pgBackRest: podstawowe komendy

Poziom: podstawy **Dotyczy: pgBackRest 2.x, PostgreSQL 9.6–17** **Czas: ~25 min**

`pg_dump` zrzuca dane przez SQL, w jednym punkcie w czasie — pgBackRest kopiuje pliki na poziomie systemu i dokłada do tego archiwum WAL, dzięki czemu potrafi odtworzyć bazę nie tylko do chwili backupu, ale do dowolnej sekundy między jednym backupem a drugim.

Skróty użyte na tej stronie (3)

Skrót	Co znaczy
RPO	Recovery Point Objective: ile danych wolno stracić przy awarii
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
WAL	Write-Ahead Log: dziennik PostgreSQL, zapis zmian przed zapisem danych

Fiszka

ZMIENIA SYSTEM

Utworzenie nowej stanzy: stanza-create

```
pgbackrest --stanza=NazwaStanzy stanza-create --log-level-console=info
```

Zakładamy w repozytorium strukturę katalogów dla nowej stanzy — wykonujemy raz, przy pierwszej konfiguracji backupu dla danego klastra.

Przykład:

```
pgbackrest --stanza=produkcja stanza-create --log-level-console=info
```

Przykład:

```
P00   INFO: stanza-create command begin
P00   INFO: stanza-create for stanza 'produkcja' on repo1
P00   INFO: stanza-create command end: completed successfully
```

pgbackrest = Linux: narzędzie do kopii fizycznych PostgreSQL z archiwizacją WAL, tu do odtworzenia bazy na maszynie testowej

TYLKO ODCZYT

Sprawdzenie poprawności konfiguracji stanzy: check

```
pgbackrest --stanza=NazwaStanzy check
```

Sprawdzamy, czy pgBackRest potrafi połączyć się z instancją, zapisać do repozytorium i czy archiwizacja WAL działa poprawnie — pierwsze polecenie do uruchomienia po każdej zmianie konfiguracji, zanim spróbujemy właściwego backupu.

Przykład:

```
P00   INFO: check command begin
P00   INFO: WAL segment 000000010000000000000003 successfully archived
P00   INFO: check command end: completed successfully
```

WAL = Write-Ahead Log: dziennik PostgreSQL, zapis zmian przed zapisem danych · **pgbackrest** = Linux: narzędzie do kopii fizycznych PostgreSQL z archiwizacją WAL, tu do odtworzenia bazy na maszynie testowej

TYLKO ODCZYT

Wykonanie backupu pełnego: backup --type=full

```
pgbackrest --stanza=NazwaStanzy backup --type=full
```

Kopiujemy wszystkie pliki bazy od zera — najdłuższy i najbardziej miejscochłonny typ backupu, wykonywany zwykle raz na tydzień albo raz na miesiąc, w zależności od rozmiaru bazy i ustalonego RPO, tematu strony RPO i RTO: dwie liczby, od których zaczyna się każdy plan backupu.

Przykład:

```
P00   INFO: backup command begin
P00   INFO: full backup size = 4.7GB
P00   INFO: new backup label = 20260924-030000F
P00   INFO: backup command end: completed successfully
```

RPO = Recovery Point Objective: ile danych wolno stracić przy awarii · **RTO** = Recovery Time Objective: jak długo wolno nie działać przy awarii · **pgbackrest** = Linux: narzędzie do kopii fizycznych PostgreSQL z archiwizacją WAL, tu do odtworzenia bazy na maszynie testowej

TYLKO ODCZYT

Wykonanie backupu różnicowego: backup --type=diff

```
pgbackrest --stanza=NazwaStanzy backup --type=diff
```

> **Pojęcie: backup różnicowy.** Kopiuje tylko to, co zmieniło się od ostatniego backupu pełnego — każdy kolejny backup różnicowy rośnie, aż do następnego pełnego, ale odtworzenie wymaga tylko dwóch backupów: ostatniego pełnego i najnowszego różnicowego.

