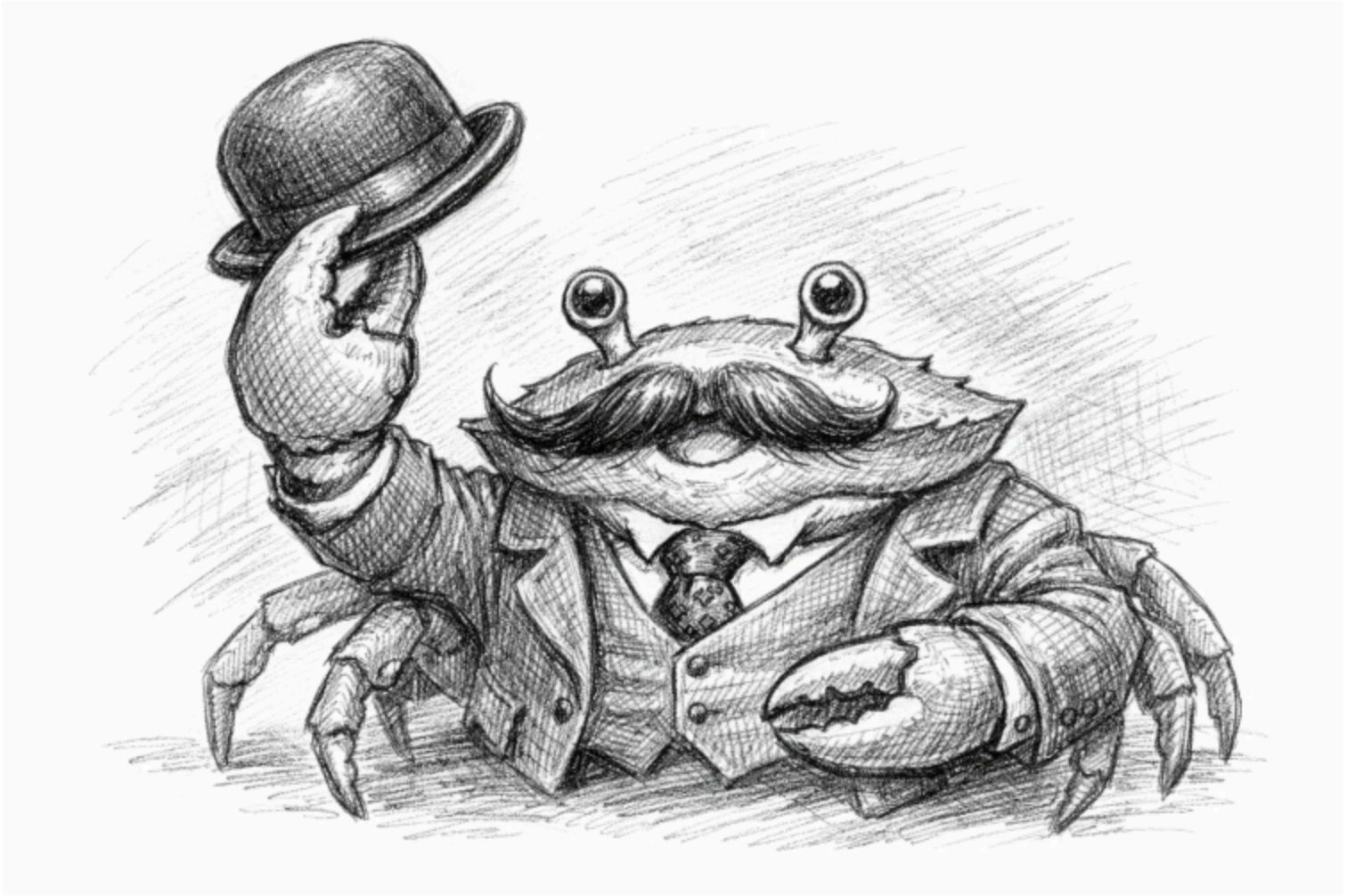
OH,REALLY?

Reciepts masters of qbackup



Qbackup Cookbook

QHB PRESS

the team of authors

Команда авторов

Qbackup Cookbook

Рецепты мастеров резервного копирования
0-е издание

Москва
2026

УДК 004.658:004.65
ББК 32.973.26-018.2

Qbackup Cookbook: Рецепты мастеров резервного копирования, 0-е издание / Команда авторов, Москва, 2026. - 43 с.

В книге собраны практические рецепты работы с утилитой резервного копирования qbackup для СУБД QHB — российского форка PostgreSQL. Читатель найдёт готовые решения для типичных и нетипичных задач: от первоначальной настройки и выбора стратегии резервного копирования до тонкостей инкрементального бэкапа, восстановления на момент времени (PITR) и удалённого восстановления через SSH. Книга охватывает оба режима работы qbackup — потоковый (stream) и прямой (direct), — а также механизм захвата изменений данных (CDC), сжатие и управление политиками удержания. Каждый рецепт снабжён готовыми командами и пояснениями, позволяющими сразу применить решение на практике. Издание рассчитано на администраторов баз данных, DevOps-инженеров и всех, кто хочет выстроить надёжную стратегию резервного копирования для кластеров QHB.

УДК 004.658:004.65
ББК 32.973.26-018.2

Содержание

Введение

Данные — это основной актив любой организации. Потеря базы данных из-за аппаратного сбоя, ошибки оператора, атаки шифровальщика или случайного удаления может привести к катастрофическим последствиям: финансовым потерям, нарушению работы сервисов и ущербу для репутации.

Стратегия резервного копирования — не опциональный элемент эксплуатации СУБД, а обязательное условие надёжной работы. Современные требования к доступности данных предполагают не только наличие бэкапов, но и возможность быстрого восстановления на любой момент времени (Point-in-Time Recovery, PITR), а также регулярную проверку целостности резервных копий.

Одна из самых популярных СУБД в мире — PostgreSQL, она применяется как в небольших веб-приложениях, так и в крупных корпоративных системах. На её основе АО «Концерн ГРАНИТ» разработал QHB («Квант-Гибрид»): российскую СУБД, полностью совместимую с PostgreSQL, входящую в реестр отечественного ПО и имеющую сертификат ФСТЭК России.

