



Technologia i rozwiązania

PostgreSQL

Receptury dla administratora

Poznaj najlepsze przepisy na pracę z PostgreSQL!

Tu znajdziesz rozwiązania najczęściej spotykanych problemów!

- Jak zapewnić bezpieczeństwo bazom danych?
- Jak uruchomić kilka serwerów baz danych w ramach jednego systemu operacyjnego?
- Jak zarządzać użytkownikami i ich uprawnieniami?

Helion



Simon Riggs, Hannu Krosing

PACKT
PUBLISHING

» Idź do

- Spis treści
- Przykładowy rozdział
- Skorowidz

» Katalog książek

- Katalog online
- Zamów drukowany katalog

» Twój koszyk

- Dodaj do koszyka

» Cennik i informacje

- Zamów informacje o nowościach
- Zamów cennik

» Czytelnia

- Fragmenty książek online

» Kontakt

Helion SA
ul. Kościuszki 1c
44-100 Gliwice
tel. 32 230 98 63
e-mail: helion@helion.pl
© Helion 1991–2011

PostgreSQL. Receptury dla administratora

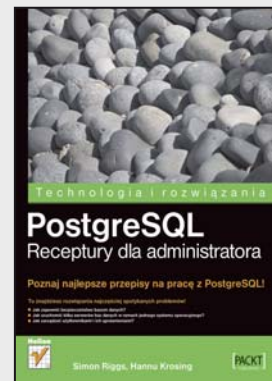
Autor: Simon Riggs, Hannu Krosing

Tłumaczenie: Mikołaj Szczepaniak

ISBN: 978-83-246-3061-5

Tytuł oryginału: [PostgreSQL 9 Administration Cookbook](#)

Format: 170×230, stron: 408



Poznaj najlepsze przepisy na pracę z PostgreSQL!

- Jak zapewnić bezpieczeństwo bazom danych?
- Jak uruchomić kilka serwerów baz danych w ramach jednego systemu operacyjnego?
- Jak zarządzać użytkownikami i ich uprawnieniami?

PostgreSQL to jedna z najbardziej zaawansowanych baz danych o otwartym kodzie źródłowym. Przez wiele lat była niedoścignionym wzorem dla innego darmowego rozwiązania – MySQL. Dziś znajduje zastosowanie wszędzie tam, gdzie wymagana jest najwyższa niezawodność i wydajność, a brak konieczności zapłaty gra kluczową rolę. Stosunek jakości do ceny w przypadku PostgreSQL zmierza do nieskończoności!

Trzymasz w rękach książkę zawierającą liczne przepisy na najlepsze wykorzystanie PostgreSQL. System ten sprawdza się zawsze, gdy chcesz szybko i bezproblemowo osiągnąć zamierzone cele. W trakcie lektury dowiesz się, jak nawiązać połączenie z serwerem, skorzystać z graficznych lub tekstowych narzędzi administracyjnych oraz bezpiecznie zmienić hasło administratora. Ponadto nauczysz się kontrolować przestrzeń dyskową wykorzystywaną przez poszczególne bazy danych, tworzyć tabele, ładować dane oraz zarządzać użytkownikami i ich uprawnieniami. Autorzy dużo miejsca poświęcają kwestii bezpieczeństwa. W końcu dane to najcenniejsza rzecz, jaką przechowuje się w bazach! Każdy z rozdziałów przynosi ogrom wiedzy o różnym poziomie skomplikowania. Zaawansowanych użytkowników zainteresuje rozdział poświęcony replikacji, a tych początkujących rozdział traktujący o uruchamianiu i zatrzymywaniu serwera baz danych. Ta książka przyda się po prostu wszystkim użytkownikom PostgreSQL!

- Zalety PostgreSQL w kontekście innych rozwiązań bazodanowych
- Udostępnianie serwera w sieci
- Zastosowanie narzędzia psql do wykonywania zapytań
- Sprawdzanie wersji serwera
- Lista baz danych na serwerze
- Planowanie nowej bazy danych
- Parametry, ich znaczenie i ustawianie
- Uruchamianie i zatrzymywanie serwera
- Ponowne ładowanie plików konfiguracyjnych
- Przyznawanie użytkownikom własnych baz danych
- Wiele serwerów baz danych w ramach jednego systemu operacyjnego
- Generowanie danych testowych
- Tworzenie kont użytkowników i zarządzanie nimi oraz ich uprawnieniami
- Równoległe wykonywanie zadań – polecenie pg_batch
- Monitorowanie i diagnostyka serwera PostgreSQL
- Przygotowywanie kopii bezpieczeństwa

Zobacz, co możesz osiągnąć razem z bazą PostgreSQL!

Spis treści

O autorach	9
O recenzentach	11
Przedmowa	13
Rozdział 1. Pierwsze kroki	19
Wprowadzenie	19
Wprowadzenie do systemu PostgreSQL 9	20
Jak zdobyć system PostgreSQL	22
Łączenie z bazą danych PostgreSQL	24
Umożliwianie zdalnego dostępu do serwera za pośrednictwem sieci	27
Korzystanie z graficznych narzędzi administracyjnych	29
Stosowanie narzędzi psql do wykonywania zapytań i skryptów	35
Bezpieczna zmiana hasła	39
Unikanie trwałego kodowania hasła	40
Stosowanie pliku usługi połączeń	42
Rozwiązywanie problemów związanych z nawiązywaniem połączenia	43
Rozdział 2. Poznawanie bazy danych	47
Wprowadzenie	47
Która wersja serwera?	48
Od kiedy działa dany serwer?	50
Lokalizacja plików serwera bazy danych	51
Lokalizacja dziennika komunikatów serwera bazy danych	53
Lokalizacja identyfikatora systemu bazy danych	56
Lista baz danych na danym serwerze bazy danych	57
Ile tabel w bazie danych?	60
Ile przestrzeni dyskowej zajmuje baza danych?	63
Ile przestrzeni dyskowej zajmuje tabela?	64
Które tabele są największe?	65
Ile wierszy w tabeli?	65
Szybkie szacowanie liczby wierszy w tabeli	67
Odkrywanie zależności łączących obiekty	71

Rozdział 3. Konfiguracja	75
Wprowadzenie	75
Lektura podręcznika użytkownika (RTFM)	76
Planowanie nowej bazy danych	77
Zmiana parametrów na poziomie programów	79
Jakie są bieżące ustawienia konfiguracyjne?	81
Które parametry zawierają wartości inne niż ustawienia domyślne?	82
Aktualizacja pliku parametrów	84
Ustawianie parametrów dla konkretnych grup użytkowników	85
Lista podstawowych zadań związanych z konfiguracją serwera	87
Dodawanie modułu zewnętrznego do systemu PostgreSQL	89
Uruchamianie serwera w trybie oszczędzania energii	91
Rozdział 4. Kontrola serwera	93
Wprowadzenie	93
Ręczne uruchamianie serwera bazy danych	94
Szybkie i bezpieczne zatrzymywanie serwera	95
Awaryjne zatrzymywanie serwera	96
Ponowne ładowanie plików konfiguracyjnych serwera	97
Szybkie restartowanie serwera	99
Zapobieganie nowym połączeniom	101
Ograniczanie liczby sesji dla każdego użytkownika do jednej	103
Rozłączanie użytkowników	104
Projektowanie pod kątem obsługi wielu podmiotów	106
Stosowanie wielu schematów	107
Przyznawanie użytkownikom własnych, prywatnych baz danych	110
Uruchamianie wielu serwerów w jednym systemie	112
Konfigurowanie puli połączeń	113
Rozdział 5. Tabele i dane	117
Wprowadzenie	117
Wybieranie właściwych nazw dla obiektów bazy danych	118
Obsługa obiektów z nazwami otoczonymi cudzysłowami	120
Wymuszanie stosowania tych samych definicji dla tak samo nazwanych kolumn	122
Identyfikacja i usuwanie powtarzających się wierszy	126
Zapobieganie występowaniu powtarzających się wierszy	129
Odnajdywanie unikatowego klucza dla zbioru danych	135
Generowanie danych testowych	137
Losowe próbkowanie danych	141
Ładowanie danych z arkusza kalkulacyjnego	143
Ładowanie danych ze zwykłych plików	146

Rozdział 6. Bezpieczeństwo	151
Wprowadzenie	151
Wycofywanie dostępu użytkownika do tabeli	153
Nadawanie użytkownikowi uprawnień dostępu do tabeli	155
Tworzenie nowego użytkownika	157
Tymczasowe uniemożliwienie użytkownikowi nawiązywania połączenia	158
Usuwanie użytkownika bez usuwania jego danych	160
Sprawdzanie, czy wszyscy użytkownicy stosują bezpieczne hasła	162
Nadawanie konkretnym użytkownikom ograniczonych uprawnień superużytkownika	163
Weryfikacja zmian wprowadzonych za pomocą wyrażeń języka DDL	166
Weryfikacja zmian w danych	168
Integracja z serwerem LDAP	171
Nawiązywanie połączenia SSL	172
Szyfrowanie poufnych danych	175
Rozdział 7. Administracja bazą danych	181
Wprowadzenie	181
Pisanie skryptu, który albo jest wykonywany w całości, albo nie jest wykonywany wcale	183
Pisanie skryptu narzędzia psql, który przerwie pracę w momencie napotkania pierwszego błędu	185
Wykonywanie operacji na wielu tabelach	187
Dodawanie i usuwanie kolumn tabeli	192
Zmiana typu danych kolumny	195
Dodawanie i usuwanie schematów	198
Przenoszenie obiektów pomiędzy schematami	200
Dodawanie i usuwanie przestrzeni tabel	201
Przenoszenie obiektów pomiędzy przestrzeniami tabel	205
Uzyskiwanie dostępu do obiektów należących do innych baz danych PostgreSQL	208
Umożliwianie aktualizacji perspektyw	214
Rozdział 8. Monitoring i diagnostyka	221
Wprowadzenie	221
Czy użytkownik jest połączony?	225
Co uruchamiają użytkownicy?	226
Czy użytkownicy są aktywni, czy zablokowani?	229
Kto blokuje użytkowników?	231
Zabijanie konkretnej sesji	232
Rozstrzygnięcie transakcji, której przygotowanie budzi wątpliwości	235
Czy ktokolwiek używa określonej tabeli?	235
Kiedy ktoś po raz ostatni używał tej tabeli?	237
Ile przestrzeni dyskowej zajmują dane tymczasowe?	240
Dlaczego spadła szybkość wykonywania zapytań?	242
Badanie błędów i przygotowywanie raportów	246
Generowanie codziennych podsumowań błędów zarejestrowanych w pliku dziennika	248

Rozdział 9. Bieżąca konserwacja	251
Wprowadzenie	251
Kontrola automatycznej konserwacji bazy danych	252
Unikanie automatycznego mrożenia i uszkodzeń stron	258
Unikanie przekręcania licznika transakcji	260
Usuwanie starych, przygotowanych transakcji	262
Czynności na rzecz użytkowników często korzystających z tabel tymczasowych	265
Identyfikacja i naprawianie przerośniętych tabel i indeksów	266
Konserwacja indeksów	271
Odnajdywanie nieużywanych indeksów	275
Ostrożne usuwanie niepotrzebnych indeksów	276
Planowanie konserwacji	278
Rozdział 10. Wydajność i przetwarzanie współbieżne	281
Wprowadzenie	281
Odnajdywanie wolnych wyrażeń języka SQL	282
Gromadzenie standardowych statystyk z perspektyw pg_stat*	285
Identyfikacja przyczyn wolnego działania wyrażeń języka SQL	287
Ograniczanie liczby zwracanych wierszy	291
Upraszczenie złożonych wyrażeń języka SQL	293
Przyspieszanie zapytań bez ich przebudowywania	299
Dlaczego zapytanie nie używa indeksu?	302
Jak wymusić na zapytaniu użycie indeksu?	303
Stosowanie techniki blokowania optymistycznego	305
Raportowanie o problemach związanych z wydajnością	307
Rozdział 11. Kopie zapasowe i odzyskiwanie baz danych	309
Wprowadzenie	310
Rozumienie procesu odzyskiwania danych	
po awarii oraz kontrola odpowiednich mechanizmów	310
Planowanie tworzenia kopii zapasowych	312
Logiczna kopia zapasowa jednej bazy danych tworzona w czasie rzeczywistym	316
Logiczna kopia zapasowa wszystkich baz danych tworzona w czasie rzeczywistym	318
Logiczna kopia zapasowa wszystkich tabel	
w pojedynczej przestrzeni tabel tworzona w czasie rzeczywistym	319
Kopia zapasowa definicji obiektów bazy danych	321
Autonomiczna, fizyczna kopia zapasowa bazy danych tworzona w czasie rzeczywistym	322
Fizyczna kopia bazy danych tworzona w czasie rzeczywistym i archiwizacja ciągła	325
Odzyskiwanie wszystkich baz danych	328
Odzyskiwanie do punktu w czasie	332
Odzyskiwanie usuniętej lub uszkodzonej tabeli	335
Odzyskiwanie usuniętej lub uszkodzonej przestrzeni tabel	338
Odzyskiwanie usuniętej lub uszkodzonej bazy danych	340
Podnoszenie wydajności tworzenia kopii zapasowych i (lub) odzyskiwania baz danych	341
Przyrostowe i różnicowe tworzenie kopii zapasowych i odzyskiwanie baz danych	345

Rozdział 12. Replikacja i aktualizacje	349
Wprowadzenie	349
Terminologia związana z replikacją	350
Zalecane praktyki replikacji	354
Replikacja poprzez przesyłanie dzienników w formie plików	356
Konfigurowanie replikacji poprzez strumieniowe przesyłanie dzienników	361
Zarządzanie replikacją poprzez przesyłanie dzienników	366
Zarządzanie trybem gorącej gotowości	370
Replikacja selektywna przy użyciu narzędzia Londiste	375
Replikacja selektywna przy użyciu narzędzia Slony 2.0	380
Równoważenie obciążeń za pomocą narzędzia pgbpool-II 3.0	385
Aktualizacje (podwersje)	389
Aktualizacje działającego serwera do wersji głównych	390
Aktualizacje do wersji głównych w sieci przy użyciu narzędzi do replikacji	393
Skorowidz	395

Konfiguracja

Ten rozdział zawiera następujące podrozdziały:

- „Lektura podręcznika użytkownika (RTFM)”
- „Planowanie nowej bazy danych”
- „Zmiana parametrów na poziomie programów”
- „Jakie są bieżące ustawienia konfiguracyjne?”
- „Które parametry zawierają wartości inne niż ustawienia domyślne?”
- „Aktualizacja pliku parametrów”
- „Ustawianie parametrów dla konkretnych grup użytkowników”
- „Lista podstawowych zadań związanych z konfiguracją serwera”
- „Dodawanie modułu zewnętrznego do systemu PostgreSQL”
- „Uruchamianie serwera w trybie oszczędzania energii”

Wprowadzenie

Otrzymuję mnóstwo pytań dotyczących ustawiania parametrów w systemie PostgreSQL.