Przykład:

```
P00   INFO: backup command begin
P00   INFO: diff backup size = 340MB
P00   INFO: new backup label = 20260925-030000F_20260925-030512D
P00   INFO: backup command end: completed successfully
```

pgbackrest = Linux: narzędzie do kopii fizycznych PostgreSQL z archiwizacją WAL, tu do odtworzenia bazy na maszynie testowej

TYLKO ODCZYT

Wykonanie backupu przyrostowego: backup --type=incr

```
pgbackrest --stanza=NazwaStanzy backup --type=incr
```

> **Pojęcie: backup przyrostowy.** Kopiuje tylko to, co zmieniło się od poprzedniego backupu, niezależnie od tego, czy tamten był pełny, różnicowy czy również

przyrostowy — najmniejszy i najszybszy z trzech typów, ale odtworzenie wymaga całego łańcucha backupów od ostatniego pełnego aż do wybranego przyrostowego.

Przykład:

```
P00   INFO: backup command begin
P00   INFO: incr backup size = 62MB
P00   INFO: new backup label = 20260925-030000F_20260926-030214I
P00   INFO: backup command end: completed successfully
```

pgbackrest = Linux: narzędzie do kopii fizycznych PostgreSQL z archiwizacją WAL, tu do odtworzenia bazy na maszynie testowej

TYLKO ODCZYT

Sprawdzenie dostępnych backupów: info

```
pgbackrest info
```

Wyświetlamy listę stanz skonfigurowanych w repozytorium wraz z historią backupów każdej z nich — pierwsze polecenie przy planowaniu odtworzenia, żeby wiedzieć, do jakich punktów w czasie w ogóle można wrócić.

Przykład:

```
stanza: produkcja
    status: ok
    cipher: none

    db (current)
        wal archive min/max (14): 000000010000000000000001/000000010000000000000009

        full backup: 20260924-030000F
            timestamp start/stop: 2026-09-24 03:00:00 / 2026-09-24 03:41:12
            wal start/stop: 000000010000000000000001 / 000000010000000000000001
            database size: 4.7GB, database backup size: 4.7GB

        diff backup: 20260925-030000F_20260925-030512D
            timestamp start/stop: 2026-09-25 03:00:00 / 2026-09-25 03:04:30
            database size: 4.8GB, database backup size: 340MB
```

pgbackrest = Linux: narzędzie do kopii fizycznych PostgreSQL z archiwizacją WAL, tu do odtworzenia bazy na maszynie testowej

ZMIENIA SYSTEM

Odtworzenie z ostatniego backupu: restore

```
pgbackrest --stanza=NazwaStanzy restore
```

Odtwarzamy `PGDATA` z najnowszego dostępnego backupu i dogrywamy WAL z archiwum aż do ostatniego zarchiwizowanego segmentu — instancja musi być zatrzymana, a katalog danych pusty albo usunięty przed uruchomieniem tego polecenia.

Przykład:

```
P00   INFO: restore command begin
P00   INFO: restore backup set 20260925-030000F_20260925-030512D
P00   INFO: restore command end: completed successfully
```

WAL = Write-Ahead Log: dziennik PostgreSQL, zapis zmian przed zapisem danych · **pgbackrest** = Linux: narzędzie do kopii fizycznych PostgreSQL z archiwizacją WAL, tu do odtworzenia bazy na maszynie testowej

ZMIENIA SYSTEM

Odtworzenie do punktu w czasie: restore --type=time --target

```
pgbackrest --stanza=NazwaStanzy --type=time --target="2026-09-24 10:00:00" restore
```

> **Pojęcie: odtwarzanie do punktu w czasie.** Zamiast zatrzymać się na ostatnim dostępnym backupie, pgBackRest odtwarza backup pełny, a następnie odtwarza z archiwum WAL wszystkie zmiany aż do wskazanej sekundy — nie ani chwili dalej. To jest jedyny sposób na cofnięcie bazy do stanu sprzed konkretnego, znanego zdarzenia, na przykład tuż przed błędnym `UPDATE` bez `WHERE`, temat rozwinięty na stronie Odtwarzanie do punktu w czasie (PITR).