QHB, как и PostgreSQL, поставляется со встроенными средствами резервного копирования: `pg_dump` для логических дампов отдельных баз, `qhb_basebackup` для физического снимка всего кластера и ручное копирование файлов с настройкой WAL-архивирования. Этих инструментов достаточно для простых сценариев, но у них есть общее ограничение: каждый раз копируется весь объём данных целиком, а восстановление на произвольный момент времени требует ручной настройки.

В экосистеме QHB утилита **qbackup** — это инструмент резервного копирования промышленного уровня, разработанный специально для СУБД «Квант-Гибрид». Он расширяет стандартные возможности:

- **инкрементальное копирование** — только изменённые страницы данных, что существенно сокращает время и место на диске;
- **два режима копирования** — потоковый (stream, через протокол репликации) и прямой (direct, через файловую систему);
- **восстановление на момент времени (PITR)** — если настроено архивирование WAL;
- **проверка целостности** резервных копий командой `validate`.

Об архивировании WAL.
Непрерывное архивирование WAL-сегментов нужно для PITR и является стандартным инструментом QHB, не зависящим от топологии развёртывания. Настройка QLUSTER-кластера (с лидером и фолловерами) в этом руководстве не рассматривается. Рецепты ниже описывают работу с одиночным экземпляром QHB, если явно не указано иное.

Как подключаться к базе данных

Все SQL-команды в рецептах выполняются через `psql` — интерактивный клиент командной строки, поставляемый вместе с QHB. Если вы работаете прямо на сервере от пользователя `qhb`, достаточно запустить:

```
psql
```

Если команда psql не найдена, добавьте каталог в переменную окружения:

```
export PATH=/usr/local/qhb/bin:$PATH
```

Чтобы не выполнять эту команду при каждом входе, добавьте её в /.bashrc или /.bash_profile.

Если подключаетесь с другого хоста или под другим пользователем ОС, укажите параметры явно:

```
psql -U qhb -h localhost
```

Далее в рецептах блоки с SQL-командами подразумевают, что вы уже находитесь внутри `psql`. Команды, начинающиеся с обратного слэша (`\c`, `\l`, `\dt` и т. д.) — это встроенные команды самого `psql`, а не SQL.

Сквозной пример: городской справочник

Все рецепты в этом руководстве используют одну учебную базу данных — городской справочник. Она состоит из трёх таблиц которые содержат данные о районах города, улицах и организациях. Это позволяет проследить полный цикл: от создания данных через бэкап и «аварию» до восстановления.

Схема базы:

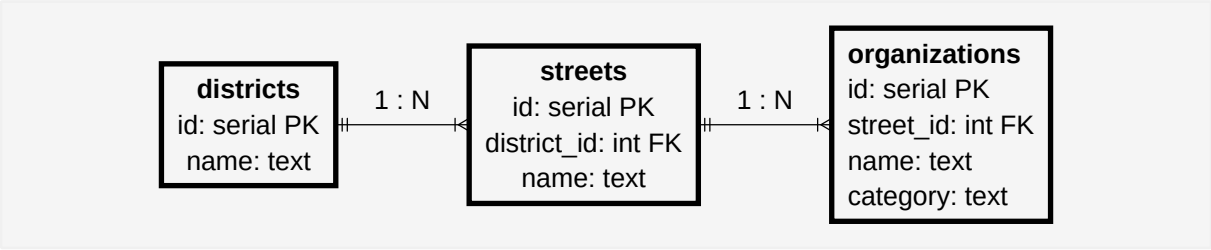


Схема 1. Схема БД

Создайте учебную базу данных

Подключитесь к серверу и создайте базу:

```
psql -U qhb
```

```
CREATE DATABASE city_directory;
\c city_directory
```

Команда `\c city_directory` переключает `psql` на только что созданную базу — теперь все последующие команды выполняются в ней.

Создайте таблицы:

```
CREATE TABLE districts (
    id   SERIAL PRIMARY KEY,
    name TEXT NOT NULL
);

CREATE TABLE streets (
    id          SERIAL PRIMARY KEY,
    district_id INTEGER REFERENCES districts(id),
    name        TEXT NOT NULL
);

CREATE TABLE organizations (
    id        SERIAL PRIMARY KEY,
    street_id INTEGER REFERENCES streets(id),
    name      TEXT NOT NULL,
    category  TEXT NOT NULL
);
```

Вставьте первую партию данных — районы и улицы:

```
INSERT INTO districts (name)
VALUES ('Центральный'),
       ('Северный'),
       ('Южный');

INSERT INTO streets (district_id, name)
```

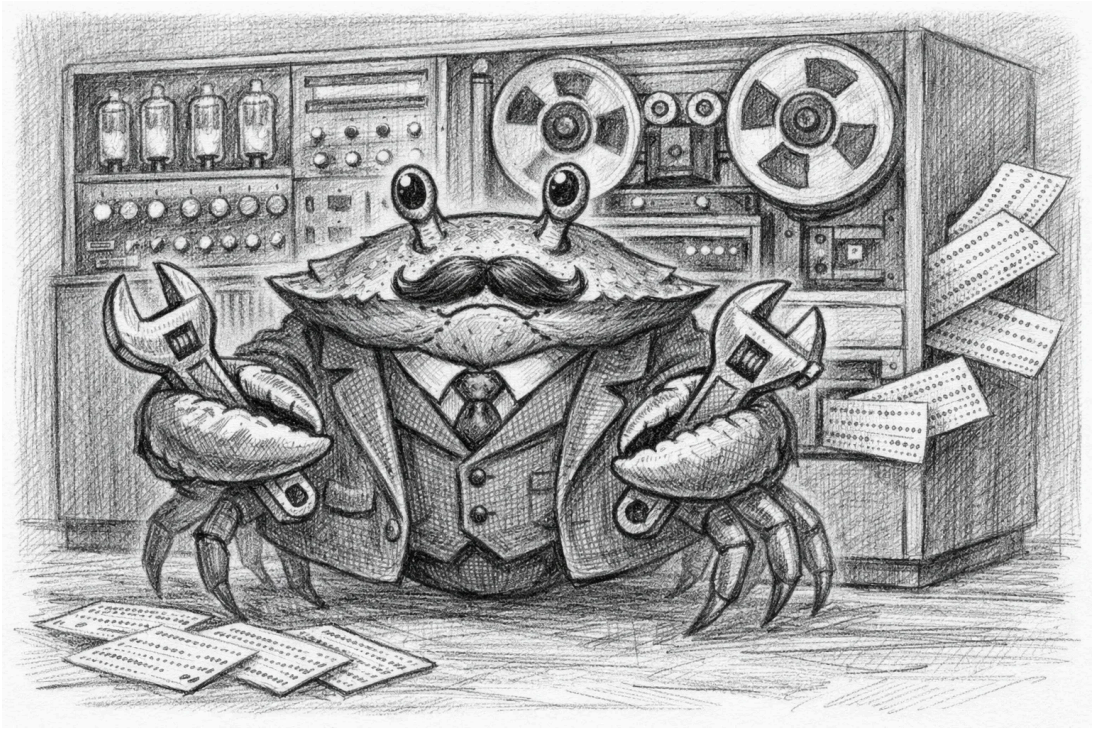
```
VALUES (1, 'ул. Ленина'),
       (1, 'ул. Советская'),
       (2, 'ул. Мира'),
       (2, 'ул. Победы'),
       (3, 'ул. Садовая'),
       (3, 'пр. Строителей');
```