Wszyscy jesteście zajęci, zatem większość z nas szuka źródła informacji, które pozwoliłoby w ciągu pięciu minut rozstrzygnąć wszystkie wątpliwości. Właśnie taki jest cel każdej książki złożonej z receptur, więc jesteśmy na najlepszej drodze do osiągnięcia wymarzonego ideału.

Niektórzy sądzą, że istnieją jakieś magiczne, uniwersalne ustawienia parametrów, które podniosą wydajność ich systemów. Poświęcają długie godziny na lekturę rozmaitych książek w poszukiwaniu wskazówek. Innych nie opuszcza zadowolenie, ponieważ odkryli jakąś witrynę internetową, która **wszystko wyjaśnia** — są pewni, że ich baza danych jest skonfigurowana prawidłowo.

W większości przypadków zrozumienie ustawień nie stanowi większego problemu. Dużo trudniejsze jest wypracowanie najlepszych ustawień. Co więcej, nawet optymalne ustawienia nierzadko zmieniają się wraz z rozwojem sytuacji. W tym rozdziale skoncentrujemy się przede wszystkim na tym, jak, kiedy i gdzie należy zmieniać ustawienia parametrów.

Lektura podręcznika użytkownika (RTFM)

Skrót **RTFM** (od ang. *Reading the Fine Manual*) często jest używany w roli bardziej opryskliwego odpowiednika stwierdzenia „nie przeszkadzaj, jestem zajęty”. Co ciekawe, zachęta do przeczytania podręcznika użytkownika w zdecydowanej większości przypadków jest najlepszą możliwą radą. Nie należy z góry odrzucać podobnych sugestii — powinniśmy raczej korzystać z tak cennych wskazówek. Warto przy tym pamiętać, że zawsze należy sięgać po podręcznik użytkownika, którego wersja odpowiada wersji serwera, na którym pracujemy.

Podręcznik użytkownika systemu PostgreSQL jest napisany w bardzo przemyślany sposób. Co więcej, omawia wybrane zagadnienia w sposób wyczerpujący. Jedną z jego największych wad jest to, że dokumenty składające się na ten podręcznik nie są zorganizowane z myślą o osobach, które dopiero poznają system PostgreSQL. Rozkład materiału przystosowano raczej do potrzeb osób poszukujących konkretnych informacji technicznych, tak aby mogły błyskawicznie ocenić, czy napotymane problemy wynikają na przykład z błędów samych użytkowników. Podręcznik w niektórych przypadkach odpowiada na pytanie **Co?**, ale już dużo rzadziej odpowiada na pytania **Dlaczego?** czy **Jak?**

Ponieważ sam uczestniczyłem w pisaniu wybranych sekcji dokumentacji systemu PostgreSQL, uważam za swój obowiązek zachęcić czytelnika do lektury tego materiału. Mimo wszystkich zalet podręcznika, ta książka zawiera mnóstwo przydatnych informacji, których nie można znaleźć w tamtych dokumentach.

Jak to zrobić...

Najważniejsze dokumenty dla każdego wydania są dostępne na następującej stronie internetowej: <http://www.postgresql.org/docs/manuals/>.

Największym zainteresowaniem cieszą się następujące fragmenty dokumentacji:

- dokumentacja poleceń języka SQL oraz podręcznik użytkownika narzędzi klienta i serwera (<http://www.postgresql.org/docs/9.0/interactive/reference.html>);
- konfiguracja (<http://www.postgresql.org/docs/9.0/interactive/runtime-config.html>);
- funkcje (<http://www.postgresql.org/docs/9.0/interactive/functions.html>).

Czytelnik może też pobrać podręcznik użytkownika w formie pliku PDF, który w pewnych przypadkach ułatwia wyszukiwanie informacji. Szczerze odradzam drukowanie tego dokumentu! Cała dokumentacja zajmuje ponad 2000 stron A4.

Jak to działa...

Dokumenty systemu PostgreSQL napisano w języku SGML, który pod wieloma względami przypomina język XML, choć w paru aspektach jest nieco inny. Pliki języka SGML są następnie przetwarzane i konwertowane na pliki w formatach HTML, PDF itp.

To nie wszystko...

Warto też zajrzeć do serwisu wiki w ramach witryny *postgresql.org*. Dodatkowe informacje można znaleźć także pod adresem *http://wiki.postgresql.org*.

Planowanie nowej bazy danych

Planowanie nowej bazy danych bywa żmudnym zadaniem. Dla wielu użytkowników mnogość opcji jest przytłaczająca — w tym podrozdziale zaproponujemy kilka przydatnych koncepcji. Należy wystrzegać się dróg na skróty i nieprzemyślanych działań w nadziei, że nasza ograniczona wiedza na zawsze pozwoli rozstrzygać wszelkie wątpliwości.

Przygotowania

Czytelnik powinien być przygotowany na to zadanie. Nie należy jednak oczekiwać, że ktoś dokładnie powie, co robić. Jeśli nie dysponujemy jasno sprecyzowanymi wymaganiami, powinniśmy sami je spisać, oznaczając każdą pozycję jako założenie, nie wymaganie — nie wolno mylić obu pojęć.

Należy przeprowadzić tyle operacji, ile będzie konieczne do uzgodnienia wymagań ze wszystkimi zainteresowanymi. Po sporządzeniu wymagań można przystąpić do budowy prototypu.

Jak to zrobić...

Należy przygotować dokument obejmujący następujące aspekty:

- **Projekt bazy danych:** należy zaplanować projekt bazy danych.
 - Należy obliczyć początkowy rozmiar bazy danych.
- **Analiza transakcji:** jak użytkownicy będą uzyskiwali dostęp do bazy danych?
 - Należy przeanalizować najczęściej używane ścieżki dostępu.
 - Jakie są wymagania odnośnie czasów odpowiedzi?

- **Konfiguracja sprzętowa**
 - Wstępna analiza wydajności — czy wszystkie dane zmieszczą się w pamięci operacyjnej?
- **Plan lokalizacji**
 - Należy wybrać obowiązujące na serwerze schemat kodowania, ustawienia regionalne i strefę czasową.
- **Plan dostępu i zabezpieczeń**
 - Należy zidentyfikować systemy klienckie i wskazać wymagane sterowniki.
 - Należy utworzyć role zgodnie z przyjętym wcześniej planem kontroli dostępu.
 - Należy przygotować plik *pg_hba.conf*.
- **Plan konserwacji:** kto będzie dbał o bieżące funkcjonowanie bazy danych? Jak będzie to robił?
- **Plan dostępności:** należy przemyśleć wymagania w zakresie dostępności.
 - Należy określić parametr *checkpoint_timeout*.
 - Należy zaplanować mechanizm tworzenia kopii zapasowych i poddać je testom.
- **Plan wysokiej dostępności**
 - Należy zdecydować, czy będzie konieczna replikacja, a jeśli tak — w jakiej formie.

Jak to działa...

Jednym z najważniejszych powodów planowania bazy danych z wyprzedzeniem są utrudnienia związane ze zmianą pewnych aspektów na późniejszych etapach pracy. Problem jest szczególnie dotkliwy w przypadku schematów kodowania i ustawień regionalnych, których zmiana już po wdrożeniu bazy danych może nie tylko wymagać sporych nakładów pracy, ale też wiązać się z koniecznością długiego przestoju. Także konfigurowanie zabezpieczeń po wdrożeniu systemu jest nieporównanie trudniejsze niż definiowanie odpowiednich ustawień z wyprzedzeniem.

To nie wszystko...

Planowanie zawsze jest pomocne. O ile sami doskonale wiemy, co robimy, nasi współpracownicy mogą nie dysponować podobnymi informacjami. Należy więc wszystkim wyjaśnić nasze zamiary, aby nie tracić czasu w przyszłości. Jeśli nie jesteśmy pewni słuszności jakiegoś założenia, powinniśmy zbudować prototyp, który ułatwi nam podjęcie ostatecznej decyzji — szkielet rozwiązań administracyjnych powinien mieć priorytet nie mniejszy niż właściwe czynności. Warto też przygotować listę decyzji do rozstrzygnięcia i kolejno pracować nad jej pozycjami.

Ta receptura celowo jest bardzo krótka. Każdy ma własny sposób osiągania wyznaczonych celów, zatem niezwykle ważne jest unikanie precyzyjnego opisywania, jak zrealizować poszczególne zadania. Jeśli czytelnik dysponuje już planem, to świetnie. Jeśli nie, należy zastanowić się, co należy zrobić, sporządzić listę i wreszcie wykonać zaplanowane zadania punkt po punkcie.

Zmiana parametrów na poziomie programów

System PostgreSQL umożliwia ustawianie niektórych parametrów osobno dla każdej sesji lub każdej transakcji.

Jak to zrobić...

Wartość parametru można zmienić w czasie trwania sesji, na przykład za pomocą następującego wyrażenia:

```
SET work_mem = '16MB';
```

W takim przypadku użyta wartość będzie obowiązywała dla każdej kolejnej transakcji. Istnieje też możliwość zmiany ustawień tylko na czas trwania bieżącej transakcji:

```
SET LOCAL work_mem = '16MB';
```

Tak zdefiniowane ustawienie będzie obowiązywało tylko do momentu użycia następującego wyrażenia:

```
RESET work_mem;
```

lub

```
RESET ALL;
```

Wyrażenia SET i RESET to polecenia języka SQL, które można stosować za pośrednictwem dowolnego interfejsu. Oba wyrażenia mogą być używane tylko dla parametrów serwera PostgreSQL, czyli parametrów wpływających na funkcjonowanie tego serwera (choć niekoniecznie na cały serwer). W systemie mogą występować inne parametry, na przykład parametry sterownika JDBC, których nie można ustawiać w ten sposób. Więcej informacji na ten temat można znaleźć w rozdziale „Kontrola serwera”.

Jak to działa...

Zmiana wartości parametru w czasie trwania sesji, na przykład za pomocą następującego wyrażenia:

```
SET work_mem = '16MB';
```

powoduje, że w perspektywie katalogowej `pg_settings` są zapisywane następujące dane:

```
postgres=# SELECT name, setting, reset_val, source
              FROM pg_settings WHERE source = 'session';
 name | setting | reset_val | source
-----+-----+-----+-----
work_mem | 16384 | 1024 | session
```

Przytoczone ustawienia obowiązują do momentu użycia następującego polecenia:

```
RESET work_mem;
```

Powyższe wyrażenie powoduje przywrócenie wartości `reset_val` i — tym samym — ponowne umieszczenie w kolumnie `source` wartości domyślnej.

```
 name | setting | reset_val | source
-----+-----+-----+-----
work_mem | 1024 | 1024 | default
```

To nie wszystko...

Wartość parametru można zmienić także w czasie trwania transakcji — wystarczy użyć wyrażenia w postaci:

```
SET LOCAL work_mem = '16MB';
```

Powyższe polecenie wprowadza następującą zmianę w perspektywie katalogowej `pg_settings`:

```
postgres=# SELECT name, setting, reset_val
              FROM pg_settings WHERE source = 'session';
 name | setting | reset_val | source
-----+-----+-----+-----
work_mem | 1024 | 1024 | session
```

Co to? Co stało się z użytą wartością parametru? Okazuje się, że wyrażenie `SET LOCAL` wpływa tylko na transakcję, w ramach której jest wykonywane, czyli w naszym przypadku właśnie na polecenie `SET LOCAL`. Musimy więc wykonać to polecenie w ramach bloku transakcji, aby wprowadzona zmiana była uwzględniana w kolejnych operacjach składających się na tę transakcję:

```
BEGIN;
SET LOCAL work_mem = '16MB';
```

Efekt wykonania tych wyrażeń widać teraz w perspektywie katalogowej `pg_settings`:

```
postgres=# SELECT name, setting, reset_val, source
              FROM pg_settings WHERE source = 'session';
 name | setting | reset_val | source
-----+-----+-----+-----
work_mem | 16384 | 1024 | session
```

Warto też zwrócić uwagę na to, że w kolumnie `source` widnieje wartość `session`, nie wartość `transaction`, która w tych okolicznościach wydawałaby się bardziej naturalna.