Przykład:

```
P00   INFO: restore command begin
P00   INFO: restore backup set 20260924-030000F
```

```
P00   INFO: recovery type is time, target is '2026-09-24 10:00:00'
P00   INFO: restore command end: completed successfully
```

WAL = Write-Ahead Log: dziennik PostgreSQL, zapis zmian przed zapisem danych · **PITR** = Point-In-Time Recovery: odtworzenie bazy do dowolnej chwili, np. 14:31 · **pgbackrest** = Linux: narzędzie do kopii fizycznych PostgreSQL z archiwizacją WAL, tu do odtworzenia bazy na maszynie testowej

TYLKO ODCZYT

Usunięcie przeterminowanych backupów: expire

```
pgbackrest --stanza=NazwaStanzy expire
```

> **Pojęcie: expire (wygaszanie retencji).** Usuwa z repozytorium backupy starsze niż retencja zdefiniowana parametrami `repo1-retention-*` w `pgbackrest.conf`, wraz z segmentami WAL, które nie są już potrzebne żadnemu z pozostałych backupów. `expire` uruchamiany jest zwykle automatycznie po każdym backupie, ale da się go wywołać też ręcznie.

Przykład:

```
P00   INFO: expire command begin
P00   INFO: expire full backup 20260910-030000F
P00   INFO: expire command end: completed successfully
```

WAL = Write-Ahead Log: dziennik PostgreSQL, zapis zmian przed zapisem danych · **pgbackrest** = Linux: narzędzie do kopii fizycznych PostgreSQL z archiwizacją WAL, tu do odtworzenia bazy na maszynie testowej

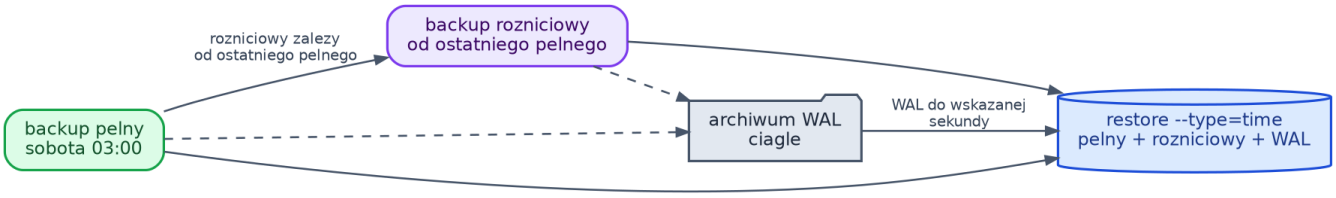
Z życia wzięte

Z ŻYCIA WZIĘTE

Wieczorem w sobotę pgBackRest robi pełny backup bazy Zabbixa na pg01, a przez kolejny tydzień dokłada tylko backupy różnicowe. W czwartek Ola przypadkowo kasuje dane z panelu i pytamy `pgbackrest info`, do jakiego punktu w czasie w ogóle możemy wrócić. Odtwarzamy backup pełny plus najnowszy różnicowy i dogrywamy WAL do minuty tuż przed błędnym poleceniem Oli.

Czym jest pgBackRest

> **Pojęcie: pgBackRest.** Narzędzie do backupu fizycznego i archiwizacji WAL dla PostgreSQL, zarządzające całym cyklem: backupem pełnym, backupami różnicowymi i przyrostowymi, kompresją, retencją i odtwarzaniem, łącznie z odtwarzaniem do punktu w czasie. W przeciwieństwie do `pg_dump`, opisanego na stronie pg_dump, pg_dumpall, pg_restore, pgBackRest nie odpytuje bazy przez SQL — kopiuje bezpośrednio pliki danych z `PGDATA`, korzystając z tego samego trybu backupu na gorąco, którego mechanizm pokazuje `pg_start_backup` / `pg_stop_backup` na stronie Logi WAL: włączenie i archiwizacja.



