   websearch_to_tsquery
---------------------------
 'awk' & 'regex' <-> 'bug'
(1 row)

postgres=# SELECT websearch_to_tsquery('english', 'awk or regex');
 websearch_to_tsquery
----------------------
 'awk' | 'regex'
(1 row)
```

```
postgres=# SELECT subject,
       ts_rank(document, websearch_to_tsquery('english', 'awk bug'))
from emails where document @@ websearch_to_tsquery('english', 'awk bug')
order by ts_rank(document, websearch_to_tsquery('english', 'awk bug')) desc;
                            subject                                |   ts_rank
-------------------------------------------------------------------+-------------
 awk regex bug                                                     |    0.995528
 Re: awk regex bug                                                 |    0.995412
 Re: awk regex bug                                                 |    0.987844
 Re: awk regex bug                                                 |    0.985063
 Flex / Regexps (Re: awk regex bug)                                |    0.985009
 Re: Flex / Regexps (Re: awk regex bug)                            |    0.985009
 Re: FAQ14: Growing disk partitions: fdisk                         | 1.03356e-07
 Re: awk in OpenBSD                                                | 1.58471e-08
 OpenBSD 5.8 released                                              | 3.55271e-15
 OpenBSD 6.0 released, September 1, 2016                           | 2.22045e-15
 Re: OpenBSD 6.0 released, September 1, 2016                       | 2.22045e-15
 Re: sh /etc/netstart interface counter intuitive behaviour with+|       4e-16
  multiple inet aliases 6.4 and 6.3                                |
 simple DNS lookup utility                                         |       4e-16
 Re: sh /etc/netstart interface counter intuitive behaviour with+|       4e-16
  multiple inet aliases 6.4 and 6.3                                |
(14 rows)
```

```
                                                        QUERY PLAN
-----------------------------------------------------------------------------------------------------
 Sort  (cost=4161.45..4161.48 rows=10 width=46) (actual time=227.466..227.482 rows=14 loops=1)
   Output: subject, (ts_rank(document, '''awk'' & ''bug'''::tsquery))
   Sort Key: (ts_rank(emails.document, '''awk'' & ''bug'''::tsquery)) DESC
   Sort Method: quicksort  Memory: 26kB
   Buffers: shared hit=49365
   ->  Seq Scan on public.emails  (cost=0.00..4161.29 rows=10 width=46) (actual time=7.627..227.358 rows=14 loops=1)
         Output: subject, ts_rank(document, '''awk'' & ''bug'''::tsquery)
         Filter: (emails.document @@ '''awk'' & ''bug'''::tsquery)
         Rows Removed by Filter: 26247
         Buffers: shared hit=49365
 Planning Time: 0.510 ms
 Execution Time: 227.576 ms
(12 rows)
```

Dodajemy index!

```
CREATE INDEX emails_idx ON emails USING gin(document);

postgres=# \dti+ emails_idx
                           List of relations
 Schema |    Name    | Type  |  Owner   | Table  | Size  | Description
--------+------------+-------+----------+--------+-------+-------------
 public | emails_idx | index | postgres | emails | 27 MB |
(1 row)
```

```
                                                                QUERY PLAN
--------------------------------------------------------------------------------------------------------------------------------
----
 Sort  (cost=66.86..66.89 rows=10 width=46) (actual time=1.518..1.576 rows=14 loops=1)
   Output: subject, (ts_rank(document, '''awk'' & ''bug'''::tsquery))
   Sort Key: (ts_rank(emails.document, '''awk'' & ''bug'''::tsquery)) DESC
   Sort Method: quicksort  Memory: 26kB
   Buffers: shared hit=62
   ->  Bitmap Heap Scan on public.emails  (cost=28.08..66.70 rows=10 width=46) (actual time=0.264..1.386 rows=14
loops=1)
         Output: subject, ts_rank(document, '''awk'' & ''bug'''::tsquery)
         Recheck Cond: (emails.document @@ '''awk'' & ''bug'''::tsquery)
         Heap Blocks: exact=11
         Buffers: shared hit=62
         ->  Bitmap Index Scan on emails_idx  (cost=0.00..28.08 rows=10 width=0) (actual time=0.208..0.211 rows=14
loops=1)
               Index Cond: (emails.document @@ '''awk'' & ''bug'''::tsquery)
               Buffers: shared hit=8
 Planning Time: 0.618 ms
 Execution Time: 1.759 ms
(15 rows)
```

Co dalej?

Pobierz kod
https://github.com/mulander/googling-in-postgresql

Otwórz dokumentację PostgreSQL
https://www.postgresql.org/docs/11/textsearch.html

Pobierz korpus maili
https://junk.tintagel.pl/talks/misc.tar.gz

```
tar zxvf misc.tar.gz
make
```

Co dalej?

Spróbuj swoich sił:

Wyświetl email z podświetlonym dopasowaniem przy użyciu **ts_headline**

Synonim awk - gawk

Thesaurus OpenBSD 5.7 i Release 5.7

Wyszukiwanie pomimo literówek przy użyciu **pg_trgm**

ts_rank dopasowany do domeny problemu

Indeksowanie linkowanych dokumentów

Warszawa
Aleje Jerozolimskie 178

JOIN FUDO SQUAD

OUR TECH POSITIONS

- QA ENGINEER
- C DEVELOPER
- PYTHON DEVELOPER
- JAVASCRIPT DEVELOPER
- TECH SUPPORT ENGINEER

FUDO STRENGTHS

- PRODUCT UNIQUENESS & QUALITY FOCUS
- CONSTANT SEARCH FOR NEW POSSIBILITIES
- WORKING (ALMOST :)) ALL TOGETHER
- OPENNESS FOR EMPLOYEE FEEDBACK
- TEAM!

"Everything is possible with those developers!"

hr@fudosecurity.com

FUDO SECURITY