Убедитесь, что данные на месте:

```
SELECT d.name AS район, s.name AS улица
  FROM streets s
       JOIN districts d ON d.id = s.district_id
 ORDER BY d.name, s.name;
```

```
    район    |      улица
─────────────┼──────────────────
 Северный    | ул. Мира
 Северный    | ул. Победы
 Центральный | ул. Ленина
 Центральный | ул. Советская
 Южный       | ул. Садовая
 Южный       | пр. Строителей
(6 строк)
```

База готова. Мы готовы установить qbackup и сделать первую резервную копию этих данных.



Рецепт 1. Установка и начальная настройка

Задача

Нет инструмента резервного копирования: нужно установить qbackup, подготовить место для хранения копий и настроить сервер QHB.

Решение

Установить пакет `qbackup` из репозитория Quantom, создать каталог для хранения резервных копий и задать минимально необходимые параметры в `qhb.conf`.

Как это сделать

Шаг 1. Добавьте репозиторий Quantom и установите пакет

Инструкции для всех поддерживаемых дистрибутивов приведены на (*https://repo.quantom.info/qhb/std-1/doc/Appendix/download.html*). Ниже — примеры для наиболее распространённых ОС.

RHEL / CentOS 8 / РЕД ОС 8 / РОСА:

```
rpm --import https://repo.quantom.info/qhb/keys/RPM-GPG-KEY-qhb
dnf install dnf-plugin-config-manager
dnf config-manager --add-repo \
    https://repo.quantom.info/qhb/std-1/centos/8/x86_64/qhb.repo
dnf install qbackup
```

Debian / Ubuntu / Astra Linux:

```
curl -fsSL https://repo.quantom.info/qhb/keys/RPM-GPG-KEY-qhb \
    | gpg --dearmor \
    | sudo tee /etc/apt/trusted.gpg.d/qhb.gpg > /dev/null
echo "deb https://repo.quantom.info/qhb/std-1/ubuntu/22.04/x86_64/ ./" \
    | sudo tee /etc/apt/sources.list.d/qhb.list
apt update && apt install qbackup
```

Проверьте установку:

```
qbackup --version
```

Шаг 2. Создайте каталог для хранения резервных копий

Резервные копии занимают существенный объём дискового пространства. Прежде чем выбирать место, оцените размер ваших баз данных и количество копий, которые планируете хранить.

Где хранить:

- **На том же сервере** — просто, но рискованно: при отказе диска вы теряете и базу, и бэкап. Подходит только для тестирования.
- **На отдельном диске на том же сервере** — лучше, но не защищает от полного выхода сервера из строя.
- **На другом хосте** — надёжнее всего. Об этом — отдельный рецепт в *главе 5*

Создайте каталог:

```
mkdir -p /opt/qhb_backups
```

О владельце каталога. При установке QHB в системе автоматически создаётся системный пользователь `qhb` — от его имени работает сервер базы данных. Первые рецепты этого руководства запускают qbackup именно от него: не нужно создавать дополнительных пользователей и настраивать права. Передайте каталог в его владение:

```
chown qhb:qhb /opt/qhb_backups
chmod 750 /opt/qhb_backups
```

Важно.
Каталог должен быть создан до первого запуска qbackup и обязан быть **пустым**. Ручное редактирование файлов внутри каталога не допускается — qbackup ведёт там свои метаданные.

Задайте переменную окружения, чтобы не указывать путь при каждом вызове:

```
export BACKUPS_DIR=/opt/qhb_backups
# Чтобы сохранить для всех сеансов пользователя qhb:
echo 'export BACKUPS_DIR=/opt/qhb_backups' >> ~qhb/.bashrc
```

Шаг 3. Настройте `qhb.conf`

qbackup умеет делать резервные копии двумя способами. **Прямой (direct) режим** читает файлы данных напрямую с диска — для этого нужен доступ к файловой системе сервера. **Потоковый (stream) режим** получает данные по сети через протокол репликации — тот же механизм, который QHB использует для синхронизации с резервными серверами. Это означает, что бэкап можно делать с любого хоста, имеющего сетевой доступ к серверу, не останавливая базу и не заходя на сам сервер.

Мы будем использовать потоковый режим — он удобнее и поддерживает инкрементальные бэкапы. Для его работы уровень WAL должен быть выше минимального:

```
# qhb.conf
wal_level = 'replica'
```

`wal_level` требует **перезапуска** сервера:

```
export PATH="/usr/local/qhb/bin:$PATH"
qhb_ctl restart -D $PGDATA
```

Примечание.
Переменная `$PGDATA` не задаётся автоматически при входе под пользователем `qhb`. Если она не определена в вашем окружении, укажите путь явно. Путь к каталогу данных зависит от способа установки QHB и может отличаться от стандартного. Уточнить фактический путь можно двумя способами — через запущенный процесс:

```
ps aux | grep "qhb -D"
```