Backup pełny w sobotę, backup różnicowy zależny od niego i ciągłe archiwum WAL, odtworzenie do punktu w czasie łączy wszystkie trzy (kliknięcie otwiera pełny rozmiar)

	pg_dump	pgBackRest
Poziom działania	logiczny, przez SQL	fizyczny, na poziomie plików
Wynik	dane w formacie przenośnym między wersjami	kopia plików danego klastra
Backup przyrostowy	nie istnieje	tak, na poziomie zmienionych bloków
Odtwarzanie do punktu w czasie	nie	tak, dzięki archiwum WAL
Migracja między wersjami głównymi	tak	nie — odtwarza dokładnie tę samą wersję, z której backup powstał

Ogólny podział typów backupu — pełny, różnicowy, przyrostowy — i to, kiedy który wybrać, rozwija strona Backup pełny, różnicowy, przyrostowy, logi. pgBackRest jest narzędziem, które w PostgreSQL realizuje wszystkie trzy typy jednym spójnym mechanizmem retencji.

Konfiguracja pgbackrest.conf

> **Pojęcie: stanza.** W terminologii pgBackRest to nazwana konfiguracja jednego klastra PostgreSQL objętego backupem — jeden plik `pgbackrest.conf` może trzymać konfigurację kilku stanz, po jednej sekcji na każdy klaster, jeśli na serwerze repozytorium backupujemy więcej niż jedną instancję.

Konfiguracja backupów leży domyślnie w `/etc/pgbackrest/pgbackrest.conf`. Przykład konfiguracji z zapisem na dedykowanym wolumenie `/pgbackrest` i retencją czternastu dni:

```
[global]
repo1-path=/pgbackrest
repo1-retention-full-type=time
repo1-retention-full=14
process-max=2
log-level-console=info
log-level-file=debug

[produkcja]
pg1-path=/postgres/14/data
pg1-port=5432
pg1-user=postgres
```

Sekcja `[global]` obejmuje ustawienia wspólne dla wszystkich stanz na danym serwerze repozytorium, sekcja nazwana tak jak stanza (`[produkcja]` w przykładzie) wskazuje, gdzie leży `PGDATA` konkretnego klastra i jakim kontem systemowym pgBackRest ma się do niego łączyć.

Parametr	Znaczenie
`repo1-path`	katalog repozytorium backupów
`repo1-retention-full-type`	tryb retencji — `time` (dni) albo `count` (liczba backupów pełnych)
`repo1-retention-full`	liczba dni albo backupów pełnych do zachowania, zależnie od trybu wyżej
`process-max`	liczba równoległych procesów przy backupie i odtwarzaniu
`log-level-console` / `log-level-file`	poziom szczegółowości logów na konsoli i w pliku
`pg1-path`	ścieżka `PGDATA` klastra objętego backupem
`pg1-port`	port instancji
`pg1-user`	konto systemowe, którym pgBackRest łączy się z instancją

Częste błędy

Wykonanie stanza-create bez wcześniej działającej archiwizacji WAL. `stanza-create` sprawdza, czy `archive_command` z Logi WAL: włączenie i archiwizacja w ogóle działa — jeśli nie, polecenie kończy się błędem, zanim jeszcze dojdzie do pierwszego backupu.

Odtwarzanie backupu przyrostowego bez zachowania całego łańcucha. Backup przyrostowy jest bezużyteczny bez wszystkich poprzedzających go backupów aż do ostatniego pełnego — usunięcie pojedynczego ogniwa łańcucha (np. ręcznie, mimo działającego `expire`) unieważnia wszystkie kolejne po nim backupy różnicowe i przyrostowe.

Powiązane: Backup pełny, różnicowy, przyrostowy, logi · Logi WAL: włączenie i archiwizacja · Odtwarzanie do punktu w czasie (PITR) · pg_dump, pg_dumpall, pg_restore

Źródła: dokumentacja pgBackRest, `pgbackrest.org/user-guide.html`

Revision #44
Created 2026-09-24 23:08:12 UTC by Tomasz Konieczny
Updated 2026-10-26 14:12:37 UTC by Tomasz Konieczny