Jakie są bieżące ustawienia konfiguracyjne?

W pewnym momencie każdy z nas zada sobie pytanie „Jakie są **bieżące** ustawienia konfiguracyjne?”.

Jak to zrobić...

W pierwszym odruchu większość użytkowników odpowiada: „wystarczy zajrzeć do pliku `postgresql.conf`”. To rozwiązanie zdaje egzamin pod warunkiem, że dysponujemy tylko jednym plikiem z parametrami. Jeśli istnieją dwa takie pliki, może okazać się, że czytamy niewłaściwy plik! (Skąd właściwie mielibyśmy wiedzieć, który jest właściwy)? W tej sytuacji najbezpieczniejszym sposobem jest rezygnacja z przeglądania plików tekstowych na rzecz odwołania się do samego serwera.

Z podrozdziału „Zmiana parametrów na poziomie programów” wiemy też, że każdy parametr ma określony zasięg, decydujący o tym, kiedy jego wartość może być skutecznie zmieniana. O ile niektóre parametry można ustawiać za pośrednictwem pliku `postgresql.conf`, o tyle pozostałe podlegają zmianom już po załadowaniu zawartości tego pliku. Oznacza to, że bieżący stan ustawień konfiguracyjnych może być inny, niż to wynika z zapisów we wspomnianym pliku.

Do uzyskania obowiązujących ustawień można użyć polecenia `SHOW`, na przykład w tej formie:

```
postgres=# SHOW work_mem;
work_mem
-----
1MB
(1 row)
```

Warto jednak pamiętać, że polecenie `SHOW` prezentuje wartość wskazanego parametru obowiązującą w momencie wywołania i że ta wartość może być w tym czasie zmieniana.

Innym sposobem uzyskiwania bieżących ustawień jest dostęp do perspektywy katalogowej systemu PostgreSQL, nazwanej `pg_settings`:

```

postgres=# \x
Expanded display is on.
postgres=# SELECT * FROM pg_settings WHERE name = 'work_mem';
[ RECORD 1 ] -----
name          | work_mem
setting       | 1024
unit          | kB
category      | Resource Usage / Memory
short_desc    | Sets the maximum memory to be used for query workspaces.
extra_desc    | This much memory can be used by each internal sort operation and
              | ↪ hash table before switching to temporary disk files.
context       | user
vartype       | integer
source        | default
min_val       | 64
max_val       | 2147483647
enumvals      |
boot_val      | 1024
reset_val     | 1024
sourcefile    |
sourceline    |

```

Oznacza to, że polecenia `SHOW` można używać zarówno do uzyskiwania wybranych ustawień, jak i do uzyskiwania dostępu do pełnego wykazu szczegółów za pośrednictwem odpowiedniej tabeli katalogowej.

Jak to działa...

Każdy parametr jest buforowany w ramach każdej sesji, dzięki czemu dysponujemy błyskawicznym dostępem do ustawień reprezentowanych przez poszczególne parametry. Zastosowany model powoduje, że odczytywanie ustawień zapisanych w parametrach jest dziecinnie proste.

Warto pamiętać, że wyświetlane wartości nie zawsze odzwierciedlają ustawienia dotyczące całego serwera — wiele spośród tych parametrów jest ściśle związanych z bieżącą sesją. To odróżnia system PostgreSQL od wielu innych baz danych i w dużej mierze decyduje o jego elastyczności.

Które parametry zawierają wartości inne niż ustawienia domyślne?

Często musimy sprawdzić, które parametry zostały już zmienione lub czy nasze zmiany zostały prawidłowo uwzględnione w ustawieniach systemu PostgreSQL.

Jak to zrobić...

```
postgres=# SELECT name, source, setting
           FROM pg_settings
           WHERE source != 'default'
             AND source != 'override'
           ORDER by 2, 1;
```

name	source	setting
application_name	client	psql
log_timezone	command line	PL
TimeZone	command line	PL
timezone_abbreviations	command line	Default
archive_command	configuration file	(disabled)
archive_mode	configuration file	off
archive_timeout	configuration file	5
bgwriter_delay	configuration file	10
checkpoint_timeout	configuration file	30
log_checkpoints	configuration file	on
log_destination	configuration file	stderr
log_filename	configuration file	log%Y
logging_collector	configuration file	on
log_line_prefix	configuration file	%t[%p]
log_min_messages	configuration file	log
max_prepared_transactions	configuration file	5
max_standby_delay	configuration file	90
port	configuration file	5443
max_stack_depth	environment variable	2048
work_mem	session	204800
(29 rows)		

(Wartość `override` wykluczono z wyników dla poprawy czytelności).

Jak to działa...

Na podstawie danych widocznych w perspektywie `pg_settings` można bez trudu stwierdzić, które wartości są różne od ustawień domyślnych i jakie jest źródło bieżących wartości.

Polecenie `SHOW` nie określa, czy poszczególne parametry mają przypisane wartości domyślne. Działanie tego polecenia ogranicza się do zwrócenia interesującej nas wartości, co w żaden sposób nie rozstrzyga, co i dlaczego zostało zmienione.

Jeśli źródłem zmiany są zapisy w pliku konfiguracyjnym, wartości w kolumnach `sourcefile` i `sourceline` także są ustawione. Analiza tych wartości bywa przydatna w procesie identyfikacji źródeł poszczególnych ustawień konfiguracyjnych.

To nie wszystko...

Kolumna `setting` perspektywy `pg_settings` zawiera co prawda bieżącą wartość parametru, jednak warto zwrócić uwagę na kolumny `boot_val` i `reset_val`, które zawierają odpowiednio wartość ustawioną na etapie inicjalizacji klastra baz danych PostgreSQL ("`initdb`") oraz wartość, która zostanie przywrócona w danym parametrze wskutek użycia komendy `RESET`.

Kto to ustawił?

Parametr `max_stack_depth` jest wyjątkowy — z danych prezentowanych przez perspektywę `pg_settings` wynika, że jego wartość została ustawiona przez zmienną środowiskową, mimo że w rzeczywistości została zmieniona przez polecenie `ulimit -s` (w systemach Linux i Unix). Tylko w systemie Windows parametr `max_stack_depth` wymaga bezpośredniego ustawienia.

Także ustawienia strefy czasowej są określane na podstawie środowiska systemu operacyjnego, zatem ich bezpośrednie ustawianie nie jest konieczne. Co ciekawe, z danych prezentowanych przez perspektywę `pg_settings` wynika, że są to ustawienia definiowane z poziomu wiersza poleceń.

Aktualizacja pliku parametrów

Podstawowym miejscem definiowania wartości parametrów na potrzeby serwera PostgreSQL jest plik parametrów. W pliku parametrów nazwanym `postgresql.conf` można ustawić wszystkie parametry tego serwera.

Istnieją też dwa inne pliki parametrów, nazwane `pg_hba.conf` i `pg_ident.conf`. Zapisy zawarte w obu tych plikach mają związek z zarządzaniem połączeniami i zabezpieczeniami, zatem wrócimy do nich w dalszych rozdziałach poświęconych tym zagadnieniom.

Przygotowania

W pierwszym kroku należy znaleźć plik `postgresql.conf` (odpowiednią procedurę opisano już wcześniej).

Jak to zrobić...

Wszystkie parametry można ustawić w pliku parametrów nazwanym `postgresql.conf`. Niektóre spośród tych parametrów są uwzględniane tylko podczas pierwszego uruchamiania serwera. Typowym przykładem jest parametr `shared_buffers`, definiujący rozmiar współdzielonej pamięci podręcznej.

Wiele innych parametrów można zmienić w czasie działania serwera. Po zmianie wybranych parametrów należy wykonać na serwerze operację reload, aby wymusić na systemie PostgreSQL ponowne załadowanie pliku *postgresql.conf*:

```
pg_ctl -D data reload
```

Plik konfiguracyjny *postgresql.conf* ma postać normalnego pliku tekstowego, którego zawartość można bez trudu edytować. Większość parametrów od początku jest zapisana w tym pliku, zatem nasza rola sprowadza się do ich wyszukania i zastąpienia dotychczasowych ustawień własnymi wartościami.

Jak to działa...

W przypadku dwukrotnego ustawienia tego samego parametru w dwóch różnych miejscach pliku konfiguracyjnego, zostanie uwzględnione ostatnie ustawienie. Taka sytuacja może rodzić sporo nieporozumień, jeśli na przykład dopisujemy ustawienia na końcu pliku, zatem należy tego unikać.

Zalecaną praktyką jest pozostawienie oryginalnej struktury pliku konfiguracyjnego i sama edycja wybranych wartości lub wręcz rozpoczęcie pracy od pustego pliku i umieszczenie w nim tylko tych parametrów, których wartości chcemy zmienić. Sam preferuję model polegający na stosowaniu plików złożonych z wartości innych niż domyślne. Takie rozwiązanie ułatwia ocenę sytuacji.

Niezależnie od wybranej metody zaleca się zapisywanie kopii starszych wersji plików *.conf*. Wystarczy albo ręcznie skopiować pliki przeznaczone do modyfikacji, albo posłużyć się systemem kontroli wersji, na przykład systemem SVN.

To nie wszystko...

W pliku *postgresql.conf* można też stosować dyrektywę `include`. Oznacza to, że plik *postgresql.conf* można skonstruować w taki sposób, aby odwoływał się do innych plików, które z kolei mogą odwoływać się do jeszcze innych plików itd. Takie rozwiązanie może nam ułatwić lepszą organizację parametrów, o ile nie zastosujemy zbyt skomplikowanej struktury.

Ustawianie parametrów dla konkretnych grup użytkowników

System PostgreSQL obsługuje wiele różnych sposobów definiowania wartości parametrów dla poszczególnych grup użytkowników.

Jak to zrobić...

Ustawienia dla wszystkich użytkowników bazy danych saas można ustawiać za pomocą wyrażenia:

```
ALTER DATABASE saas
SET configuration_parameter = value1;
```

Ustawienia dla użytkownika simon połączonego z dowolną bazą danych można ustawić za pomocą wyrażenia:

```
ALTER ROLE simon
SET configuration_parameter = value2;
```

Istnieje też możliwość ustawienia parametru dla konkretnego użytkownika połączonego z określoną bazą danych:

```
ALTER ROLE simon
IN DATABASE saas
SET configuration_parameter = value3;
```

Sam użytkownik nie wie, że specjalnie z myślą o nim wykonano tego rodzaju wyrażenie. Tak zdefiniowane ustawienia w większości przypadków pełnią funkcję wartości domyślnych i jako takie mogą być zmieniane przez użytkownika potrzebującego innych wartości parametrów.

Jak to działa...

Parametry można ustawiać na następujących poziomach:

- bazy danych;
- użytkownika (w systemie PostgreSQL użytkownika określa się mianem roli);
- kombinacji bazy danych i użytkownika.

Wartość domyślna tych parametrów są nadpisywane wartościami definiowanymi na kolejnych poziomach powyższej listy.

Oznacza to, że w przypadku wykonania powyższych trzech wyrażeń języka SQL będą obowiązywały następujące ustawienia:

- użytkownik hannu łączy się z bazą danych saas — zastosowanie ma wartość value1;
- użytkownik simon łączy się z bazą danych inną niż saas — zastosowanie ma wartość value2;
- użytkownik simon łączy się z bazą danych saas — zastosowanie ma wartość value3.

System PostgreSQL implementuje te reguły dokładnie tak, jakby użytkownik ręcznie użył odpowiednich wyrażeń SET bezpośrednio po nawiązaniu połączenia.

Lista podstawowych zadań związanych z konfiguracją serwera

System PostgreSQL domyślnie jest skonfigurowany pod kątem pracy w roli systemu współdzielonego, jednak wielu użytkowników woli dysponować dedykowanymi systemami baz danych. Jednym z celów projektu PostgreSQL jest zapewnienie możliwości zgodnego współistnienia tego systemu z innymi programami działającymi po stronie serwera, zatem wykorzystywanie pełnych zasobów serwera byłoby niepożądane. Jeśli jednak administrator systemu wie, że w tym samym systemie nie będą działały inne ważne serwery, może bez obaw zmienić parametry serwera PostgreSQL na wyższe.

Przygotowania

Zanim przystąpimy do modyfikowania parametrów systemu, musimy dysponować dwiema informacjami:

Po pierwsze, musimy znać wielkość fizycznej pamięci RAM przeznaczonej dla systemu PostgreSQL.

Po drugie, musimy wiedzieć, jakiego rodzaju aplikacje będą używały systemu PostgreSQL.

Jak to zrobić...