или через конфигурацию systemd-сервиса:

```
systemctl cat qhb | grep PGDATA
```

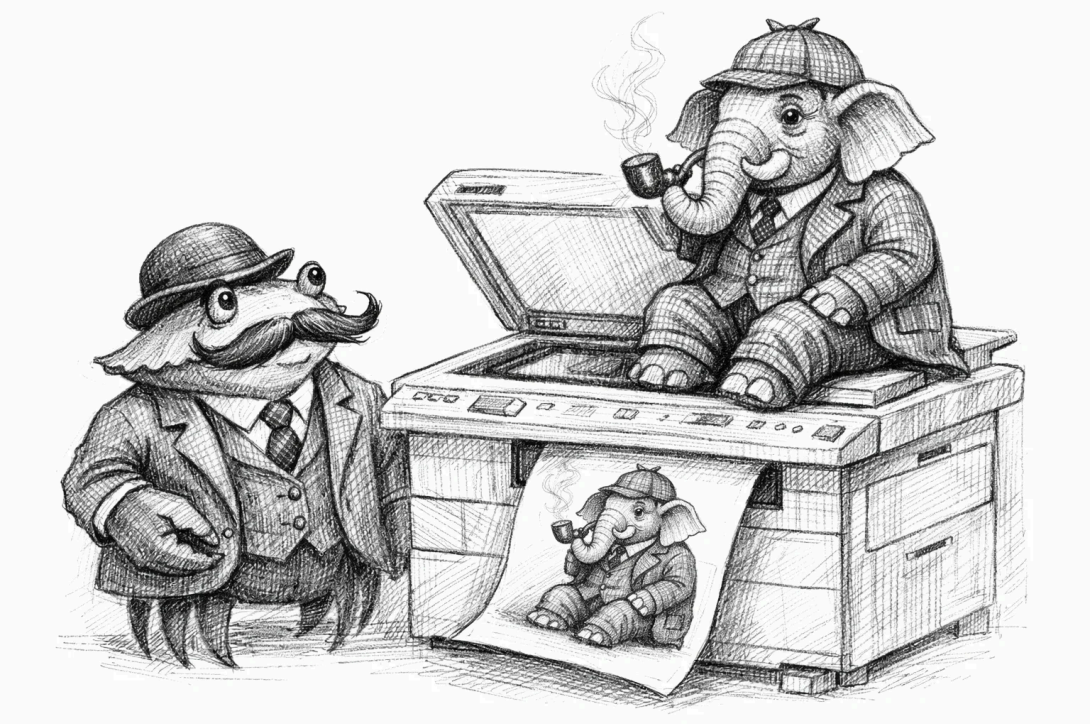
На этом начальная настройка завершена. CDC и роль `backup` понадобятся позже — в главе 6, когда перейдём к инкрементальным бэкапам.

Итоги

После выполнения этих шагов у вас есть:

- установленный qbackup;
- каталог `/opt/qhb_backups` с правильными правами;
- `qhb.conf` с уровнем WAL, достаточным для потокового копирования.

Теперь можно делать первый бэкап.



Рецепт 2. Первый полный бэкап

Задача

Нет резервной копии базы данных. Если сервер сейчас упадёт — данные будут потеряны безвозвратно.

Решение

Сделать первую полную резервную копию учебной базы командой `qbackup backup`.

Как это сделать

Шаг 1. Войдите под пользователем qhb

Команды qbackup выполняются от пользователя `qhb` — он владеет каталогом данных и каталогом резервных копий, поэтому дополнительных настроек прав не требуется:

```
su - qhb
```

Шаг 2. Запустите резервное копирование

```
qbackup backup \
    -B /opt/qhb_backups \
    -h localhost \
    -p 5432
```

Почему нет флага `-U`? Мы подключаемся от пользователя `qhb`, который является суперпользователем QHB — он имеет все необходимые права. Указывать роль явно не нужно. Как настроить отдельную роль с минимальными правами для регулярного автоматического копирования — рассказано в *главе об инкрементальных бэкапах*.

Почему нет флага `-D`? В stream-режиме qbackup получает данные по протоколу репликации, а не читает файлы напрямую. Флаг `-D` нужен только в direct-режиме.

qbackup выведет прогресс и сообщит об успешном завершении:

```
[2026-04-01 14:22:53][INFO] Запуск резервного копирования...
[2026-04-01 14:22:54][INFO] Стартовый LSN = 0/2000028
[2026-04-01 14:22:54][INFO] NOTICE: WAL archiving is not enabled; you must ensure
that all required WAL segments are copied through other means to complete the
backup
[2026-04-01 14:22:54][INFO] Конечный LSN = 0/2000138
[2026-04-01 14:22:54][INFO] Ожидание завершения потока WAL файлов
[2026-04-01 14:23:14][INFO] Размер несжатой резервной копии 58.95 MiB
[2026-04-01 14:23:14][INFO] Резервное копирование завершено, идентификатор:
MNFYINPC, контрольная сумма 14434293179986838863
```

Об уведомлении NOTICE.
Сообщение «WAL archiving is not enabled» появляется потому, что WAL-архивирование не настроено. Это нормально для первых рецептов — без него восстановление на произвольный момент времени (PITR) недоступно, но полный бэкап работает корректно. Настройка WAL-архивирования рассматривается в *главе о PITR*.

Шаг 3. Проверьте, что бэкап появился в каталоге

```
qbackup list -B /opt/qhb_backups
```

```
+---------------+---------+----------+-------------+-----------------+--------------+----------+---------------------------+---------------------------+-------------+
| идентификатор |  статус |   размер | сжатый размер |       тип       | стартовый lsn | родитель |          начало           |          конец            | комментарий |
+---------------+---------+----------+-------------+-----------------+--------------+----------+---------------------------+---------------------------+-------------+
|   LE4D8X48    | Завершён | 43.26 MiB |      -      | Полный (Потоковый) |  0/2000028   |    -     | 2024-03-16 12:00:00 +03:00 | 2024-03-16 12:04:32 +03:00 |     -       |
+---------------+---------+----------+-------------+-----------------+--------------+----------+---------------------------+---------------------------+-------------+
```

Запишите **ID** бэкапа (`LE4D8X48` в этом примере) — он понадобится при восстановлении. Обратите внимание на поля `status` (должен быть `Завершён`) и `начало` — по ним можно будет ориентироваться при выборе точки восстановления.

Примечание.
Размер резервной копии зависит от версии QHB и объёма системных таблиц кластера — в вашем случае он будет отличаться от приведённого в примере.

