Bazy Danych

Andrzej M. Borzyszkowski
Instytut Informatyki

Uniwersytetu Gdańskiego

materiały dostępne elektronicznie
http://inf.ug.edu.pl/~amb

PostgreSQL – administracja

© Andrzej M. Borzyszkowski
Bazy Danych

Instalacja

- Pliki konfiguracyjne
  - pg_hba.conf uwierzytelnianie hostów
  - pg_ident.conf uwierzytelnianie użytkowników
  - PG_VERSION numer wersji
  - postgresql.conf konfiguracja serwera
  - postmaster.opts opcje postmastera w czasie startu
  - postmaster.pid backup w razie awarii
- Na początku musi być użytkownik postgres
  - może specjalny katalog
  - uruchamia on initdb – pierwsza baza danych

© Andrzej M. Borzyszkowski
Bazy Danych

Serwer

- Działa non-stop
- Może/ raczej jest uruchamiany automatycznie, w czasie startu systemu operacyjnego
- Typowe wywołanie:

/usr/bin/postgres -D /var/lib/pgsql/data -p 5432

  - można umożliwić dostęp sieciowy
  - można zażądać połączenia bezpiecznego przez SSL
- Ręczne zatrzymanie/uruchomienie bazy danych poprzez polecenie pg_ctl

© Andrzej M. Borzyszkowski
Bazy Danych

Zarządzanie bazą danych PostgreSQL

- Instalacja
  - aktualizacja
- Sterowanie serwerem
  - uruchamianie, zatrzymywanie, dzienniki
- Zarządzanie użytkownikami
  - tworzenie i usuwanie kont
  - prawa dostępu
- Zarządzanie danymi
  - tworzenie i usuwanie baz danych
  - kopie zapasowe

Dane

- Polecenia dla powłoki postgresa i powłoki uniksa
  - `CREATE DATABASE moja_baza TEMPLATE=template1 ENCODING='utf-8';`
    - (tu już wiadomo, kto jest użytkownikiem)
  - `DROP DATABASE moja;`
  - createdb/dropdb -U użytkownik moja_baza
- Kopie zapasowe
  - pg_dump moja_baza > plik.bak
  - otworzenie 1. createdb nowa, 2. psql -f plik.bak nowa
  - można działać globalnie pg_dumpall, ale wówczas odtwarzane są dawne nazwy baz na czystym systemie
  - pg_upgrade stosuje się przy zmianie wersji PostgreSQL
  - jest sensowne kompresowanie plików backup-ów

Typy danych w PostgreSQL

- Napisy: stałej długości `char`, zmiennej długości `varchar`, dowolnie długie `text`
  - 'napis'
  - '3.14' to też tylko napis
  - '' pusty napis, nie jest to NULL
  - problem z apostrofem:
    'O''Hara', 'O\'Hara', 'O\047Hara', $$O'Hara$$, $cokolwiek$O'Hara$cokolwiek$
  - konkatenacja napisów oznaczana ||
  - ~~ albo LIKE dopasowanie wzorca

Typy danych w PostgreSQL, c.d.

- Liczby
  - dokładne: smallint, integer, bigint, numeric(n,k), decimal (nawet dziesiątki tysięcy dokładnych cyfr)
  - money (dwa miejsca po przecinku)
  - przybliżone: real, double
  - 3.14, 2.0e+10 zawsze są przybliżone
- Liczniki (*sequence*)
  - tworzone są automatycznie dla atrybutu typu SERIAL
  - `SELECT currval ('klient_nr_seq');`
  - `SELECT setval ('klient_nr_seq',0)`
  - najlepiej pozostawić sekwencje systemowi
- Data/godzina
  - tylko jedno z powyższych lub oba (timestamp)
  - może być razem ze strefą czasową lub bez niej

Szczególne typy danych w PostgreSQL

- Typ wyliczeniowy
  - `CREATE TYPE week AS ENUM ('Mon','Tue','Wed','Thu','Fri','Sat','Sun');`
- Adresy sieciowe
  - MACADDR (sprzętowe) 00:04:a2:35:98:f2
  - INET (Internetowe) 192.134.14.1
  - możliwość badania części adresu: ip1<<=ip2
- Dane geometryczne
  - punkty
  - odcinki
  - prostokąty
  - ścieżki
  - wielokąty
  - okręgi

Szczególne typy danych w PostgreSQL, c.d.