Jeśli rozmiar bazy danych przekracza 32 MB, zwiększenie wartości parametru `shared_buffers` powinno podnieść wydajność systemu. Teoretycznie można tę wartość zwielokrotnić, jednak musimy mieć na uwadze, że w systemach Linux nieużywany bufor może trafić do pliku wymiany, zatem powinniśmy zachować ostrożność. Nową wartość można ustawić w pliku `postgresql.conf` i stopniowo zwiększać, każdorazowo sprawdzając, czy zmiana przyniosła pożądany efekt.

W przypadku zwiększenia wartości parametru `shared_buffers` na serwerze z systemem operacyjnym innym niż Windows niemal zawsze należy dodatkowo zwiększyć wartość parametru `SHMMAX` samego systemu operacyjnego (na niektórych platformach możemy stanąć przez koniecznością zmiany także innych parametrów).

W systemach Linux, Mac OS i FreeBSD należy albo zmienić zawartość pliku `/etc/sysctl.conf`, albo użyć polecenia `sysctl -w` z następującymi wartościami:

- Linux: `kernel.shmmax=wartość`
- Mac OS: `kern.sysv.shmmax=wartość`
- FreeBSD: `kern.ipc.shmmax=wartość`

Więcej informacji na temat tych wartości można znaleźć na stronie <http://www.postgresql.org/docs/8.4/static/kernel-resources.html#SYSVIPC>.

Na przykład w systemie Linux należałoby dodać do pliku */etc/sysctl.conf* następujący wiersz:

```
kernel.shmmax=wartość
```

Wartość parametru *effective_cache_size* nie jest taka ważna. Wbrew pozorom ma dużo mniejsze znaczenie niż parametr *shared_buffers*, zatem wybór właściwej wartości nie powinien nam zająć zbyt wiele czasu.

Jeśli w naszym systemie planujemy wykonywanie wielu operacji zapisu, powinniśmy rozważyć znaczne zwiększenie wartości parametru *wal_buffers* względem wartości domyślnej.

Jeśli planujemy dużą liczbę operacji zapisu i (lub) operowanie na wielkich ilościach danych, być może powinniśmy zwiększyć wartość parametru *checkpoint_segments* względem ustawień domyślnych.

Jeśli na bazie danych są wykonywane wielkie zapytania, powinniśmy rozważyć zmianę domyślnej wartości parametru *work_mem* na większą.

Warto upewnić się, że parametr *autovacuum* jest włączony, chyba że z jakiegoś powodu świadomie podjęliśmy decyzję o jego wyłączeniu. W zdecydowanej większości przypadków wspomniany parametr powinien jednak być włączony (więcej informacji na ten temat można znaleźć w dalszej części tej książki).

Sporym uproszczeniem jest użycie narzędzia dostępnego pod następującym adresem URL: <http://pgfoundry.org/projects/pgtune/>.

Na tym możemy zakończyć proces wstępnego ustawiania parametrów. Nie powinniśmy poświęcać zbyt dużo czasu na poszukiwanie optymalnych ustawień. Większość tych parametrów można zmienić w przyszłości, zatem warto zastosować model przyrostowego doskonalenia poszczególnych wartości.

Początkowo należy koncentrować uwagę przede wszystkim na najważniejszych parametrach i zadbać o ich prawidłowe ustawienia. Po opanowaniu podstaw warto zaopatrzyć się w książkę Grega Smitha poświęconą wydajności systemu PostgreSQL.

W szczególności nie należy zmieniać parametru *fsync*, który zapewnia bezpieczeństwo naszemu systemowi.

Dodawanie modułu zewnętrznego do systemu PostgreSQL

Jedną z zalet systemu PostgreSQL jest jego rozszerzalność. Właśnie rozszerzalność była jednym z oryginalnych celów projektowych stawianych sobie przez twórców tego systemu już w latach osiemdziesiątych ubiegłego wieku. Dla współczesnej wersji PostgreSQL 9.0 istnieje wiele dodatkowych modułów, które można łatwo dołączać do podstawowego serwera PostgreSQL.

Dodatkowe moduły mogą wprowadzać do rozszerzanego systemu udoskonalenia różnych typów:

- dodatkowe funkcje,
- dodatkowe typy danych,
- dodatkowe operatory,
- dodatkowe indeksy.

Co ciekawe, wiele narzędzi i interfejsów klienckich współpracuje z systemem PostgreSQL bez konieczności jakiegokolwiek specjalnej instalacji. W tym podrozdziale omówimy moduły rozszerzające lub zmieniające zachowanie serwera, w tym standardową składnię, funkcje i zachowania wyrażen języka SQL.

Przygotowania

Musimy najpierw wybrać właściwy moduł do zainstalowania.

Ponieważ do tej pory nie powstał system zarządzania pakietami dla projektu PostgreSQL, dostępne moduły znajdują się w wielu różnych miejscach, w tym na następujących witrynach internetowych:

- **Contrib** — tzw. jądro systemu PostgreSQL — zawiera wiele przydatnych funkcji. Okazuje się jednak, że istnieje też oficjalna sekcja modułów dodatkowych, określanych mianem modułów contrib. Ich dokumentację można znaleźć pod następującym adresem URL:
 - <http://www.postgresql.org/docs/9.0/static/contrib-dblink-connect.html>
- **pgFoundry** — witryna internetowa tego projektu open source ma na celu przede wszystkim udostępnianie modułów i narzędzi dla systemu PostgreSQL. Witryna PgFoundry korzysta z tego samego oprogramowania co popularny serwis *SourceForge.net*. Warto więc zajrzeć pod następujący adres URL:
 - <http://pgFoundry.org/>
- **Odrębne projekty** — wielkie projekty zewnętrzne, na przykład PostGIS, oferują rozbudowane, złożone moduły dodatkowe dla systemu PostgreSQL. Zachęcam do odwiedzenia następującego adresu URL:
 - <http://www.postgis.org/>

Jak to zrobić...

W pewnych przypadkach moduły można dodawać już na etapie instalacji, jeśli korzystamy z autonomicznej aplikacji instalacyjnej, na przykład z programu instalatora **OneClick**.

W pozostałych przypadkach istnieje możliwość instalacji modułu z pakietu, na przykład modułu zapewniającego zgodność z systemem Oracle (http://www.postgres.cz/index.php/Oracle_functionality).

W pierwszym kroku należy pobrać interesujący nas pakiet, na przykład: <http://pgfoundry.org/frs/download.php/2420/orafce-3.0.1-1.pg82.rhel5.i386.rpm>.

Pobrany pakiet należy zainstalować, stosując następujące polecenia:

```
rpm -ivh orafce-3.0.1-1.pg90.rhe15.i386.rpm
sudo apt-get install postgresql-8.4-orafce
```

W wielu przypadkach przydatne moduły nie są udostępniane w formie gotowych pakietów. Takie moduły należy instalować ręcznie. Instalacja nie jest trudna i może być ciekawym ćwiczeniem ułatwiającym zrozumienie odpowiednich procedur.

Każdy moduł wymaga nieco innej instalacji. Ogólnie należy zwracać szczególną uwagę na następujące dwa aspekty instalacji modułu:

- instalację obiektów języka SQL dla danego modułu;
- instalacje wymaganych przez ten moduł bibliotek ładowanych dynamicznie (DLL).

Instalacja większości przydatnych modułów wymaga od użytkownika zadbania o oba wymienione powyżej aspekty. Istnieją też moduły, na przykład AutoExplain, które korzystają wyłącznie z bibliotek ładowanych dynamicznie.

- Należy skompilować odpowiednie biblioteki.

Należy postępować według instrukcji dla danego modułu:

- Należy zainstalować bibliotekę w miejscu, w którym będzie dostępna dla serwera:

```
shared_preload_libraries = '$libdir/modlib'
```

Należy utworzyć obiekty bazy danych:

```
psql -d dbname -f SHAREDIR/contrib/module.sql
```

Jak to działa...

System PostgreSQL może dynamicznie ładować biblioteki na trzy sposoby:

- skutek wydania przez użytkownika bezpośredniego polecenia LOAD w ramach sesji;

- na podstawie parametru `shared_preload_libraries` zapisanego w pliku `postgresql.conf` i używanego podczas uruchamiania serwera;
- na początku sesji na podstawie parametru `local_preload_libraries` ustawionego dla konkretnego użytkownika (zdefiniowanego przy użyciu polecenia `ALTER ROLE`).

Funkcje i obiekty systemu PostgreSQL mogą odwoływać się do kodu zawartego w tych bibliotekach — takie rozwiązanie umożliwia ściśle wiązanie rozszerzeń z działającym procesem serwera. Tak ściśle związki powodują, że opisana metoda sprawdza się nawet w aplikacjach, które muszą spełniać najwyższe wymagania w zakresie wydajności. Dodatkowe opcje i funkcje nie odbiegają wydajnością od rdzennych elementów systemu.

Uruchamianie serwera w trybie oszczędzania energii

Zużycie energii elektrycznej jest obecnie jednym z najczęściej dyskutowanych tematów. Dziś każdy szuka sposobów, by choć w minimalnym stopniu przyczynić się do ochrony środowiska naturalnego. Nie inaczej jest w przypadku użytkowników systemu PostgreSQL.

Przygotowania

Jeśli serwer PostgreSQL ma być używany sporadycznie lub pozostaje nieaktywny przez dłuższe okresy, warto rozważyć zastosowanie przynajmniej części z poniższych sugestii. Taki serwer może działać na laptopie lub mieć postać nieaktywnego serwera wirtualnego.

Jak to zrobić...

System PostgreSQL to przykład bazy danych działającej po stronie serwera, zatem w czasie, gdy nie ma aktywnych klientów, serwer w praktyce pozostaje bezczynny. Aby zminimalizować aktywność serwera, wystarczy ustawić następujące parametry w pliku `postgresql.conf`:

- `autovacuum = off`
- `wal_writer_delay = 10000`
- `bgwriter_delay = 10000`

Przytoczone ustawienia w wielu zastosowaniach są dalece nieoptymalne i jako takie powinny być stosowane tylko dla serwera, o którym wiadomo, że przez znaczną część czasu będzie nieaktywny. Kiedy obciążenie serwera ponownie wzrośnie, należy przywrócić w powyższych parametrach poprzednie wartości.

Jak to działa...

Istnieje kilka procesów, które stale pozostają aktywne w oczekiwaniu na aktywnych klientów, których żądania będą wymagały efektywnej obsługi. W systemie PostgreSQL istnieje pięć takich procesów:

- proces zapisujący (znany też jako proces zapisujący w tle);
- proces zapisujący dziennik WAL;
- archiwizator (aktywny tylko wtedy, gdy archiwizacja dziennika WAL jest włączona);
- proces odbiorcy dziennika WAL (aktywny tylko wtedy, gdy jest stosowany mechanizm replikacji strumieniowej);
- proces automatycznego czyszczenia (ang. *autovacuum*).

Proces zapisujący w tle domyślnie jest aktywowany co 200 ms. Wartość maksymalna dzieląca kolejne próby wybudzania tego procesu wynosi 10 s, co nie wydaje się szczególnie długim czasem bezczynności. Okazuje się jednak, że można ten proces całkowicie wyłączyć, przypisując parametrowi `bgwriter_lru_maxpages` wartość 0.

Proces zapisujący w dzienniku WAL domyślnie jest aktywowany co 200 ms. Także w tym przypadku maksymalna wartość odpowiedniego parametru to 10 s. Tego procesu nie można jednak wyłączyć. W przypadku braku operacji zapisu proces nie podejmuje żadnych działań — jego działanie sprowadza się do wybudzania i sprawdzania, czy istnieją ewentualne operacje oczekujące.

Proces archiwizatora (ang. *archiver*) jest aktywowany co 15 sekund i sprawdza, czy zapisano jakieś nowe pliki dziennika WAL. Każda aktywacja powoduje wykonanie operacji na katalogu systemu plików. Okresu dzielącego kolejne wybudzanie tego procesu nie można zmienić za pośrednictwem żadnego parametru.

Proces odbiorcy dziennika WAL jest aktywowany co 100 ms w celu sprawdzenia, czy do serwera nie dotarły nowe dane replikacji. W razie braku takich danych proces ponownie przechodzi w stan uśpienia. Okresu dzielącego kolejne wybudzenia tego procesu nie można zmienić za pośrednictwem żadnego parametru.

Proces automatycznego czyszczenia domyślnie jest aktywowany co minutę. Częstotliwość wybudzania tego procesu można zmienić, przypisując nową wartość parametrowi `autovacuum_naptime`. Proces automatycznego czyszczenia można całkowicie wyłączyć, stosując parametr `autovacuum = off`.

Jeśli stosujemy mechanizm replikacji strumieniowej, nasz serwer będzie aktywowany co 200 ms. Jeśli nie korzystamy z tego mechanizmu, możemy wydłużyć czas pomiędzy kolejnymi wybudzeniami do 10 sekund (zamiast domyślnych 200 ms).