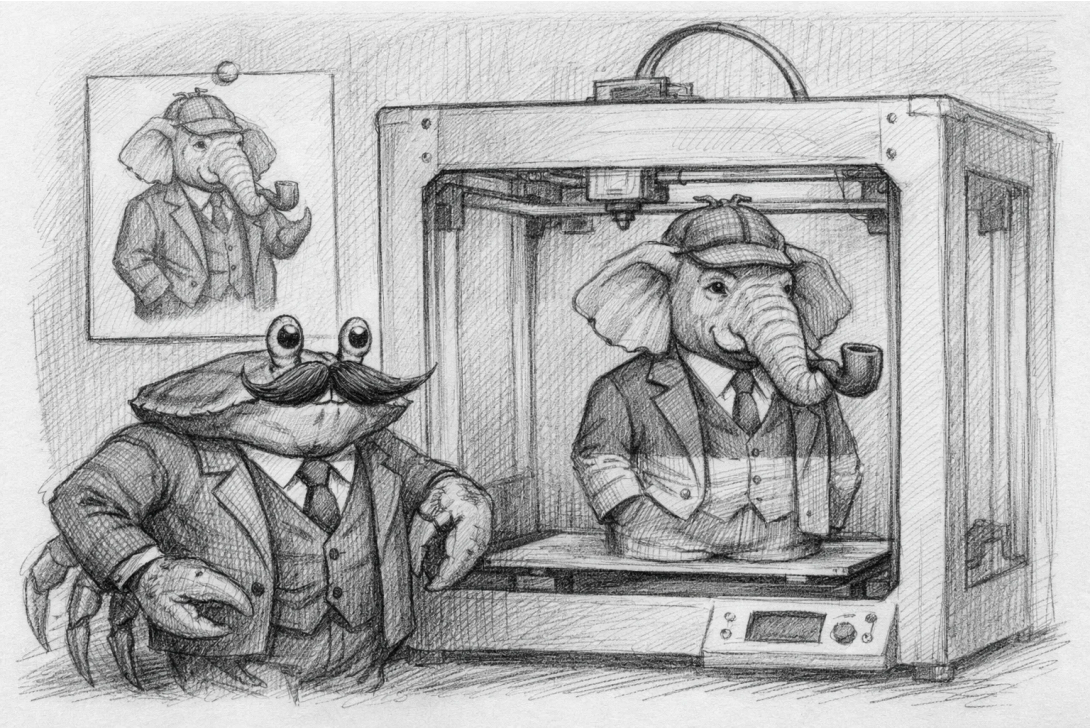
На этом первый бэкап завершён.

Итоги

После выполнения этих шагов у вас есть:

- учебная база данных с районами и улицами;
- полный бэкап этой базы в каталоге `/opt/qhb_backups`.

В следующей главе мы смоделируем аварию и восстановим базу из этой копии.



Рецепт 3. Восстановление из полного бэкапа

Задача

База данных уничтожена в результате аварии — случайное удаление, отказ диска, ошибка администратора. Нужно вернуть данные.

Решение

Восстановить кластер из полного бэкапа командой `qbackup restore`.

Как это сделать

Проверьте наличие бэкапа

Перед восстановлением убедитесь, что нужная копия есть в каталоге и её состояние `Завершён`:

```
qbackup list -B /opt/qhb_backups
```

```
+---------------+---------+-----------+---------------+--------------------+--------------+----------+---------------------------+---------------------------+-------------+
| идентификатор |  статус |   размер  | сжатый размер |        тип         | стартовый lsn | родитель |          начало           |          конец            | комментарий |
+---------------+---------+-----------+---------------+--------------------+--------------+----------+---------------------------+---------------------------+-------------+
|   LE4D8X48    | Завершён | 43.26 MiB |       -       | Полный (Потоковый) |  0/2000028   |    -     | 2024-03-16 12:00:00 +03:00 | 2024-03-16 12:04:32 +03:00 |      -      |
+---------------+---------+-----------+---------------+--------------------+--------------+----------+---------------------------+---------------------------+-------------+
```

Всё в порядке: бэкап есть, сделан в 12:00, размер 43 МБ, статус `Завершён`.

Смоделируйте аварию

Подключитесь к QHB и удалите базу:

```
psql -U qhb
```

```
DROP DATABASE city_directory;
```

Убедитесь, что база исчезла:

```
\l city_directory
```

```
List of databases
 Name | Owner | Encoding | Locale Provider | Collate | Ctype | ICU Locale | ICU Rules | Access privileges
------+-------+----------+-----------------+---------+-------+------------+-----------+-------------------
(0 rows)
```

Восстановите кластер

Важно. `qbackup restore` восстанавливает **весь кластер** целиком, а не отдельную базу данных. Каталог данных (`$PGDATA`) должен быть полностью очищен перед восстановлением.

Перед очисткой убедитесь, что `$PGDATA` указывает на нужный каталог. Если переменная не задана, пустая строка превратит команду в `rm -rf /*` — это уничтожит все данные на сервере. Узнать путь: `ps aux | grep "qhb -D"`.

Шаг 1. Остановите кластер

```
qhb_ctl stop -D $PGDATA -m fast
```

Шаг 2. Очистите каталог данных

```
# Укажите ваш путь к каталогу данных (пример — путь при установке по умолчанию)
PGDATA=/usr/local/qhb/data

rm -rf $PGDATA/*
```

Шаг 3. Запустите восстановление, подставив ваш ID бэкапа

```
qbackup restore \
    -B /opt/qhb_backups \
    -D $PGDATA \
    -i LE4D8X48
```

Шаг 4. Запустите кластер

```
qhb_ctl start -D $PGDATA
```

При запуске в выводе могут появиться сообщения об ошибках вида:

```
[ERROR] Ошибка восстановления WAL файла "/opt/qhb_backups/
wal/000000010000000000000002"
Caused by:
  -> Невозможно скопировать файл ... в "pg_wal/RECOVERYXLOG"
  -> No such file or directory (os error 2)
```

Почему это происходит.
После восстановления из бэкапа QHB запускает процедуру crash recovery и пытается применить WAL-сегменты через `restore_command`, который qbackup прописывает в конфигурацию автоматически. Поскольку WAL-архивирование не настроено, файлов в `/opt/qhb_backups/wal/` нет — отсюда ошибки.

Тем не менее кластер успешно восстанавливается до консистентного состояния на момент бэкапа и запускается. Строки `completed backup recovery` и `database system is ready to accept connections` подтверждают, что всё прошло штатно.

Эти ошибки исчезнут, когда будет настроено WAL-архивирование — об этом рассказано в *главе о PITR*.

