Užitečné (a neprávem opomíjené) extenze

Tomáš Vondra, GoodData
tomas.vondra@gooddata.com
tv@fuzzy.cz / @fuzzycz

contrib

- 42 modulů přímo v PostgreSQL repositáři
  http://www.postgresql.org/docs/devel/static/contrib.html
- administrační nástroje
  - monitoring dotazů, analýza databáze
- knihovny užitečných funkcí
  - pgcrypto, adminpack, ...
- příklady možností rozšíření
  - nové datové typy, podpora indexů, FDW

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

```
CREATE TABLE moje_tabulka (
    id          INTEGER,
    fulltext    TSVECTOR
);

CREATE INDEX fulltext_idx
     ON moje_tabulka USING GIST (id, fulltext);
```

```
CREATE TABLE moje_tabulka (
    id          INTEGER,
    fulltext    TSVECTOR
);

CREATE INDEX fulltext_idx
     ON moje_tabulka USING GIST (id, fulltext);
```

ERROR: data type integer has no default operator class for access method "gist"

:-(

```
CREATE TABLE moje_tabulka (
    id          INTEGER,
    fulltext    TSVECTOR
);

CREATE EXTENSION btree_gist;

CREATE INDEX fulltext_idx
     ON moje_tabulka USING GIST (id, fulltext);
```

;-)

- B-tree - běžné (stromové) indexy
  - standardní "skalární" datové typy (INT, TEXT, ...)
- GIN / GiST - "prostorové" indexy
  - fulltext, intervaly (range), ...
- btree_gin / btree_gist
  - umožňuje vytvořit GIN / GiST index nad skalárními datovými typy
- Proč ne prostě dva samostatné indexy?
  - lze je zkombinovat pomocí bitmap index scanu
  - existují situace vyžadující složený index

- exclusion constraints
  - omezení podobná UNIQUE
  - např. u intervalů chceme aby se nepřekrývaly

- příklad
  - rezervace několika místností
  - nechceme aby se doby rezervací překrývaly

```
CREATE TABLE rezervace_mistnosti (
    id_rezervace  SERIAL PRIMARY KEY,
    id_mistnost   INTEGER,
    rezervace     TSRANGE,
    EXCLUDE USING gist (mistnost  WITH =,
                        rezervace WITH &&)
);
```

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance **file_fdw** fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

```
CREATE EXTENSION file_fdw;

CREATE SERVER file_server
       FOREIGN DATA WRAPPER file_fdw;

CREATE FOREIGN TABLE sklad_csv (
  produkt        INT,
  nazev          TEXT,
  ks_na_sklade   INT
)
SERVER file_server OPTIONS (
  filename '/data/stav_skladu.csv',
  format 'csv'
);

SELECT * FROM sklad_csv;
```

```
WITH upsert as
(UPDATE sklad SET nazev = src.nazev,
            ks_na_sklade = src.ks_na_sklade
    FROM sklad_csv src
  WHERE sklad.produkt = src.produkt
  RETURNING sklad.produkt
)
INSERT INTO sklad
SELECT produkt, nazev, ks_na_sklade
  FROM sklad_csv
 WHERE produkt NOT IN (SELECT * FROM upsert);
```

- standardní postup
  - načíst data do pomocné tabulky pomocí COPY
  - použít writable CTE

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance **file_fdw** fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

```
CREATE EXTENSION hstore;

SELECT 'a=>1, b=>2, c=>3'::hstore;
            hstore
------------------------------
 "a"=>"1", "b"=>"2", "c"=>"3"
```

`hstore ? key`

obsahuje hstore klíč "key"?

`hstore -> key`

vrátí hodnotu pro klíč "key"

`hstore @> hstore`

obsahuje první hstore ten druhý (všechny klíče se stejnými hodnotami)?