Skorowidz

\$BACKUPNAME, 322, 323, 326
\$OTHERNODE, 325, 326
\$PGARCHIVE, 357
\$PGDATA, 322, 326
\$STANDBYNODE, 357
.pgpass, 41
::, 140
2PC, 235, 262

A

AccessExclusiveLock, 68, 193, 207, 271
ADD COLUMN, 193
administracja bazą danych, 181
aktualizacja perspektyw, 214
dodawanie kolumn tabeli, 192
dodawanie przestrzeni tabel, 201
dodawanie schematów, 198
dostęp do obiektów należących do innych baz danych PostgreSQL, 208
dynamiczne generowanie skryptów, 190
operacje na wielu tabelach, 187
pg_batch, 192
przenoszenie obiektów między przestrzeniami tabel, 205
przenoszenie obiektów między schematami, 200
reguły, 218
równoległe wykonywanie zadań, 190
skrypty, 183
usuwanie kolumn tabeli, 192
usuwanie przestrzeni tabel, 201
usuwanie schematów, 198
zmiana typu danych kolumny, 195
administratorzy, 152
adres IP serwera, 27, 133
aktualizacja HOT, 301
aktualizacja perspektyw, 214
aktualizacja pliku parametrów, 84
aktualizacje, 301, 389
aktualizacje do wersji głównych w sieci przy użyciu narzędzi do replikacji, 393
aktualizacje działającego serwera do wersji głównych, 390
aktualizacje online, 393
pg_upgrade, 390
proces, 389
algorytm MD5, 162
ALTER DATABASE, 86, 207
ALTER DATABASE CONNECTION LIMIT, 101
ALTER DATABASE SET TABLESPACE, 207
ALTER DEFAULT PRIVILEGES, 110, 200
ALTER INDEX RENAME, 119
ALTER INDEX SET TABLESPACE, 206
ALTER ROLE, 86, 91, 159
ALTER ROLE CONNECTION LIMIT, 103
ALTER SCHEMA RENAME TO, 200
ALTER TABLE, 68, 132, 191, 193, 266, 372
ALTER TABLE ADD COLUMN, 193
ALTER TABLE ADD EXCLUDE, 132
ALTER TABLE ADD FOREIGN KEY, 72
ALTER TABLE CASCADE, 194
ALTER TABLE DROP COLUMN, 193
ALTER TABLE SET OWNER, 168
ALTER TABLE SET TABLESPACE, 206
ALTER TABLESPACE OWNER, 204
ALTER TABLESPACE SET, 205
ALTER USER PASSWORD, 40
analitka biznesowa, 351
analiza bazy danych, 246
analiza danych historycznych, 223
analiza dat modyfikacji plików, 238
analiza pliku dziennika serwera, 279
analiza transakcji, 77
analiza wydajności, 279
ANALYZE, 135, 137, 252, 254, 279
anulowanie zapytania, 233
apply delay, 352
architektura bez dzielenia, 354
architektura OFA, 52
archive_cleanup_command, 359, 360
archive_command, 323, 326, 327, 358, 365, 366
archive_mode, 323, 324, 326, 365

archive_timeout, 327, 358
 archiving_active, 324
 archiwizacja ciągła, 325
 arkusz kalkulacyjny, 143
 ataki, 279
 DoS, 102
 automatyczna konserwacja bazy danych, 252
 automatyczne czyszczenie, 252
 tabele TOAST, 255
 automatyzacja zadań, 182
 autonomiczna, fizyczna kopia zapasowa bazy danych
 tworzona w czasie rzeczywistym, 322
 autovacuum, 88, 91, 92, 253, 257
 autovacuum.conf.day, 257
 autovacuum.conf.night, 257
 autovacuum_analyze_scale_factor, 253, 254
 autovacuum_analyze_threshold, 253, 254
 autovacuum_enabled, 253
 autovacuum_freeze_max_age, 253, 257, 259
 autovacuum_freeze_min_age, 253
 autovacuum_freeze_table_age, 253
 autovacuum_max_workers, 253, 254, 257
 autovacuum_naptime, 92, 253, 254
 autovacuum_vacuum_cost_delay, 253, 254
 autovacuum_vacuum_cost_limit, 253, 254
 autovacuum_vacuum_scale_factor, 253, 254
 autovacuum_vacuum_threshold, 253, 254
 av_threshold, 268
 awarie, 310
 awaryjne zatrzymywanie serwera, 96

B

backup_label, 327, 331
 BACKUPNAME, 322
 badanie błędów, 246
 base, 52
 batches, 352
 baza danych, 19, 20, 25
 planowanie bazy danych, 77
 rozmiar, 63
 sekwencja układania, 60
 tabele, 60
 tworzenie, 59
 BEGIN, 129, 183, 185, 290
 bezpieczeństwo, 151
 hasła, 162
 szyfrowanie danych, 175
 bezpieczne zatrzymywanie serwera, 95
 bgwriter_delay, 91
 bgwriter_lru_maxpages, 92
 BI, 351
 bieżąca baza danych, 27
 bieżące ustawienia konfiguracyjne, 81
 blokady, 129, 193, 290
 AccessExclusiveLock, 68

bloki, 69
 bloki brudne, 100
 blokowanie aktualizacji perspektyw, 214
 blokowanie optymistyczne, 305
 błędy, 246
 Business Intelligence, 351

C

CA, 174
 Cacti, 222, 224, 243
 camel case, 118
 CASCADE, 153, 194, 199
 catch up period, 353
 cel odzyskiwania, 333
 CentOS, 51
 Certificate Authority, 174
 certyfikaty SSL, 174
 check_postgres, 224, 270
 check_postgres_bloat, 270
 checkpoint_segments, 88, 311
 checkpoint_timeout, 311, 358
 checkpoints, 311
 CIDR-ADDRESS, 28
 clean switchover, 394
 cluster, 351
 CLUSTER, 300, 351
 clusterware, 352
 clustname_londiste.ini, 376
 collation sequence, 60
 COMMIT, 129, 183, 185
 COMMIT PREPARED, 263
 CONNECT, 26
 CONNECTION LIMIT, 101, 103
 connection service file, 42
 connection_timeout, 364
 consume_balance(), 307
 Continuent Tungsten, 370
 contrib, 89, 209
 COPY, 145, 217
 copy_from(), 165
 count(*), 66
 CREATE, 184
 CREATE DATABASE, 58, 110, 111
 CREATE FOREIGN DATA WRAPPER, 209
 CREATE FUNCTION, 213
 CREATE GROUP, 158
 CREATE INDEX, 184, 202
 CREATE INDEX CONCURRENTLY, 272, 274, 277
 CREATE LANGUAGE, 164, 238
 CREATE MATERIALIZED VIEW AS, 298
 CREATE OR REPLACE, 184
 CREATE OR REPLACE SCHEMA, 199
 CREATE ROLE, 110, 157, 158
 CREATE SCHEMA, 108, 198, 199
 CREATE SCHEMA AUTHORIZATION, 198

CREATE TABLE, 132
 CREATE TABLE AS SELECT, 132
 CREATE TABLESPACE, 202
 CREATE TEMPORARY TABLE, 297
 CREATE TYPE, 238
 CREATE UNIQUE INDEX, 131, 273
 CREATE USER, 158
 CREATE VIEW, 214, 215
 createdb, 58
 CREATEROLE, 157, 158, 160
 CREATEUSER, 157
 cross-tab query, 294
 CSV, 145
 current_database(), 27
 current_setting(), 240
 current_user, 27
 czas działania serwera, 50
 czyste przełączanie, 394
 czyszczenie bazy, 252
 sesje autonomiczne, 261
 czyszczenie tabeli TOAST, 255

D

dane CSV, 145
 dane testowe, 137
 dane tymczasowe, 240, 242
 Data Definition Language, 166, 183
 database replication, 351
 daty modyfikacji plików, 238
 dblink, 170, 209, 210, 211
 dblink_connect(), 209
 dblink_disconnect(), 209
 dblink_get_connections(), 212
 dblink_get_result(), 213
 dblink_is_busy(), 213
 dblink_send_query(), 213
 DDL, 166, 183
 Debian, 51
 debugging_info_on(), 165
 debugging_info_reset(), 166
 decrypt_using_my_secret_key(), 177
 DEFAULT CURRENT_TIMESTAMP, 168
 default_statistics_target, 299, 300
 default_tablespace, 205
 DEFERRABLE, 132
 DELETE, 128, 276
 denial of service, 102
 diagnostyka funkcjonowania bazy danych, 221
 diff_table_definition(), 125
 długie zapytania, 283
 długoterminowa analiza wydajności, 279
 do_emp_audit(), 171
 dodawanie kolumn tabeli, 192
 dodawanie modułu zewnętrznego, 89
 dodawanie przestrzeni tabel, 201

dodawanie schematów, 198
 domyślna ścieżka wyszukiwania, 154
 DoS, 102
 dostęp do bazy danych PostgreSQL, 24
 dostęp do obiektów należących do innych baz danych PostgreSQL, 208
 dostęp do schematu, 156
 DROP, 184
 DROP COLUMN, 193, 372
 DROP IF EXISTS, 184
 DROP INDEX, 276
 DROP SCHEMA, 199
 DROP SCHEMA IF EXISTS, 199
 DROP SCHEMA IF EXISTS CASCADE, 199
 DROP TABLE, 72, 73
 DROP TABLESPACE, 202
 dynamiczne generowanie skryptów, 190
 dynamiczne ładowanie bibliotek, 90
 dyskowe operacje wejścia-wyjścia, 291
 dziennik komunikatów, 63
 dziennik serwera bazy danych, 53
 dziennik systemowy, 54
 dziennik WAL, 310, 358

E

EAI, 351
 effective_cache_size, 88
 eliminacja zbyt długo oczekujących zapytań, 233
 EMS SQLManager, 34
 enable_seqscan, 304
 encrypt_using_my_public_key(), 177, 178
 END_DATE, 134
 EPS, 142, 143
 equal probability of selection, 142
 estimated_row_count(), 69
 ETL, 351
 eventual consistency, 372, 379
 excl, 119
 EXCLUDE, 132
 EXECUTE, 285
 EXPLAIN, 294
 EXPLAIN ANALYSE, 287, 299

F

failover, 352, 368
 fan-in, 381
 fan-out, 381
 Fedora, 51
 FHS, 52
 Filesystem Hierarchy Standard, 52
 fillfactor, 269, 301
 fizyczna kopia zapasowa, 313, 322, 324
 odzyskiwanie, 329
 odzyskiwanie bazy danych, 341

fizyczna kopia zapasowa
 odzyskiwanie przestrzeni tabel, 339
 odzyskiwanie tabeli, 337
 tworzenie, 325

fizyczne odzyskiwanie bazy danych, 331

FOREIGN DATA WRAPPER, 209

FreeSpaceMap, 64

freezing, 258

FROM, 293

funkcje, 213

- copy_from(), 165
- current_setting(), 240
- dblink_get_connections(), 212
- dblink_get_result(), 213
- dblink_is_busy(), 213
- dblink_send_query(), 213
- funkcje zwracające zbiory, 298
- generate_series(), 140
- pg_cancel_backend(), 233
- pg_ls_dir(), 239
- pg_stat_file(), 239
- pg_stat_reset(), 236
- pg_stop_backup(), 324
- pg_terminate_backend(), 159, 232, 233
- pgstatindex(), 270
- pgstattuple(), 270
- quote_ident(), 121
- random(), 141, 142
- repeat(), 138
- substr(), 139
- table_file_info(), 239
- timeofday(), 133

G

generate_series(), 138, 140

generowanie

- codzienne podsumowania błędów zarejestrowanych w pliku dziennika, 248
- dane testowe, 137

GEQO, 297

get_my_public_key(), 177

global, 52

gorąca gotowość, 332, 370

graficzne narzędzia administracyjne, 29

GRANT, 111, 153, 156, 200

GRANT USAGE ON SCHEMA, 156

grep, 249

gromadzenie danych o zmianach zapisanych w dzienniku serwera, 169

gromadzenie statystyk, 237, 285

gromadzenie zmian przy użyciu wyzwalaczy, 169

- zapisywanie danych w innej bazie danych, 170

grupy uprawnień, 156

CSSAPI, 40

gzip, 344

H

harmonogram zadań, 33

hasła, 39, 162

- plik hasel, 41
- trwale zakodowane hasła, 40

HBA, 102

HEADER, 145

heap_blks_hit, 284

heap_blks_read, 289

help, 37

host, 25, 26

hostaddr, 26

host-based authentication, 102

hostname, 26

HOT, 267, 269, 301

hot standby, 332, 370

hot_standby, 44, 371

HOT_update_ratio, 269

hstore, 21

I

IANA, 26, 94

identyfikacja

- długie zapytania, 283
- powtarzające się wiersze, 126
- przyczyny wolnego działania wyrażen SQL, 287

identyfikator systemu bazy danych, 56

identyfikator transakcji, 261, 333

identyfikator użytkownika, 27

idx, 119

idx_blks_read, 284, 289

IF EXISTS, 184

ilość przetwarzanych danych, 287

immediate stop, 97

include, 256

incremental forever backup, 346

indeksy, 300, 302

- CLUSTER, 300
- indeksy TOAST, 204
- indeksy warunkowe, 300
- konserwacja, 271, 279
- nazwy, 119
- nieużywane indeksy, 275
- powtarzające się indeksy, 133
- przebudowa, 271
- UNIQUE INDEX, 130
- usuwanie niepotrzebnych indeksów, 276
- wymuszanie unikatowości, 132
- wymuszanie użycia, 303

inet_server_addr(), 27

inet_server_port(), 27

informacje o aktualnie wykonywanym zapytaniu, 226

informacje o ostatnim użyciu, 240

informacje o połączeniu, 27

informacje o systemie, 223

informacje o tabeli, 72
 Information Schema, 62
 information_schema, 65
 information_schema.tables, 61
 initdb, 113
 INSERT, 276
 instalacja modułów, 90
 instalator OneClick, 90
 INSTEAD OF, 219
 instead-of trigger, 216
 integracja z serwerem LDAP, 171
 integralność odwołań, 72
 Internet Assigned Numbers Authority, 26
 ip4r, 134
 izolacja migawki, 21