- OID (*object identifier*)
  `SELECT OID, * FROM klient`
  - wyświetli również ukryte identyfikatory (w dawniejszych wersjach)
- BLOB (*binary large object*)
  - duży niekoniecznie, ale najczęściej (do 1Gb)
  - np. fotografie, muzyka, oprogramowanie
  - jedyną operacją jest konkatenacja
  - można przechowywać tylko referencję do obiektu

```
CREATE TABLE mojecd ...., cd_id OID;
INSERT INTO mojecd VALUES .....
lo_import('/cd/niemen');
SELECT * FROM mojecd  wyświetli m.in.
identyfikator płyty
SELECT lo_export (1234567, '/cd/niemen');
SELECT lo_unlink (1234567);
```


Pola wielokrotne

- Jest to naruszenie 1 postaci normalnej !!

```
CREATE TABLE pracownik   .... płaca
NUMERIC(7,2)[12]
INSERT INTO pracownik VALUES ....'{1200, 1250,
1190, …}'
SELECT płaca[10:12] FROM pracownik
```

- Trzeba zachować ostrożność:
  - system pozwala wstawiać do dalszych pól niż deklarowano, [12] jest tylko komentarzem
  - nie ma wartości NULL, elementy w polu wielokrotnym są wstawiane po kolei bez przerw

```
INSERT INTO pracownik(.... płaca[3])
VALUES ....'{1200}'
```

zakończy się wstawieniem na pole pierwsze

Dziedziczenie

- Pewne elementy obiektowości

```
CREATE TABLE film (tytuł varchar(100), ...);
CREATE TABLE dvd (wersja_jęz  varchar[])
INHERITS film;
CREATE TABLE video () INHERITS film;
INSERT INTO dvd VALUES ('Harry', ..., '{pol,
ang, napisy}');
```

część danych znajdzie się w tabeli “film”, a część “dvd”

```
SELECT * FROM dvd; --wyświetli wszystkie dane o dvd
SELECT * FROM video; --wyświetli tylko taśmy
SELECT * FROM film;  --wyświetli wszystkie filmy, na taśmie i dvd
SELECT ONLY dvd FROM film;  -- wyświetli dane o filmach umieszczonych na dvd
```

Typy indeksów w PostgreSQL

- B-drzewo
- Hash (adresowanie bezpośrednie)
- GiST, SP-GiST
- BRIN (block range index)

- Indeksy mogą być tworzone przez użytkownika,
  - są domyślnie zakładane jeśli atrybut jest kluczem kandydującym
  - mogą być zakładane również dla zestawów atrybutów
  - mogą być zakładane dla wartości funkcji od atrybutów, np. lower()
  - mogą wymusić jednoznaczność wartości funkcji
    CREATE UNIQUE INDEX nazwisko_jedn ON klient (lower(nazwisko));

Indeksy w PostgreSQL

- Indeksy mogą być tworzone przez użytkownika,
  - są domyślnie zakładane jeśli atrybut jest kluczem kandydującym
  - mogą być zakładane również dla zestawów atrybutów
  - mogą być zakładane dla wartości funkcji od atrybutów, np. lower()
  - mogą wymusić jednoznaczność wartości funkcji
    ```
    CREATE UNIQUE INDEX nazwisko_jedn ON klient (lower(nazwisko));
    ```
- Nie powinny być zakładane dla danych często się zmieniających
  - dla rzadko występujących w pytaniach
  - dla wartości, które same się często powtarzają

Problemy wydajności

- Pamięć operacyjna vs. pamięć dyskowa
  - pamięć operacyjną trzeba oszczędzać
  - pamięć dyskowa jest wolna
- Można się dowiedzieć, które tablice są najczęściej używane pg_stat_all_tables
- Dane są umieszczane w kolejności klucza głównego – przyspiesza to wyszukiwanie w/g tego klucza
- Automatyczna optymalizacja
  - PostgreSQL oblicza koszt kilku sposobów wykonania polecenia i wybiera najmniejszy
  - np. dla złączenia T1 JOIN T2 ON T1.A1=T2.A2 może istnieć indeks dla atrybutu A1 w T1, wówczas warto przejrzeć po kolei T2 i dla każdego wiersza znajdować szybko wiersz w T1

Automatyczna optymalizacja

- Można zbadać sposób optymalizacji:
  ```
  EXPLAIN ANALYZE SELECT * FROM moje_dane;
  ```
  - wykonane jest zapytanie oraz jest starannie objaśnione
  - można tylko prosić o plan wykonania, ale bez działania
  - typowa odpowiedź
    ```
    Merge Join
     -> Sort
        -> Seq Scan on .....
     -> Sort
        -> Index Scan on .....
    ```
  - inne operacje: Unique, LIMIT, Aggregate, Append (używany przy UNION oraz przy dziedziczeniu), Group, Hash Join
- Generalnie lepiej nie próbować zmieniać optymalizacji