- chceme si uložit všechny e-maily
  - nic nechceme zahodit (ani hlavičky)
  - efektivní a co nejjednodušší dotazování

```
CREATE TABLE mail (

    id          INT PRIMARY KEY,
    date_sent   TIMESTAMP,
    addr_from   TEXT,
    addr_to     TEXT[],
    addr_cc     TEXT[],
    subject     TEXT,
    body        TEXT,

    ... a asi bambilión různých hlaviček ...

);
```

- sloupec pro každou hlavičku?
  - vede na megaširoké tabulky
  - ... a navíc hlavičky nejsou pevně dané :-(
- generické sloupce (PARAM1_NAME, PARAM2_VALUE)
  - naprosto proti relačním principům
  - v podstatě nemožné dotazy na PARAM sloupcích
  - zkrátka "Fuj!"
- EAV schéma

```
CREATE TABLE mail_headers (
    id              INT  REFERENCES mail(id),
    header_name     TEXT,
    header_value    TEXT
);
```

Naštěstí hstore přichází na pomoc!

```
CREATE TABLE mail (
    id          INT PRIMARY KEY,
    ...
    reply_to    TEXT,
    subject     TEXT,
    body        TEXT,
    headers     HSTORE
);

CREATE INDEX mail_headers_idx
          ON mail USING GIST (headers);

SELECT * FROM mail
 WHERE headers ? 'reply-to';

SELECT (headers -> "content-type") FROM mail
 WHERE headers @> '"reply-to" => "tv@fuzzy.cz"';
```

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo **ltree**
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

- stromová struktura v relační DB je PITA ;-)
- např. hierarchie kategorií v eshopu

```
CREATE TABLE kategorie (
    id          INT PRIMARY KEY,
    parent_id   INT REFERENCES kategorie(id),
    nazev       TEXT
);
```

- ... a teď mi pro kategorii X vyhledejte např.
  - všechny nadřazené kategorie
  - všechny (nejen přímé) podkategorie
  - všechny produkty (včetně podkategorií)
- nepříliš efektivní :-(

- LTREE typ - cesta stromem

```
KATEGORIE1
KATEGORIE1.KATEGORIE2
KATEGORIE1.KATEGORIE3.KATEGORIE4
```

- tj. například

```
Top
Top.Komponenty
Top.Pocitace.Desktopy
Top.Pocitace.Notebooky
```

```
CREATE TABLE kategorie (
    path        LTREE PRIMARY KEY,
    nazev       TEXT
);
```

- nemohou vznikat cykly (na rozdíl od FK)
- mohou vznikat "gapy" (stejně jako u FK)

<u>možnosti dotazování</u>

- ltree vs. ltree

```
SELECT * FROM kategorie WHERE 'Top.Pocitace' @>
path;
```

- ltree vs. lquery - jednoduché dotazy

```
SELECT * FROM kategorie WHERE path ~ '*.Pocitace.*';
```

- ltree vs. ltxquery - fulltextové dotazy (AND, OR, ...)

```
SELECT * FROM kategorie WHERE path @ 'Pocitac*';
```

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo **ltree**
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

- pg_stat_activity je "aktuální stav"
  - fajn pro monitoring dlouhých dotazů
  - krátké dotazy se tak monitorují těžko

- pg_stat_statements to the rescue!
  - `calls` - počet volání
  - `total_time` - čas strávený v dotazech
  - `rows` - počet vrácených řádek
  - `shared_blks_*` - statistiky shared bloků
  - `local_blks_*` - statistiky lokálních bloků
  - `temp_blks_*` - statistiky temp bloků
  - `blk_*_time` - čas čtení/zápisu bloků

DEMO

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple pg_trgm
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple **pg_trgm**
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

- trigramy - skupiny trojic písmen ve slově

```
SELECT show_trgm('auto');

           show_trgm
-------------------------------
 {"  a"," au",aut,"to ",uto}
```

- podobnost slov

```
SELECT similarity('auto', 'autem');

 similarity
------------
   0.375
```

- podobnost je vlastně inverzní vzdálenost

```
SELECT ('auto' <-> 'autem') AS distance;

 distance
----------
    0.625
```