J

Java Transaction API, 262
 jądro systemu PostgreSQL, 89
 język DDL, 166, 183
 język PL/Proxy, 170
 język PL/PythonU, 238
 język SQL, 13
 JTA, 262

K

katalog danych, 51, 94, 112
 kernel.shmmax, 88
 key, 119
 kill, 99, 233, 234
 klaster, 26, 94, 351
 klient-serwer, 26
 klucz SSL, 174
 kodowanie hasła, 40
 kolumny, 122
 komentarze, 37
 kompresja kopii zapasowych, 344
 kompresja plików WAL, 344
 konfiguracja klienta pod kątem stosowania protokołu SSL, 174
 konfiguracja puli połączeń, 113
 konfiguracja serwera, 87
 konfiguracja sprzętowa, 78
 konflikty miękkie, 372
 konflikty twarde, 372
 konserwacja bazy danych, 251
 ANALYZE, 254
 automatyczna konserwacja, 252
 autovacuum, 253
 autovacuum.conf.day, 257
 autovacuum.conf.night, 257
 cykl czynności, 278
 HOT, 267, 269
 indeksy, 271, 279
 kopie zapasowe, 279

nieużywane indeksy, 275
 parametry przechowywania, 253
 planowanie konserwacji, 278
 planowanie pojemności, 278
 przekroczenie licznika transakcji, 260
 przerośnięte indeksy, 266
 przerośnięte tabele, 266
 tabele TOAST, 254, 255
 tabele tymczasowe, 265
 track_counts, 253
 unikanie automatycznego mrożenia, 258
 usuwanie niepotrzebnych indeksów, 276
 usuwanie przygotowanych transakcji, 262
 VACUUM, 254
 zmiana reguł, 278
 kontrola automatycznej konserwacji bazy danych, 252
 kontrola serwera, 93
 konwersja typu danych, 196
 kopie zapasowe, 279, 310
 archiwizacja ciągła, 325
 autonomiczna, fizyczna kopia zapasowa bazy danych
 tworzona w czasie rzeczywistym, 322
 backup_label, 327
 dziennik WAL, 310, 311
 fizyczna kopia zapasowa, 313, 322, 324, 325, 329
 kompresja, 344
 kopia zapasowa definicji obiektów bazy danych, 321
 logiczna kopia zapasowa, 313, 316, 328
 logiczna kopia zapasowa jednej bazy danych
 tworzona w czasie rzeczywistym, 316
 logiczna kopia zapasowa wszystkich baz danych
 tworzona w czasie rzeczywistym, 318
 logiczna kopia zapasowa wszystkich tabel
 w pojedynczej przestrzeni tabel tworzona w czasie
 rzeczywistym, 319
 odzyskiwanie bazy danych, 340
 odzyskiwanie do punktu w czasie, 332
 odzyskiwanie przestrzeni tabel, 338
 odzyskiwanie usuniętej lub uszkodzonej tabeli, 335
 odzyskiwanie wszystkich baz danych, 328
 pg_dump, 313, 317
 pg_dumpall, 313, 315, 318
 pg_restore, 315, 328
 pg_rman, 346
 pg_stop_backup(), 324
 planowanie tworzenia, 312
 przyrostowa kopia zapasowa, 345
 punkty kontrolne, 311
 różnicowa kopia zapasowa, 345
 strategia zawsze przyrostowej kopii zapasowej, 346
 tworzenie, 316
 wydajność, 341
 rzut definicji obiektów, 321
 rzut definicji przestrzeni tabel, 321
 rzut definicji ról, 321
 krotki, 69, 189

L

latency, 352
 LDAP, 40, 171
 libpq, 25
 licencja BSD, 22
 licencja systemu PostgreSQL, 22
 licencja TPL, 22
 liczba połączeń, 26, 101
 liczba tabel w bazie danych, 60
 liczba wierszy w tabeli, 65, 67
 liczba zwracanych wierszy, 291
 liczby losowe, 138
 licznik transakcji, 260
 Lightning Admin, 34
 Lightweight Directory Access Protocol, 171
 LIMIT, 244, 292
 limit czasowy wyrażen, 233
 limit połączeń, 103
 lista baz danych na danym serwerze, 57
 listen_addresses, 28, 29, 44
 local_preload_libraries, 91
 localhost, 26
 LOCK, 129, 290
 Log Sequence Number, 346
 log_autovacuum_min_duration, 253, 254
 log_connections, 44
 log_disconnections, 44
 log_duration, 283
 log_emp_audit(), 170, 171
 log_error_verbosity, 55
 log_filename, 167
 log_line_prefix, 55, 168, 285
 log_min_duration_statement, 283, 284
 log_min_messages, 55
 log_rotation_age, 167, 248
 log_rotation_size, 167
 log_statement, 166, 169
 logiczna kopia zapasowa, 313, 316

- logiczna kopia zapasowa wszystkich baz danych, 318
- logiczna kopia zapasowa wszystkich tabel w pojedynczej przestrzeni tabel, 319
- odzyskiwanie, 328
- odzyskiwanie bazy danych, 340
- odzyskiwanie przestrzeni tabel, 338
- odzyskiwanie tabeli, 335
- tworzenie, 316

 logiczne odzyskiwanie bazy danych, 331
 lokalizacja dziennika komunikatów, 53
 lokalizacja identyfikatora systemu bazy danych, 56
 lokalizacja plików serwera bazy danych, 51
 Londiste, 375

- awaryjne zatrzymanie, 379
- konfiguracja replikacji, 376
- monitorowanie opóźnienia replikacji, 380

ostateczna spójność, 379
 przerywanie pracy, 379
 schemat, 378
 lookup tables, 118
 losowe dane liczbowe, 138
 losowe próbkowanie danych, 141
 losowy dobór próby, 142
 LSN, 346

Ł

ładowanie bibliotek, 90
 ładowanie danych z arkusza kalkulacyjnego, 143
 ładowanie danych z plików, 146

M

maintenance_work_mem, 342, 343
 Man-In-The-Middle, 175
 mapa widoczności, 64
 mapa wolnej przestrzeni, 64
 max_prepared_transactions, 263
 max_stack_depth, 84
 max_Standby, 373
 max_Standby_archive_delay, 371, 372, 373
 max_Standby_streaming_delay, 371, 372, 373
 max_wal_senders, 362, 363, 364
 md5, 28
 MD5, 162
 mechanizm HOT, 269
 mechanizm MVCC, 66, 266, 267
 mechanizm odzyskiwania bazy danych po awarii, 310, 311
 mechanizm optymalizacji, 299
 mechanizm optymalizacji genetycznej, 297
 mechanizm przesyłania dzienników w formie plików, 357
 mechanizm TOAST, 255
 mechanizm udostępniania informacji o ostatnim użyciu, 240
 menedżer transakcji, 262
 metapolecenia, 37
 metoda losowego doboru próby, 142
 MITM, 175
 moduły zewnętrzne, 89
 modyfikacja parametrów systemu, 87
 monitorowanie, 221, 222

- analiza danych historycznych, 223
- blokowanie zapytań, 231
- czas ostatniego użycia tabeli, 237
- dostarczanie informacji o systemie PostgreSQL do narzędzi monitorujących, 223
- gromadzenie danych historycznych, 222
- gromadzenie dziennych statystyk użycia, 237
- informacje o aktualnie wykonywanym zapytaniu, 226
- monitorowanie w czasie rzeczywistym, 224
- narzędzia, 222, 224
- pgAdmin3, 224

polecenia oczekujące na blokady, 230
 przesyłanie dzienników, 367
 przesyłanie dzienników w formie plików, 359
 sprawdzanie, czy tabela jest używana, 235
 statystyki użycia tabeli, 236
 szybkość wykonywania zapytań, 242
 monitorowanie replikacji, 360, 367
 Mumin, 367
 mrożenie, 258, 279
 mtime, 346
 multi-tenancy, 106
 Multi-Version Concurrency Control, 21, 66, 266
 Mumin, 222, 224, 243, 367
 MVCC, 21, 66, 245, 266
 MySQL, 21

N

n_dead_tup, 268
 nadawanie nazw, 118
 nadawanie uprawnień, 152
 ograniczone uprawnienia superużytkownika, 163
 uprawnienia dostępu do tabeli, 155, 156
 uprawnienia dostępu do wszystkich obiektów
 w schemacie, 156
 Nagios, 223, 224, 270
 naprawa przerośniętych indeksów, 266
 narzędzia monitorujące, 224
 Navicat, 34
 nawiązywanie połączenia z systemem PostgreSQL, 25, 43
 połączenia SSL, 172
 nazwy obiektów bazy danych, 118
 indeksy, 119
 tabele, 118
 wyzwalacze, 119
 nazwy otoczone cudzysłowami, 120
 nazwy systemów, 355
 nextval(), 133
 nieużywane indeksy, 275
 nodes, 351
 NOLOGIN, 159
 NOT NULL, 130, 196
 notacja wielbłądzia, 118
 NTP, 355
 numer wersji serwera, 48

O

obiekty TOAST, 202, 206
 Object Relational Mappers, 292
 obserwacja zapytań, 228
 ps, 229
 obsługa wielu podmiotów, 106
 odbijanie serwera, 113
 odbiorca, 352
 odnajdywanie nieużywanych indeksów, 275

odnajdywanie unikatowego klucza, 135, 137
 odroczone ograniczenia unikatowości, 132
 odtwarzanie dziennika transakcji, 312
 odzyskiwanie bazy danych, 340
 fizyczna kopia zapasowa, 341
 logiczna kopia zapasowa, 340
 wydajność, 341
 odzyskiwanie danych po awarii, 310
 odzyskiwanie do punktu w czasie, 332
 cel odzyskiwania, 333
 odzyskiwanie przy użyciu identyfikatora transakcji, 333
 odzyskiwanie przestrzeni tabel, 338
 fizyczna kopia zapasowa, 339
 logiczna kopia zapasowa, 338
 odzyskiwanie tabeli, 335
 fizyczna kopia zapasowa, 337
 logiczna kopia zapasowa, 335, 336
 odzyskiwanie wszystkich baz danych, 328
 fizyczna kopia zapasowa, 329
 logiczna kopia zapasowa, 328
 OFA, 52
 OFFSET, 292
 ograniczanie liczby jednoczesnych połączeń jednego
 użytkownika, 159
 ograniczanie liczby sesji dla każdego użytkownika do
 jednej, 103
 ograniczanie liczby zwracanych wierszy, 291
 ograniczenia klucza głównego, 130
 ograniczenia NOT NULL, 130, 196
 ograniczenia UNIQUE, 130, 132
 okres doganiania, 353
 OLTP, 228
 ON_ERROR_STOP, 186, 187
 OneClick, 90
 online upgrade, 393
 OpenSSL, 174, 175
 operacje na wielu tabelach, 187
 operacje wejścia-wyjścia, 291
 opóźnienie replikacji, 352, 356
 opóźnienie stosowania, 352
 oprogramowanie klastrowe, 352
 Optimal Flexible Architecture, 52
 optymalizacja na poziomie przestrzeni tabel, 205
 Oracle, 21
 ORDER BY, 292
 ORM, 292
 OS X, 51
 ostateczna spójność, 372, 379
 ośrodek certyfikacji, 174