Проверьте результат

Подключитесь к базе и убедитесь, что данные вернулись:

```
psql -U qhb
```

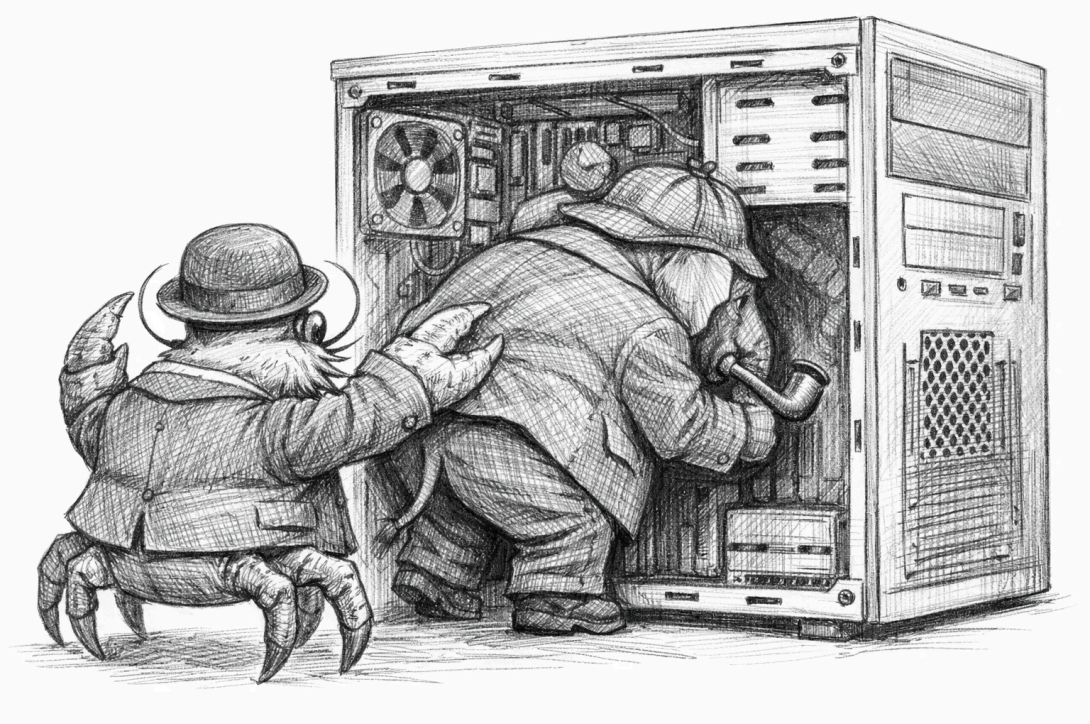
```
\c city_directory

SELECT d.name AS район, s.name AS улица
  FROM streets s
  JOIN districts d ON d.id = s.district_id
 ORDER BY d.name, s.name;
```

```
    район    |     улица
-------------+---------------
 Северный    | ул. Мира
 Северный    | ул. Победы
 Центральный | ул. Ленина
 Центральный | ул. Советская
 Южный       | пр. Строителей
 Южный       | ул. Садовая
(6 rows)
```

Все 3 района и 6 улиц восстановлены. Таблица `organizations` пуста — в момент бэкапа организации ещё не вносились, что вполне ожидаемо.

Итоги

Вы научились восстанавливать кластер QHB из полного бэкапа на тот же сервер. В следующей главе рассмотрим сценарий, когда восстанавливать нужно на другую машину.



Рецепт 4. Восстановление на новый сервер

Задача

Сервер базы данных вышел из строя. Каталог резервных копий уцелел — он находится на том же сервере (на отдельном диске) или доступен по сети (NAS, примонтированный раздел). Нужно восстановить кластер на новую машину.

Решение

Настроить SSH-доступ с машины, где хранится каталог бэкапов, на новый сервер и запустить `qbackup restore` с флагом `--connection`.

Как это сделать

Участники

- **`db-host`** — новый сервер, на котором будет развёрнут восстановленный кластер QHB. На нём уже установлены QHB и qbackup, но каталог данных пуст.

- **backup-host** — машина, на которой хранится каталог резервных копий `/opt/qhb_backups`. Именно отсюда запускается qbackup. Это может быть старый сервер (если он частично работоспособен), отдельная машина с примонтированным диском или NAS.

В примерах `db-host` — имя хоста или IP-адрес нового сервера. Замените на актуальное значение.

Предусловия

- На `db-host` установлены QHB и qbackup той же версии, что использовалась при создании бэкапа.
- qbackup на `db-host` доступен без указания полного пути (добавлен в `PATH`).
- Пользователь `qhb` существует на обоих серверах.
- Между серверами есть сетевой доступ по SSH.

Настройте SSH-доступ

qbackup подключается к `db-host` по SSH и копирует туда файлы. Для этого нужен беспарольный доступ от пользователя `qhb` на `backup-host` к пользователю `qhb` на `db-host`.

Шаг 1. Сгенерируйте SSH-ключ на backup-host

```
# Выполняется на backup-host от пользователя qhb
ssh-keygen -t ed25519 -f ~/.ssh/qhb_backup -N ""
```

Шаг 2. Скопируйте публичный ключ на db-host

```
ssh-copy-id -i ~/.ssh/qhb_backup.pub qhb@db-host
```

Шаг 3. Убедитесь, что qbackup на db-host доступен

```
ssh qhb@db-host qbackup --version
```

Если получили `command not found`, добавьте путь в `PATH` пользователя `qhb` на `db-host`:

```
ssh qhb@db-host "echo 'export PATH=/usr/local/qhb/bin:\$PATH' >> ~/.bashrc"
```

После этого повторите проверку.