- a na vzdálenosti jsou vybudovány GIN/GiST indexy

```
CREATE TABLE tabulka (t text);
CREATE INDEX trgm_idx ON tabulka
       USING GIST (t gist_trgm_ops);

SELECT t, (t <-> 'hledaný text') AS dist
  FROM tabulka ORDER BY dist DESC LIMIT 10;

SELECT t FROM tabulka WHERE t LIKE '%aaa%';  -- 9.1
SELECT t FROM tabulka WHERE t ~ '(aaa|bbb)'; -- 9.3
```

adminpack auth_delay auto_explain
btree_gin btree_gist chkpass citext cube
dblink dict_int dict_xsyn dummy_seclabel
earthdistance file_fdw fuzzystrmatch
hstore intagg intarray isn lo ltree
pageinspect passwordcheck pg_buffercache
pgcrypto pg_freespacemap pgrowlocks
pg_stat_statements pgstattuple **pg_trgm**
postgres_fdw seg sepgsql spi sslinfo
tablefunc tcn test_parser tsearch2
unaccent uuid-ossp xml2

pgxn.org

- nezávislý (a neoficiální) repositář extenzí
- web - vyhledávání apod.
- nástroje pro jednodušší instalaci
  - pgxn client
  - stáhne / rozbalí / nainstaluje do databáze

```
$ sudo apt-get install pgxnclient
$ pgxnclient --help
$ pgxnclient install quantile
$ pgxnclient load -d testDB quantile
```

PGXN

PostgreSQL Extension Network

in Documentation PGXN Search

acl adaptive **administration** **aggregate**
aggregate function amazon amqp analysis analytics analyze api
array automation average bitmap compatibility **count**
data types **datatype** **dictionary** **distinct**
estimate **external data** **fdw**
foreign data wrapper **function**
functions hash integer internet ispell **json** ldap
log logging **maintenance** md5 mysql oracle
partitioning perl pl queries **record** **replication** row
sampling sha sha1 **sql med** statistics **table** trigger
version version number web

PGXN, the PostgreSQL Extension network, is a central distribution system for open-source PostgreSQL extension libraries.

Founders

myYearbook

PGX

Dalibo

Patrons

Enova Financial

Benefactors

- Etsy
- US PostgreSQL Association
- Command Prompt, Inc.

quantile

- agregační funkce
- výpočet kvantilů (resp. percentilů)

```
SELECT
    id_oddeleni,
    quantile(plat, 0.5) AS median_platu
FROM zamestnanci
GROUP BY id_oddeleni;
```

pg_repack

- pokročilá údržba tabulek / indexů
  - online CLUSTER
  - order by column
  - online VACUUM FULL
- a to bez exclusivních zámků!
- eliminace bloatu (hlavně indexů)
- ale opatrně s DDL (data corruption)

```
$ pg_repack --table=tabulka mojedb
```

- tradičně

```
VACUUM FULL velka_dulezita_tabulka;

CLUSTER velka_dulezita_tabulka;

-- spousta křiku od uživatelů (a šéfa),
-- bere si totiž ACCESS EXCLUSIVE zámek
```

- pg_reorg

```
$ pg_reorg --no-order \
           --table velka_dulezita_tabulka mojedb

$ pg_reorg --table velka_dulezita_tabulka mojedb
```

pg_partman

- management partitioningu
  - dvě varianty partitioningu - ID a čas
  - parametrizace: volitelný detail a retence
  - sada PL/pgSQL funkcí + python skriptů (admin)
- dědění vlastností z parent tabulky
  - default hodnoty, indexy, constrainty
  - práva / vlastnictví
- konfigurovatelná retence
- fajn pokud odpovídá vašemu use-case

- vytvoření partitionované tabulky

```
CREATE TABLE moje_parent_tabulka (
    col1 SERIAL,
    col2 TEXT,
    col3 TIMESTAMPTZ DEFAULT now()
);

SELECT part.create_parent('moje_parent_tabulka',
                  'col3', 'time-static', 'daily');
```

- ještě pravidelně vytvářet nové (cron)

```
run_maintenance()
```

... a další

- pgTAP
  - unit testy pro TAP (Test Anything Protocol)
- plproxy
  - sharding pomocí procedurálního jazyka
- plv8
  - JavaScript jako procedurální jazyk (V8 engine)
- s3_fdw
  - FDW přístup do S3 na Amazon cloudu
- semver
  - datový typ pro sémantické verzování (semver.org)