P

pageinspect, 247
 pamięć, 289
 pan dostępności, 78
 PANIC, 310

- parametry, 82
 - poziomy ustawień, 86
 - ustawianie dla konkretnych grup użytkowników, 85
 - wartości inne niż ustawienia domyślne, 82
- partycjonowanie tabeli, 301
- password, 39
- PATH, 109
- pełne wyszukiwanie tekstowe, 291
- perspektywy, 214, 295
- perspektywy zmaterializowane, 298
- PFA, 52
- pg_archivecleanup, 327, 359
- pg_attrib, 168
- pg_attribute, 265
- pg_batch, 192
- pg_cache_save(), 100
- pg_cache_warm(), 100
- pg_cacheutils, 100
- pg_cancel_backend(), 233
- pg_catalog, 62, 65, 124
- pg_class, 68, 69, 168, 265
- pg_clog, 52
- pg_constraint, 73
- pg_controldata, 56
- pg_ctl, 85, 95, 113
- pg_ctlcluster, 95
- pg_current_xlog_location(), 367
- pg_database, 58, 59, 63
- pg_database_size(), 63
- pg_dump, 142, 246, 313, 316, 317, 318
- pg_dumpall, 313, 315, 318, 329
- pg_hba, 162
- pg_hba.conf, 28, 84
- pg_hba_lockdown.conf, 102
- pg_ident.conf, 84
- pg_last_xlog_apply_location(), 367
- pg_last_xlog_receive_location(), 367
- pg_lesslog, 328, 344
- pg_locks, 264
- pg_ls_dir(), 239
- pg_multixact, 53
- pg_postmaster_start_time(), 50
- pg_prepared_xacts, 264, 265
- pg_relation_size(), 64, 67, 68
- pg_relation_size_nolock(), 70
- pg_reload_conf(), 98
- pg_resetxlog, 312
- pg_restore, 149, 315, 328, 335, 337, 338, 339, 340, 343
- pg_rman, 346
- pg_service.conf, 42, 172
- pg_settings, 81, 83, 98
- pg_sleep(), 228
- pg_Standby, 360
- pg_start_backup(), 358
- pg_stat_activities, 227
- pg_stat_activity, 99, 104, 225, 226, 228, 230, 283, 366
- pg_stat_activity.waiting, 230
- pg_stat_file(), 239
- pg_stat_get_activity, 227
- pg_stat_reset(), 236
- pg_stat_statements, 228, 249
- pg_stat_user_indexes, 223, 286
- pg_stat_user_tables, 223, 236, 284, 286
- pg_statio_user_indexes, 286
- pg_statio_user_tables, 284, 286
- pg_stop_backup(), 323, 324, 358, 362, 371
- pg_subtrans, 53
- pg_tblspc, 53
- pg_terminate_backend(), 104, 105, 159, 232, 233
- pg_total_relation_size(), 64
- pg_twophase, 53
- pg_upgrade, 390
- PG_VERSION, 49
- pg_xlog, 53, 205, 311, 327
- pgAdmin3, 30, 32, 198
 - harmonogram zadań, 33
 - kreator nadawania uprawnień, 31
 - monitorowanie, 224
 - podpowiedzi guru, 11
 - raporty o obiektach, 31
 - wskazówki dnia, 31
- pgAgent, 33
- PGARCHIVE, 357
- pgbouncer, 114, 173
- pgbouncer.ini, 114
- PgCrypto, 175, 179
- PGDATA, 322
- PGDATABASE, 25
- PGDATADIR, 52, 69
- pgFouine, 248
- pgFoundry, 89
- PGHOST, 25
- PGHOSTADDR, 25
- pgloader, 146, 148
- PGOPTIONS, 342
- PGPASSFILE, 41
- PGPASSWORD, 25
- pgpool-II, 385
 - black_function_list, 388
 - delay_threshold, 387
 - health_check_period, 387
 - konfiguracja, 387
 - log_Standby_delay, 387
 - opóźnienie replikacji, 387
 - replikacja poprzez strumieniowe przesyłanie
 - dzienników, 386
 - stosowanie, 386
 - white_function_list, 388
 - wiersz poleceń, 387
- PGPORT, 25
- PgQ, 375
- PGRELEASE, 52

- PGROOT, 52
- PGSERVERNAME, 52
- pgsnmpd, 223
- pgsql_temp, 242
- PGSSLMODE, 174
- pgstatindex(), 270
- pgstattuple(), 270
- PGSYSCONFDIR, 42
- PGUSER, 25
- phpPgAdmin, 30, 33
- piaskownica, 240
- pionowy podział bazy danych, 354
- PITR, 332
- pivot query, 294
- pkey, 119
- PL/Proxy, 170, 213, 214, 354
- PL/PythonU, 230, 238
- plan dostępu i zabezpieczeń, 78
- plan konserwacji, 78
- plan lokalizacji, 78
- plan wysokiej dostępności, 78
- planowanie bazy danych, 77
- planowanie konserwacji, 278
- planowanie tworzenia kopii zapasowych, 312
- plik dziennika serwera, 54
- plik hasel, 40, 41
- plik usługi połączeń, 42
- plik wyzwalacza, 369
- pliki dziennika, 282
- pliki serwera bazy danych, 51
- pliki tymczasowe, 240
 - sprawdzanie użycia, 242
- pobieranie certyfikatu SSL, 174
- pobieranie klucza SSL, 174
- podręcznik użytkownika systemu PostgreSQL, 76
- podstawowa kopia zapasowa, 359
- podsumowanie błędów zarejestrowanych w pliku dziennika, 248
- podwersje, 389
- point-in-time recovery, 332
- pojedynczy punkt awarii, 355
- polecenia DDL, 184
- polecenia oczekujące na blokady, 230
- połączenie z bazą danych, 24, 26, 43
 - połączenia SSL, 172
- ponowne ładowanie plików konfiguracyjnych, 85, 97
- pool_mode, 115
- poradniki raportowania o błędach, 247
- porównywanie tabel, 123
- port, 25, 26, 27, 94
- PostGIS, 89, 317
- postgresql, 54
- PostgreSQL, 13, 19
- PostgreSQL 9, 20
- PostgreSQL Flexible Architecture, 52
- postgresql.conf, 81, 84, 85
 - dyrektywy include, 256
 - include, 85
- postgresql-contrib, 176
- poufne dane, 179
- powtarzające się indeksy, 133
- powtarzające się wiersze, 126
- praktyki replikacji, 354
- prefetch, 101
- PREPARE, 285
- PRIMARY KEY, 130
- primary_conninfo, 364, 365
- problemy związane z wydajnością, 307
- proces aktualizacji, 389
- proces archiwizatora, 92
- proces automatycznego czyszczenia, 92
- proces odbiorcy dziennika WAL, 92
- proces zapisujący w dzienniku WAL, 92
- procesor, 291
- process_emp_audit(), 169
- projekt bazy danych, 77
- projektowanie pod kątem obsługi wielu podmiotów, 106
- protokół SNMP, 223
- próbkowanie danych, 141
- próbkowanie losowe, 143
- przebudowa indeksów, 271
- przebudowa schematu, 302
- przedział czasu, 134
- przedziały przedrostków, 135
- przekazywanie, 352
- przekroczenie licznika transakcji, 260, 261
- przełączanie, 352, 368
- przełączanie awaryjne, 352, 368
- przełączanie powrotne, 369
- przenoszenie części zapytania do perspektywy, 295
- przenoszenie obiektów pomiędzy przestrzeniami tabel, 205
- przenoszenie obiektów pomiędzy schematami, 200
- przenoszenie obliczeń na poziom funkcji bazy danych, 306
- przenoszenie własności na innych użytkowników, 161
- przerośnięte indeksy, 245, 266
- przerośnięte tabele, 245, 266
- przestrzenie nazw, 321
- przestrzenie tabel, 201
 - odzyskiwanie, 338
- przestrzeń dyskowa zajęta przez bazę danych, 63
- przestrzeń dyskowa zajęta przez tabele, 64
- przesyłanie dzienników w formie plików, 356
 - konfiguracja, 357
 - monitorowanie, 359
- przeszukiwanie sekwencyjne, 66
- przetwarzanie dzienników, 249
- przetwarzanie współbieżne, 281
- przydzielanie przedziałów adresów IP, 133
- przygotowane transakcje, 263, 264
- przygotowanie raportów, 246
- przyrostowe kopie zapasowe, 345
- przyspieszanie zapytań, 299

- przyznawanie użytkownikom prywatnych baz danych, 110
 - psql, 35, 142, 183
 - błędy, 185
 - help, 37
 - komentarze, 37
 - \, 58
 - metapolecenia, 37
 - ON_ERROR_STOP, 187
 - parametry połączenia, 35
 - single-transaction, 183
 - skrypty, 35
 - tryb interaktywny, 36
 - zapytania SQL, 35, 37
 - PUBLIC, 155
 - pula połączeń, 113, 115
 - punkty kontrolne, 311
- Q**
- quote_ident(), 121
- R**
- RADIUS, 40
 - random(), 141, 142
 - random_page_cost, 305
 - raporty, 246
 - raportowanie o błędach, 247
 - raportowanie o problemach związanych z wydajnością, 307
 - Reading the Fine Manual, 76
 - REASSIGN OWNED, 161
 - recovery target, 333
 - recovery.conf, 324, 330, 331, 333
 - recovery.done, 331
 - recovery_end_command, 369
 - recovery_target_time, 333
 - Red Hat RHEL, 51
 - Reference Data Management, 351
 - referential integrity, 72
 - reguły, 217, 218
 - REINDEX, 271
 - REINDEX CONCURRENTLY, 275
 - reindexdb, 271
 - rejestrwanie użycia plików tymczasowych, 242
 - rekordy czyszczenia, 373
 - relay, 352
 - relname2relid(), 73
 - relpages, 68
 - reltuples, 68
 - RENAME, 119
 - repeat(), 138
 - replication delay, 352
 - replikacja, 349, 351
 - czas, 355
 - Londiste, 375
 - monitorowanie, 356, 360, 367
 - nazwy systemów, 355
 - okres doganiania, 353
 - opóźnienie replikacji, 352, 356
 - opóźnienie stosowania, 352
 - pojedynczy punkt awarii, 355
 - praktyki replikacji, 354
 - przełączanie, 368
 - przełączanie awaryjne, 368
 - przełączanie powrotne, 369
 - przesyłanie dzienników w formie plików, 356
 - replikacja asynchroniczna, 353
 - replikacja na bazie wyzwalaczy, 379
 - replikacja propagowana, 381
 - replikacja selektywna, 353, 375
 - replikacja synchroniczna, 352
 - replikacja transakcyjna, 353
 - rozdwojenie jaźni, 368
 - ruch danych, 351
 - Slony, 380
 - strefa czasowa, 355
 - strumieniowe przesyłanie dzienników, 361
 - wsady, 352
 - zabezpieczenia, 361
 - zarządzanie replikacją poprzez przesyłanie dzienników, 366
 - repmgr, 375
 - RESET, 79, 80
 - resolving in-doubt transactions, 263
 - restartowanie serwera, 99
 - restore_command, 330, 331, 358, 359, 365
 - resync, 355
 - REVOKE, 111, 156
 - REVOKE ALL ON, 153
 - REVOKE FROM PUBLIC EXCEPT, 111
 - REVOKE SCHEMA, 153
 - role, 152, 156, 157
 - ROLLBACK, 167
 - ROLLBACK PREPARED, 263
 - roll-up, 381
 - row_number(), 192
 - rozchodzenie, 381
 - rozdwojenie jaźni, 368
 - rozłączanie użytkowników, 104
 - rozmiar bazy danych, 63, 87
 - rozmiar tabel, 64
 - tabele tymczasowe, 240
 - rozstrzyganie transakcji, 235, 263
 - rozwiązywanie problemów z nawiązywaniem połączenia, 43
 - równoległe przetwarzanie potokowe, 344
 - równoległe wykonywanie zadań, 190, 192
 - równoważenie obciążenia, 385
 - różnicowe kopie zapasowe, 345
 - RRDtool, 222, 224

rsync, 327, 345, 359
 RTFM, 76
 Ruby, 249
 ruch danych, 351
 rzutowanie typów danych, 140

S

sandbox, 240
 schemat, 107, 198, 321
 przebudowa, 302
 schemat informacyjny, 62
 schodzenie, 381
 security definer, 165
 SECURITY DEFINER, 163
 sekwencja układania, 60
 sekwencje, 119
 selective replication, 353
 seq, 119
 seq_page_cost, 305
 seq_scan, 284
 seq_tup_read, 284
 Sequential Scan, 66
 SERIAL, 201
 serwer bazy danych, 13, 26, 94, 351
 awaryjne zatrzymywanie, 96
 katalog danych, 112
 ponowne ładowanie plików konfiguracyjnych, 97
 projektowanie pod kątem obsługi wielu
 podmiotów, 106
 restartowanie, 99
 stosowanie wielu schematów, 107
 uruchamianie, 94
 uruchamianie wielu serwerów w jednym systemie, 112
 zapobieganie nowym połączeniom, 101
 zatrzymywanie, 95
 serwer główny, 351
 serwer LDAP, 171
 konfiguracja klienta, 172
 serwer nadawcy, 351
 serwer podrzędny, 352
 serwer podstawowy, 351
 serwer pomocniczy, 352
 serwer źródłowy, 351
 sesje, 24
 sesje autonomiczne, 261
 SET, 79, 80
 SET LOCAL, 80
 SET SCHEMA, 200
 sharding, 354
 shared nothing, 354
 shared_buffers, 84, 87, 100, 289, 343
 shared_preload_libraries, 91
 SHOW, 81, 83, 116
 SHOW CLIENTS, 116
 SHOW CONFIG, 116
 SHOW DATABASES, 116
 SHOW FDS, 116
 SHOW LISTS, 116
 SHOW POOLS, 116
 SHOW SERVERS, 116
 SHOW SOCKETS, 116
 SHOW STATS, 116
 SHOW USERS, 116
 SHOW VERSION, 116
 shutdown abort, 97
 SIGHUP, 98
 SIGKILL, 233
 SIGQUIT, 232
 Simple Network Management Protocol, 223
 single point of failure, 355
 skalowanie poziome, 354
 skrypty, 35, 182, 183
 skrypty narzędzia psql, 185
 slonik, 384
 Slony, 317, 380
 konfiguracja, 382
 konservacja replikacji, 384
 proces pełnej replikacji, 382
 replikacja, 380
 replikacja propagowana, 381
 rozchodzenie, 381
 schodzenie, 381
 Slony1-ctl, 382
 SNMP, 223
 sourcefile, 83
 sourceline, 83
 split, 317
 split-brain, 368
 SPOF, 355
 spowolnienia w bazie danych, 282
 sprawdzanie ról użytkownika, 157
 sprawdzanie wersji serwera, 48, 49
 sprawdzanie, czy komputer jest połączony z bazą
 danych, 226
 sprawdzanie, czy plik tymczasowy jest używany, 242
 sprawdzanie, czy tabela jest używana, 235
 sprawdzanie, czy użytkownik jest połączony, 225
 SQL, 13, 20, 21
 SQL 2008, 20
 SSL, 172
 konfiguracja klienta, 174
 sprawdzanie autentyczności serwera, 175
 SSPI, 40
 stan gotowości, 44
 standalone backend, 261
 standard_conforming_strings, 149
 standby, 44
 Standby_mode, 374
 Standby_mode ani trigger_file, 360
 STANDBYNODE, 357
 START_DATE, 134

stat_user_indexes_delta, 286
 stat_user_indexes_delta_log, 286
 stat_user_tables_delta, 286
 stat_user_tables_delta_log, 286
 statement_timeout, 234
 statystyki, 285

