Полнотекстовый поиск от A до Ω

Postgres PROFESSIONAL

На чем мы можем поупражняться

На Postgres Pro Standard

- В нём есть много полезного для поиска
- Её можно поставить рядом с PostgreSQL, не мешая ему

Откда взять

https://postgrespro.ru/products/postgrespro/download/10.1.1

Транскрипт https://pgconf.ru/media/2018/ts.txt

Подключить репозиторий
Ставится в /opt/pgpro

```
sudo apt -y install postgrespro-std-10    #  выносит ванильный

sudo apt -y install postgrespro-std-10-server  #  ставит в /opt/
sudo apt -y install postgrespro-std-10-client
```

В конфиг

```
shared_preload_libraries = 'shared_ispell'
shared_ispell.max_size = 64MB

# если ставим рядом, поменять порт, например на 5400
port=5400
```

Тестовые данные

https://pgconf.ru/media/2018/doc10.sql.gz

\d doc_page

```
                        Таблица "public.doc_page"
 Столбец |   Тип   | Правило сортировки | Допустимость NULL |
---------+---------+--------------------+-------------------+--------
 id      | integer |                    | not null          |
 lang    | integer |                    |                   | 2
 uri     | text    |                    |                   |
 title   | text    |                    |                   |
 text    | text    |                    |                   |
```

1-русский
2-английский

Создание конфигурации

Конфигурация поиска = парсер + набор словарей.

https://postgrespro.ru/docs/postgrespro/10/sql-createtsconfig

```
\dF[+]  [PATTERN]      list text search configurations
\dFd[+] [PATTERN]      list text search dictionaries
\dFp[+] [PATTERN]      list text search parsers
\dFt[+] [PATTERN]      list text search templates
```

\dF

```
test=# \dF
                  Список конфигураций текстового поиска
   Схема    |    Имя     |                   Описание
------------+------------+-------------------------------------
 pg_catalog | danish     | configuration for danish language
…….
 pg_catalog | russian    | configuration for russian language
 pg_catalog | simple     | simple configuration
 pg_catalog | turkish    | configuration for turkish language
(16 строк)

\dF+ russian
Конфигурация текстового поиска "pg_catalog.russian"
Анализатор: "pg_catalog.default"
    Фрагмент      |   Словари
------------------+--------------
 asciihword       | english_stem
 asciiword        | english_stem
 email            | simple
 file             | simple
 float            | simple
 host             | simple
 hword            | russian_stem
 hword_asciipart  | english_stem
 ……
```

Готовим словари

Shared Ispell – словарь, хранящийся в shared-памяти, а не в каждом отдельном процессе

https://postgrespro.ru/docs/postgrespro/10/shared-ispell

```
CREATE EXTENSION shared_ispell;

CREATE TEXT SEARCH DICTIONARY dic_shared_en (
    TEMPLATE = shared_ispell,
    DictFile = en_us,
    AffFile = en_us
);

CREATE TEXT SEARCH DICTIONARY dic_shared_ru (
    TEMPLATE = shared_ispell,
    DictFile = ru_ru,
    AffFile = ru_ru
);
```

Словари синонимов

Синонимы определены в файле

```
/opt/pgpro/std-10/share/tsearch_data/russian_syn.syn
```

```
CREATE TEXT SEARCH DICTIONARY russian_synonym (
    TEMPLATE = synonym,
    SYNONYMS = russian_syn
);

CREATE TEXT SEARCH DICTIONARY english_synonym (
    TEMPLATE = synonym,
    SYNONYMS = english_syn
);
```

Парсер - основа всего

```
CREATE EXTENSION pg_tsparser;
CREATE TEXT SEARCH CONFIGURATION ts (
    PARSER = tsparser
);
```

Наполняем конфигурацию

См https://postgrespro.com/docs/postgrespro/10/textsearch-parsers

```
ALTER TEXT SEARCH CONFIGURATION ts
    ADD MAPPING FOR asciiword, word, hword_part,
            hword_asciipart, asciihword, hword
    WITH russian_synonym, english_synonym,
             dic_shared_ru, dic_shared_en, russian_stem;

ALTER TEXT SEARCH CONFIGURATION ts
    ADD MAPPING FOR numword, numhword, email, url,
            host, sfloat, version, hword_numpart, url_path,
                    file, "float", "int", uint
    WITH simple;
```

