Kodlamadan Dağıtıma PostgreSQL'de Eklenti Geliştirmek

Burak Yücesoy

PostgreSQL eklentisi nedir?

- PostgreSQL'e yeni yetenekler ekleyen her türlü yazılım bir PostgreSQL eklentisidir.
- Genellikle her eklenti işini yapabilmek için gerekli nesnelerle beraber paketlenmiş olarak gelir.
- "**CREATE EXTENSION** *extension_name*;", eklenti için gerekli nesneleri veritabanına yükler.

Örnek eklenti: pg_cron

- Veritabanı içinde bir görev zamanlayıcısı; veritabanında periyodik işlem yapmanıza olanak sağlar.
- Her cumartesi, saat 03:30 da eski verileri sil;
  - psql> SELECT cron.schedule('30 3 * * 6', $$DELETE FROM events WHERE event_time < now() - interval '1 week'$$);
- Her gün saat 10:00 da VACUUM çalıştır;
  - psql> SELECT cron.schedule('0 10 * * *', $$VACUUM$$);

Neden PostgreSQL eklentileri?

- Her geçen gün veritabanlarını kullanma şeklimiz dolayısıyla veritabanından beklentilerimiz değişiyor.
- Farklı ihtiyaçları karşılamak adına, şimdiye kadar PostgreSQL onlarca kez fork edildi.
- Eklenti altyapısını kullanarak, PostgreSQL'i fork etmeden, PostgreSQL'e yeni özellikler kazandır!

PostgreSQL Eklenti Altyapısı

- Veri tipleri
- Fonksiyonlar
- Operatörler
- Foreign Data Wrapper
- PostgreSQL modülleri (planlayıcı, depolama motoru, vb.)

Veri Tipleri

- PostgreSQL'de hali hazırda onlarca veri tipi var;
  - bigint, text, timestampz, jsonb...
- PostgreSQL'de normalde olmayan bir veri tipini eklentiler vasıtasıyla eklemek mümkün;
  - ip adresi, e-mail, hll...
- **CREATE TYPE** *type_name*;

Veri Tipleri

```
CREATE TYPE name AS
    ( [ attribute_name data_type [ COLLATE collation ] [, ... ] ] )

CREATE TYPE name AS ENUM
    ( [ 'label' [, ... ] ] )

CREATE TYPE name AS RANGE (
    SUBTYPE = subtype
    [ , SUBTYPE_OPCLASS = subtype_operator_class ]
    [ , COLLATION = collation ]
    [ , CANONICAL = canonical_function ]
    [ , SUBTYPE_DIFF = subtype_diff_function ]
)

CREATE TYPE name (
    INPUT = input_function,
    OUTPUT = output_function
    [ , RECEIVE = receive_function ]
    [ , SEND = send_function ]
    [ , TYPMOD_IN = type_modifier_input_function ]
    [ , TYPMOD_OUT = type_modifier_output_function ]
    [ , ANALYZE = analyze_function ]
    [ , INTERNALLENGTH = { internallength | VARIABLE } ]
    [ , PASSEDBYVALUE ]
    [ , ALIGNMENT = alignment ]
    [ , STORAGE = storage ]
    [ , LIKE = like_type ]
    [ , CATEGORY = category ]
    [ , PREFERRED = preferred ]
    [ , DEFAULT = default ]
    [ , ELEMENT = element ]
    [ , DELIMITER = delimiter ]
    [ , COLLATABLE = collatable ]
)

CREATE TYPE name
```

Fonksiyonlar (UDFs)

- PostgreSQL'de hali hazırda var olan onlarca veri tipi üzerinde çalışabilen binlerce fonksiyon var.
- Yeni bir veri tipi eklediğinizde, o veri tipiyle çalışabilen fonksiyonları da ekleyerek, veri tipinizi daha işlevsel hale getirebilirsiniz
- Ya da var olan veri tipleri üzerinde çalışan bir fonksiyon ekleyebilirsiniz.
- **CREATE FUNCTION** *function_name*;

Fonksiyonlar (UDFs)

```
CREATE [ OR REPLACE ] FUNCTION
    name ( [ [ argmode ] [ argname ] argtype [ { DEFAULT | = } default_expr ] [, ...] ] )
    [ RETURNS rettype
      | RETURNS TABLE ( column_name column_type [, ...] ) ]
  { LANGUAGE lang_name
    | TRANSFORM { FOR TYPE type_name } [, ... ]
    | WINDOW
    | IMMUTABLE | STABLE | VOLATILE | [ NOT ] LEAKPROOF
    | CALLED ON NULL INPUT | RETURNS NULL ON NULL INPUT | STRICT
    | [ EXTERNAL ] SECURITY INVOKER | [ EXTERNAL ] SECURITY DEFINER
    | PARALLEL { UNSAFE | RESTRICTED | SAFE }
    | COST execution_cost
    | ROWS result_rows
    | SET configuration_parameter { TO value | = value | FROM CURRENT }
    | AS 'definition'
    | AS 'obj_file', 'link_symbol'
  } ...
```