- użycie tabeli, 236

 stosowanie wielu schematów, 107
 strategia zawsze przyrostowej kopii zapasowej, 346
 strony, 69
 strumieniowe przesyłanie dzienników, 361
 stunnel, 115
 subskrybent, 352
 substr(), 139
 superużytkownicy, 111
 supplementary storage tables, 255
 switchback, 369
 switchover, 352, 368
 symbole ucieczki, 149
 synchronous_commit, 233, 366
 sysctl, 87
 syslog, 54, 248
 syslogd, 248
 system PostgreSQL, 20
 system raportujący, 351
 system zarządzania bazami danych, 21
 szacowanie liczby wierszy w tabeli, 67
 szybkość wykonywania zapytań, 242
 szyfrowanie danych, 175
 szyfrowanie haseł, 162

Ś

ścieżka wyszukiwania bazy danych, 154

T

tabele, 60

- blokady, 129
- czas ostatniego użycia, 237
- dane testowe, 137
- dodawanie kolumn, 192
- kolumny, 122
- liczba wierszy, 65
- lista największych tabel, 65
- nazwy, 118
- odzyskiwanie, 335
- partycjonowanie, 301
- porównywanie tabel, 123
- powtarzające się wiersze, 126
- rozmiar, 64
- statystyki użycia, 236
- szacowanie liczby wierszy, 67
- tabele asocjacyjne, 118
- tabele pamięci masowej, 255
- tabele podlegające wielu operacjom aktualizacji, 301

tabele tymczasowe, 240, 242, 265, 297
 tabele wyszukiwań, 118
 usuwanie kolumn, 192
 zapobieganie występowaniu powtarzających się wierszy, 129
 zmiana typu danych kolumny, 195
 tabele TOAST, 204, 255

- czyszczenie, 255

 table bloat, 245
 table_file_info(), 238, 239
 tcp_keepalives_idle, 364
 technika blokowania optymistycznego, 305
 technika zatwierdzania dwufazowego, 235
 temp_tablespace, 203, 240, 241
 The Outsized Attribute Storage Technique, 65
 ticker, 378
 timeofday(), 133
 TM, 262
 to_date(), 196
 TOAST, 65, 202, 204, 255
 toast.autovacuum_analyze_scale_factor, 254
 toast.autovacuum_analyze_threshold, 254
 toast.autovacuum_enabled, 254
 toast.autovacuum_freeze_max_age, 254
 toast.autovacuum_freeze_min_age, 254
 toast.autovacuum_freeze_table_age, 254
 toast.autovacuum_vacuum_cost_delay, 254
 toast.autovacuum_vacuum_cost_limit, 254
 toast.autovacuum_vacuum_scale_factor, 254
 toast.autovacuum_vacuum_threshold, 254
 toast_blks_read, 289
 trace_recovery_messages, 371
 track_activities, 227
 track_counts, 253
 Transaction Manager, 262
 transaction wraparound, 260
 transakcje, 129, 183

- rozstrzyganie transakcji, 235, 263
- usuwanie przygotowanych transakcji, 262

 trial_drop_index(), 277
 trial_undrop_index(), 277
 trwale kodowanie hasła, 40
 trwale połączenie z bazą danych, 211
 tryb gorącej gotowości, 332, 370

- rekordy czyszczenia, 373
- stosowanie, 370, 375

 tryb oszczędzania energii, 91
 tryb pojedynczego użytkownika, 261
 tryb równoważenia obciążeń, 385
 two-phase commit, 235
 tworzenie

- baza danych, 58
- fizyczna kopia zapasowa, 322
- funkcje, 213
- indeksy UNIQUE INDEX, 130
- catalog danych, 112

logiczna kopia zapasowa, 316
 role, 156
 różnicowa kopia zapasowa, 345
 schematy, 108
 tabele z danymi testowymi, 137
 użytkownicy, 157
 tymczasowe uniemożliwianie użytkownikowi
 nawiązywania połączenia, 158

U

Ubuntu, 51
 umieszczenie katalogu pg_xlog na odrębnym
 urządzeniu, 205
 unikanie automatycznego mrożenia, 258
 unikanie przekroczenia licznika transakcji, 260
 unikanie trwałego kodowania hasła, 40
 unikatowość bez indeksów, 133
 UNIQUE, 130, 132
 UNIQUE INDEX, 130
 UPDATE, 255, 276, 306
 update_process_title, 229
 upraszczanie złożonych wyrażeń języka SQL, 293
 uprawnienia, 152
 uprawnienia dostępu do odpowiedniego schematu, 156
 uprawnienia na poziomie schematu, 200
 uruchamianie serwera bazy danych, 94
 tryb oszczędzania energii, 91
 uruchamianie wielu serwerów w jednym systemie, 112
 USING, 196, 197
 ustawianie parametrów dla konkretnych grup
 użytkowników, 85
 ustawienia konfiguracyjne, 81
 usuwanie
 kolumny tabeli, 192
 niepotrzebne indeksy, 276
 powtarzające się wiersze, 126
 przestrzenie tabel, 201
 przygotowane transakcje, 262, 263
 schemat, 198
 użytkownik bez usuwania jego danych, 160
 uwierzytelnianie, 28, 39
 użycie plików tymczasowych, 242
 użytkownicy, 25, 152
 nadawanie uprawnień dostępu do tabeli, 155
 nadawanie uprawnień „usuniętego” użytkownika
 nowemu użytkownikowi, 160
 przyznawanie prywatnych baz danych, 110
 tworzenie, 157
 usuwanie użytkownika bez usuwania jego danych, 160
 wycofywanie dostępu do tabeli, 153

V

VACUUM, 128, 251, 254, 256
 VACUUM ANALYZE, 252
 vacuum_defer_cleanup_age, 371, 373

vacuum_freeze_min_age, 258
 vacuum_freeze_table_age, 259
 vacuumdb, 260
 version(), 27, 48
 Visibility Map, 64
 VPN, 173

W

WAL, 92, 310, 358
 wal_buffers, 88
 wal_keep_segments, 362, 365
 wal_level, 360, 371
 wal_sender_delay, 363
 wal_writer_delay, 91
 WALReceiver, 363
 WALSender, 363
 wersja serwera, 48
 weryfikacja zmian w danych, 168
 weryfikacja zmian wprowadzonych za pomocą wyrażeń
 języka DDL, 166
 węzły, 351
 WHERE, 73, 131, 141, 142, 192, 217, 293
 wiersze, 69
 Windows, 51
 wirtualne sieci prywatne, 173
 WITH, 296
 wolne wyrażenia języka SQL, 282
 blokady, 290
 dyskowe operacje wejścia-wyjścia, 291
 identyfikacja przyczyn, 287
 ilość przetwarzanych danych, 287
 pamięć, 289
 procesor, 291
 zapytania uruchamiane jako przygotowane
 wyrażenia, 285
 zapytania zwracające dużo danych, 289
 work_mem, 88, 128
 Write Ahead Log, 310
 wsady, 352, 379
 współczynnik wypełniania, 269
 wycofywanie dostępu użytkownika do tabeli, 153
 wydajność, 281
 wydajność odzyskiwania, 341
 wydajność tworzenia kopii zapasowych, 341
 fizyczna kopia zapasowa, 342
 logiczna kopia zapasowa, 342
 wykluczanie ograniczeń, 301
 wykonywanie skryptów, 35
 wykonywanie zapytań, 35
 wykrywanie ataków, 279
 wymiana kluczy głównych, 274
 wymuszanie rozłączenia użytkowników z flagą
 NOLOGIN, 159
 wymuszanie stosowania tych samych definicji dla tak
 samo nazwanych kolumn, 122

wymuszanie użycia indeksu, 303
 wyrażenia języka DDL, 166
 wyrażenia SQL, 37
 wysoka dostępność, 351
 wyszukiwanie wolnych wyrażeń języka SQL, 282
 zapytania uruchamiane jako przygotowane
 wyrażenia, 285
 wyzwalacze, 119
 gromadzenie zmian, 169
 INSTEAD OF, 219
 nazwy, 119
 wyzwalacze wykonywane zamiast oryginalnych
 zdarzeń, 216
 wzorce użycia, 279

X

xid, 333

Z

zabezpieczenia replikacji, 361
 zabijanie beczynnych zapytań w ramach transakcji, 234
 zabijanie sesji, 232, 234
 zadania analityczne, 292
 zależności łączące obiekty, 71
 zamrażanie bazy danych, 374

zapobieganie nowym połączeniom, 101
 zapobieganie występowaniu powtarzających się
 wierszy, 129
 zapytania, 35
 zapytania krzyżowe, 294
 zapytania wyszukujące, 292
 zarządzanie danymi referencyjnymi, 351
 zarządzanie klastrem, 352
 zarządzanie replikacją poprzez przesyłanie dzienników, 366
 zarządzanie trybem gorącej gotowości, 370
 zatrzymywanie serwera, 95
 zatwierdzanie dwufazowe, 235
 zdalne źródła danych, 212
 zdalny dostęp do serwera, 27
 zjawisko przerostu tabeli, 245
 złożone wyrażenia języka SQL, 293
 zmiana hasła, 39
 zmiana parametrów, 85
 zmiana parametrów na poziomie programów, 79
 zmiana typu danych kolumny, 195
 zmiana własności w starszych bazach danych, 161
 zmienne środowiskowe, 25
 zrzut bazy danych, 246
 zwijanie, 381

Zobacz, co możesz osiągnąć razem

z bazą PostgreSQL!

PostgreSQL to jedna z najbardziej zaawansowanych baz danych o otwartym kodzie źródłowym. Przez wiele lat była niedoścignionym wzorem dla innego darmowego rozwiązania – MySQL. Dziś znajduje zastosowanie wszędzie tam, gdzie wymagana jest najwyższa niezawodność i wydajność, a brak konieczności zapłaty gra kluczową rolę. Stosunek jakości do ceny w przypadku PostgreSQL zmierza do nieskończoności!

Trzymasz w rękach książkę zawierającą liczne przepisy na najlepsze wykorzystanie PostgreSQL. System ten sprawdza się zawsze, gdy chcesz szybko i bezproblemowo osiągnąć zamierzone cele. W trakcie lektury dowiesz się, jak nawiązać połączenie z serwerem, skorzystać z graficznych lub tekstowych narzędzi administracyjnych oraz bezpiecznie zmienić hasło administratora. Ponadto nauczysz się kontrolować przestrzeń dyskową wykorzystywaną przez poszczególne bazy danych, tworzyć tabele, ładować dane oraz zarządzać użytkownikami i ich uprawnieniami. Autorzy dużo miejsca poświęcają kwestii bezpieczeństwa. W końcu dane to najcenniejsza rzecz, jaką przechowuje się w bazach! Każdy z rozdziałów przynosi ogrom wiedzy o różnym poziomie skomplikowania. Zaawansowanych użytkowników zainteresuje rozdział poświęcony replikacji, a tych początkujących rozdział traktujący o uruchamianie i zatrzymywaniu serwera baz danych. Ta książka przyda się po prostu wszystkim użytkownikom PostgreSQL!

- Zalety PostgreSQL w kontekście innych rozwiązań bazodanowych
- Udostępnianie serwera w sieci
- Zastosowanie narzędzia psql do wykonywania zapytań
- Sprawdzanie wersji serwera
- Lista baz danych na serwerze
- Planowanie nowej bazy danych
- Parametry, ich znaczenie i ustawianie
- Uruchamianie i zatrzymywanie serwera
- Ponowne ładowanie plików konfiguracyjnych
- Przyznawanie użytkownikom własnych baz danych
- Wiele serwerów baz danych w ramach jednego systemu operacyjnego
- Generowanie danych testowych
- Tworzenie kont użytkowników i zarządzanie nimi oraz ich uprawnieniami
- Równoległe wykonywanie zadań – polecenie pg_batch
- Monitorowanie i diagnostyka serwera PostgreSQL
- Przygotowywanie kopii bezpieczeństwa

W katalogu 8182



Księgarnia internetowa:
<http://helion.pl>



Zamówienia telefonicznie:
0 801 339900
0 601 339900

helion.pl
księgarnia
internetowa

Sprzedajemy również promocje:
• <http://helion.pl/promocje>
Książki w wygodnej cenie:
• <http://helion.pl/interaktywne>
Zamów informacje o nowościach:
• <http://helion.pl/novosci>



Helion

Helion SA
ul. Bobuski 1c, 44-100 Gliwice
tel.: 32 230 94 43
e-mail: helion@helion.pl
<http://helion.pl>

Cena: 79,00 zł

ISBN 978-83-246-3061-5



Informatyka w najlepszym wydaniu