Подготовьте db-host

Шаг 4. Убедитесь, что каталог данных на db-host пуст

```
ssh qhb@db-host "ls /var/lib/qhb/data/"
```

Если каталог не пуст — остановите кластер и очистите его:

```
ssh qhb@db-host "qhb_ctl stop -D /var/lib/qhb/data -m fast"
ssh qhb@db-host "rm -rf /var/lib/qhb/data/*"
```

Запустите восстановление

Шаг 5. Запустите qbackup restore с backup-host

```
qbackup restore \
    -B /opt/qhb_backups \
    -D /var/lib/qhb/data \
    -i LE4D8X48 \
    --connection "host=db-host user=qhb"
```

Подставьте ваш ID бэкапа вместо `LE4D8X48` — его можно посмотреть командой:

```
qbackup list -B /opt/qhb_backups
```

Шаг 6. Верните конфигурационные файлы на db-host

```
scp /tmp/qhb.conf.bak qhb@db-host:/var/lib/qhb/data/qhb.conf
scp /tmp/qhb_hba.conf.bak qhb@db-host:/var/lib/qhb/data/qhb_hba.conf
```

Шаг 7. Запустите кластер на db-host

```
ssh qhb@db-host "qhb_ctl start -D /var/lib/qhb/data"
```

Проверьте результат

```
psql -U qhb -h db-host
```

```
\c city_directory
SELECT count(*) FROM streets;

 count
-------
     6
(1 row)
```

Итоги

Мы восстановили кластер на новый сервер из локального каталога бэкапов. Этот сценарий подходит, когда каталог резервных копий хранится на том же сервере или доступен по сети.

Однако надёжнее хранить бэкапы на выделенной машине и делать их туда напрямую — так, чтобы потеря основного сервера вообще не затрагивала каталог копий. Об этом — следующий рецепт.

Рецепт 5. Выделенный сервер резервного копирования

Задача

Бэкап хранится на том же сервере, что и база данных. Если сервер полностью выходит из строя — теряется и база, и каталог резервных копий. Нужно выстроить надёжную схему: бэкапы должны храниться отдельно от данных.

Решение

Настроить выделенный сервер резервного копирования (`backup-host`), делать бэапы с сервера базы данных (`db-host`) напрямую туда и восстанавливать оттуда же.

Как это сделать

Участники

- **`db-host`** — сервер базы данных. На нём работает кластер QHB.

- **backup-host** — выделенный сервер резервного копирования. Здесь хранится каталог `/opt/qhb_backups` и запускается qbackup.

Оба сервера должны иметь сетевой доступ друг к другу.

Как это работает

В потоковом (stream) режиме qbackup подключается к QHB по **протоколу репликации** — тому же механизму, который используется для синхронизации с резервными серверами. Данные передаются по сети напрямую от `db-host` к `backup-host`. SSH при этом не нужен.

Важно.
SSH требуется только для удалённого **восстановления** (`restore --connection`), как в рецепте 4. Для удалённого **резервного копирования** в stream-режиме SSH не нужен — достаточно сетевого доступа к порту QHB.

Предусловия

- На `backup-host` установлен qbackup и создан каталог `/opt/qhb_backups` с правами пользователя `qhb`.
- В `qhb.conf` на `db-host` задан `wal_level = 'replica'` (выполнено в рецепте 1).
- В `qhb_hba.conf` на `db-host` разрешены репликационные подключения с `backup-host`.

Шаг 1. Разрешите репликационные подключения с backup-host

На `db-host` откройте `qhb_hba.conf` и добавьте строки, разрешающие пользователю `qhb` подключаться с `backup-host` для репликации:

```
# qhb_hba.conf на db-host
host  replication  qhb  <backup-host-ip>  trust
host  all          qhb  <backup-host-ip>  trust
```

Замените `<backup-host-ip>` на IP-адрес вашего `backup-host`. После изменения перезагрузите конфигурацию:

```
# На db-host
qhb_ctl reload -D $PGDATA
```

Шаг 2. Войдите на backup-host под пользователем qhb

```
# Выполняется на backup-host
su - qhb
```

Шаг 3. Запустите резервное копирование с backup-host

```
qbackup backup \
    -B /opt/qhb_backups \
```

```
    -h db-host \
    -p 5432
```

qbackup подключится к `db-host` по протоколу репликации и скопирует данные в каталог `/opt/qhb_backups` на `backup-host`. База данных продолжает работать в штатном режиме.

Шаг 4. Проверьте результат

```
qbackup list -B /opt/qhb_backups
```

```
+--------------+---------+------------+--------------+-------------------+--------------+----------+---------------------------+---------------------------+-------------+
| идентификатор |  статус  |   размер   | сжатый размер |        тип         | стартовый lsn | родитель |           начало          |           конец           | комментарий |
+--------------+---------+------------+--------------+-------------------+--------------+----------+---------------------------+---------------------------+-------------+
|   LE4D8X48    | Завершён | 43.26 MiB  |       -      | Полный (Потоковый) |  0/2000028   |    -     | 2024-03-16 12:00:00 +03:00 | 2024-03-16 12:04:32 +03:00 |      -      |
+--------------+---------+------------+--------------+-------------------+--------------+----------+---------------------------+---------------------------+-------------+
```

Бэкап создан на `backup-host` и физически не зависит от состояния `db-host`. Если основной сервер выйдет из строя, каталог резервных копий останется нетронутым.

Восстановление с backup-host на db-host

Для восстановления из каталога на `backup-host` на сервер `db-host` используйте схему из рецепта 4: настройте SSH-доступ и запустите `qbackup restore --connection`. Каталог бэкапов уже там, где надо.

Итоги

Вы перенесли резервное копирование на выделенный сервер. Теперь бэкапы надёжно отделены от данных: `backup-host` запускает qbackup по сети, забирает данные с `db-host` напрямую по протоколу репликации, и хранит их у себя.

В следующем рецепте рассмотрим инкрементальные бэкапы — как копировать только изменившиеся данные, а не весь кластер целиком.

Рецепт 6. Инкрементальные бэкапы и восстановление из цепочки

Задача

Полный бэкап копирует весь кластер — даже если за прошедшие сутки изменился лишь один процент данных. При большом размере базы это долго и требует много места. Нужно копировать только то, что изменилось.

Решение

Включить CDC (Change Data Capture) в `qhb.conf`, создать роль `backup` с правами репликации, сделать новую полную копию — и дальше запускать `qbackup backup --incremental`. Каждый инкремент содержит только изменившиеся страницы данных.

Как это сделать

Как работают инкрементальные бэкапы

Для отслеживания изменённых страниц qbackup использует механизм CDC.

Он запоминает, какие страницы данных были записаны с момента последнего бэкапа. При инкрементальном копировании в резервную копию попадают только эти страницы. При восстановлении qbackup автоматически применяет полную копию и все инкременты в правильном порядке — читателю не нужно управлять этим вручную.