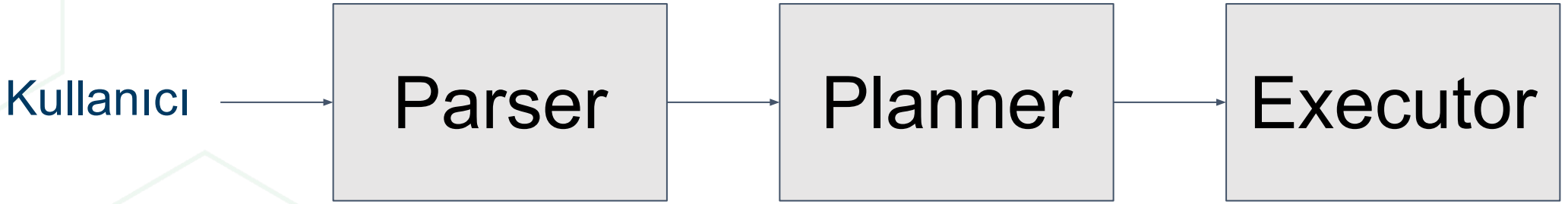
Operatörler

- +, -, *, /, >, <, >=, <=, =, <>...
- Var olan veri tipleri üzerinde veya yeni tanımladığınız veri tipleri için operatörler tanımlayabilirsiniz.
- Mesela ip adresi veri tipi için >, < gibi operatörler mantıklı olabilir.
  - Ör: 34.56.125.200 > 12.250.130.10

Foreign Data Wrappers

- Başka veri kaynaklarına PostgreSQL vasıtasıyla erişebilirsiniz;
  - file_fdw
  - mongo_fdw
  - oracle_fdw
  - s3_fdw
  - twitter_fdw
    - psql> SELECT from_user, created_at, text FROM twitter WHERE q = '#postgresql';

PostgreSQL modülleri



Kendi Eklentimizi Yazalım...

- Renk veri tipi;
  - pg_color
  - Kullanıcılarımın en sevdiği rengi veritabanında tutmak istiyorum

Kendi Eklentimizi Yazalım... Kodlama

- Gerekenler;
  - pg_color.control
  - pg_color.sql
  - pg_color.c
  - Makefile

Kendi Eklentimizi Yazalım... Test Etme

- PostgreSQL'in kendi test etme altyapısı eklentiler için de kullanılabilir.

Kendi Eklentimizi Yazalım... Paketleme

- PostgreSQL yaygın olarak RedHat ve Debian tabanlı sistemlerde kullanılıyor.
- Her sistemde çalışacak binaryler için derleme ve paketlemeyi ilgili sistemde yapmanız gerekiyor.
- Bu alanda docker kurtarıcı.
- Citus Data, açık kaynak paketleme toolları; https://github.com/citusdata/packaging

Kendi Eklentimizi Yazalım... Paketleme

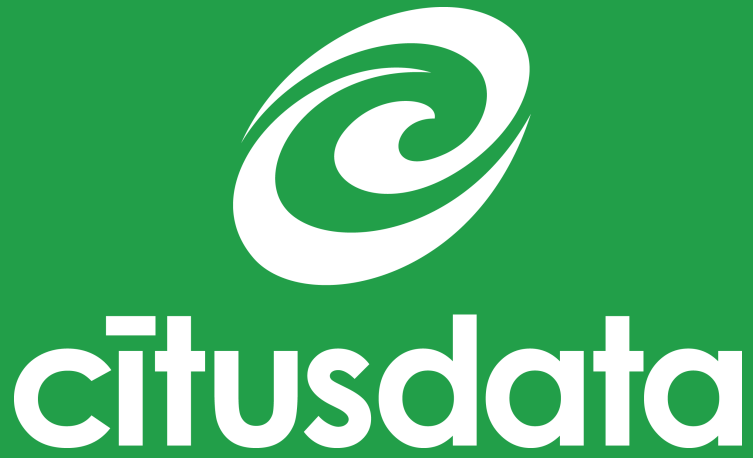
- Gerekenler;
  - debian/pgversions
  - debian/control.in
  - debian/changelog
  - debian/copyright
  - debian/rules
  - debian/compat

Kendi Eklentimizi Yazalım... Paketleme

- > pg_buildext updatecontrol
- > debuild -uc -us -B --lintian-opts --profile debian --allow-root

Kendi Eklentimizi Yazalım... Dağıtım

- PostgreSQL community software repositories
- PGXN
- Kendi reponuz;
  - Herhangi bir servera kendi repositorynizi kurabilir, insanların eklentinizi yükleme isteklerinize kendi serverınızdan cevap verebilirsiniz.
  - Managed servisleri kullanabilirsiniz; packagecloud.io



Teşekkürler & Sorular

Burak Yücesoy
burak@citusdata.com
@byucesoy

www.citusdata.com | **@citusdata**