Для английского:

```
CREATE TEXT SEARCH CONFIGURATION ts_en (
    PARSER = tsparser
);

ALTER TEXT SEARCH CONFIGURATION ts_en
   ADD MAPPING FOR asciiword, word, hword_part,
          hword_asciipart, asciihword, hword
   WITH russian_synonym, english_synonym,
           dic_shared_ru, dic_shared_en, english_stem;

ALTER TEXT SEARCH CONFIGURATION ts
   ADD MAPPING FOR numword, numhword, email, url,
          host, sfloat, version, hword_numpart, url_path,
                  file, "float", "int", uint
   WITH simple;
```

Функция создания tsvector

```
CREATE FUNCTION compose_tsvector(
        config regconfig,      title text,       text text
)
  RETURNS tsvector LANGUAGE sql AS $$
      SELECT
        setweight(to_tsvector(config, coalesce(title, '')),'A') ||
        setweight(to_tsvector(config, coalesce(text,  '')),'D');
  $$;
```

Поле tsvector и работа с ним

```
ALTER TABLE doc_page ADD tsvector tsvector;

CREATE FUNCTION doc_page_make_tsvector() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
BEGIN
  NEW.tsvector    := compose_tsvector(
     CASE WHEN NEW.lang = 1 THEN 'ts'::regconfig
                                    ELSE 'ts_en'::regconfig
     END,
     NEW.title, NEW.text);
  RETURN NEW;
END;
$$;
```

Поле tsvector и работа с ним

```
ALTER TABLE doc_page ADD tsvector tsvector;

CREATE FUNCTION doc_page_t () RETURNS trigger
    LANGUAGE plpgsql
    AS $$
BEGIN
  NEW.tsvector    := compose_tsvector(
     CASE WHEN NEW.lang = 1 THEN 'ts'::regconfig
                                    ELSE 'ts_en'::regconfig
     END,
     NEW.title, NEW.text);
  RETURN NEW;
END;
$$;
CREATE TRIGGER doc_page_ts BEFORE INSERT ON doc_page
     FOR EACH ROW EXECUTE PROCEDURE  doc_page_t();
```

Заполним tsvector

```
UPDATE doc_page set tsvector = compose_tsvector(
   CASE WHEN lang = 1
      THEN 'ts'::regconfig
      ELSE 'ts_en'::regconfig END,
    title,
    text);
```

Поиск!

```
WITH q AS (SELECT plainto_tsquery('ts', 'Конфигурация') AS q)

  SELECT title,
     ts_headline('ts', title,  q.q) s1,
     ts_headline('ts', text,  q.q) s2
  FROM doc_page, q
  WHERE tsvector @@ q.q
  ORDER BY ts_rank_cd(ARRAY[0.01,0.05,0.1,1], tsvector, q.q);
```

Индекс!

```
CREATE INDEX doc_fts ON doc_page USING GIN (tsvector);
```

Фасетный поиск

Надо поделить как-то документы в нашей БД на группы.
Сделаем это грубо.

```
ALTER TABLE doc_page ADD part INT;

UPDATE doc_page
    SET part = CASE WHEN m IS NULL THEN 0 ELSE m[1]::int END
    FROM (
       SELECT id,
         regexp_match(text, 'Глава.([0-9]+)') as m from doc_page) x
      WHERE x.id = doc_page.id;
```

Фасетный запрос

```
WITH q AS (SELECT plainto_tsquery('ts', 'следующую задачу') AS q),
  c AS (
   SELECT *, RANK() OVER (
        PARTITION BY part
        ORDER BY
        ts_rank_cd(ARRAY[0.01,0.05,0.1,1], tsvector, q.q ) DESC, id
   ) rank,
   COUNT(*) OVER (PARTITION BY part) cnt
   FROM (
      SELECT doc_page.title, part, id, tsvector, ts_headline('ts', text, q.q) AS snp
            FROM doc_page, q WHERE tsvector @@ q.q
            ORDER BY  ts_rank_cd(ARRAY[0.01,0.05,0.1,1], tsvector,  q.q)
   ) p, q
) SELECT part, rank, cnt, title FROM c WHERE rank <=3;
```

Как сагрегировать см
http://akorotkov.github.io/blog/2016/06/17/faceted-search/

Спасибо за внимание!

Контакты:
info@postgrespro.ru
https://www.postgrespro.ru/