Шаг 1. Создайте роль backup в базе данных

До этого рецепта мы запускали qbackup от пользователя `qhb` без явного указания роли. Для регулярного автоматического копирования лучше выделить роль с минимально необходимыми правами.

```
psql -U qhb
```

```
CREATE USER backup REPLICATION PASSWORD 'secret';
```

Замените `'secret'` на надёжный пароль в вашей системе.

Шаг 2. Разрешите подключение роли backup в qhb_hba.conf

Откройте `qhb_hba.conf` и добавьте:

```
# qhb_hba.conf
host  replication  backup  127.0.0.1/32  md5
host  all          backup  127.0.0.1/32  md5
```

Если qbackup будет запускаться с другого хоста, укажите его IP вместо `127.0.0.1/32`.

Перезагрузите конфигурацию:

```
qhb_ctl reload -D $PGDATA
```

Шаг 3. Создайте файл паролей .qhbpass

Чтобы qbackup подключался без интерактивного ввода пароля, создайте файл паролей:

```
# От пользователя qhb
echo "localhost:5432:replication:backup:secret" >> ~/.qhbpass
echo "localhost:5432:*:backup:secret" >> ~/.qhbpass
chmod 600 ~/.qhbpass
```

Шаг 4. Включите CDC в qhb.conf

Откройте `qhb.conf` и добавьте параметр `qcdc_registry_size`. Его значение выбирается из расчёта: предполагаемый размер кластера / 1000. Минимум — 1 МБ, максимум — 10 240 МБ.

```
# qhb.conf
qcdc_registry_size = 30MB
```

Параметр требует **перезапуска** сервера:

```
qhb_ctl restart -D $PGDATA
```

После перезапуска в каталоге данных появится файл `qhb-cdc.map` — это карта изменённых страниц.

Шаг 5. Сделайте новую полную копию

Важно.
Бэкап из рецепта 2 не подходит для инкрементов — он был сделан до включения CDC. Необходима новая полная копия.

Перед созданием новой копии добавим вторую партию данных в нашу учебную базу — больницы и поликлиники:

```
psql -U qhb
```

```
\c city_directory

INSERT INTO organizations (street_id, name, category)
VALUES (1, 'Городская больница №1', 'больница'),
       (2, 'Поликлиника №3',       'поликлиника'),
       (3, 'Городская больница №2', 'больница'),
       (4, 'Поликлиника №7',       'поликлиника');
```

Теперь сделайте новую полную резервную копию от имени роли `backup`:

```
su - qhb
qbackup backup \
    -B /opt/qhb_backups \
    -h localhost \
    -p 5432 \
    -u backup \
    --db-name qhb
```

Почему `--db-name qhb`?

qbackup всегда резервирует **весь кластер** — каталог данных `PGDATA` целиком со всеми базами. Отдельную базу выбрать нельзя.

Однако перед запуском потокового копирования qbackup устанавливает обычное SQL-соединение с сервером, чтобы считать метаданные (версию, настройки, стартовый LSN). По умолчанию QHB подключается к базе данных, имя которой совпадает с именем пользователя. Поскольку базы `backup` не существует, без явного указания команда завершится ошибкой:

```
FATAL: database "backup" does not exist
```

Флаг `--db-name` задаёт базу только для этого начального соединения. На состав и содержимое резервной копии он не влияет.

Запишите ID новой полной копии — он понадобится в следующих шагах:

```
qbackup list -B /opt/qhb_backups
```

```
+--------------+----------+-----------+--------------+-----------------+--------------+----------+---------------------------+---------------------------+------------+
| идентификатор |  статус  |   размер  | сжатый размер |       тип       | стартовый lsn | родитель |          начало           |          конец            | комментарий |
+--------------+----------+-----------+--------------+-----------------+--------------+----------+---------------------------+---------------------------+------------+
|   LE4D8X48   | Завершён | 43.26 MiB |      -       | Полный (Потоковый) |  0/2000028   |    -     | 2024-03-16 12:00:00 +03:00 | 2024-03-16 12:04:32 +03:00 |     -      |
+--------------+----------+-----------+--------------+-----------------+--------------+----------+---------------------------+---------------------------+------------+
|   MF2A1K9Q   | Завершён | 45.80 MiB |      -       | Полный (Потоковый) |  0/4000028   |    -     | 2024-03-17 09:00:00 +03:00 | 2024-03-17 09:04:15 +03:00 |     -      |
+--------------+----------+-----------+--------------+-----------------+--------------+----------+---------------------------+---------------------------+------------+
```

Новая полная копия `MF2A1K9Q` содержит 3 района, 6 улиц и 4 медицинских организации.

Шаг 6. Добавьте третью партию данных и сделайте первый инкремент

Добавьте школы и детские сады:

```
psql -U qhb
```

```
\c city_directory

INSERT INTO organizations (street_id, name, category)
VALUES (1, 'Школа №5',          'школа'),
       (3, 'Школа №12',         'школа'),
       (5, 'Детский сад №2',    'детский сад'),
       (6, 'Детский сад №8',    'детский сад');
```

Сделайте первый инкрементальный бэкап:

```
qbackup backup \
    -B /opt/qhb_backups \
    -h localhost \
    -p 5432 \
    -u backup \
```

```
    --db-name qhb \
    --incremental
```

`--db-name qhb` здесь по той же причине, что и в шаге 5: флаг задаёт базу для начального SQL-соединения. Без него qbackup попытается подключиться к базе `backup` и завершится с ошибкой.

Шаг 7. Посмотрите цепочку копий

```
qbackup list -B /opt/qhb_backups
```

```
+---------------+----------+------------+--------------+-----------------------------+--------------+----------+----------------------+----------------------+-------------+
| идентификатор |  статус  |   размер   | сжатый размер |            тип              | стартовый lsn | родитель |       начало        |        конец         | комментарий |
+---------------+----------+------------+--------------+-----------------------------+--------------+----------+----------------------+----------------------+-------------+
|   LE4D8X48    | Завершён | 43.26 MiB  |      -       |     Полный (Потоковый)      |  0/2000028   |    -     | 2024-03-16 12:00:00  | 2024-03-16 12:04:32  |      -      |
+---------------+----------+------------+--------------+-----------------------------+--------------+----------+----------------------+----------------------+-------------+
|   MF2A1K9Q    | Завершён | 45.80 MiB  |      -       |     Полный (Потоковый)      |  0/4000028   |    -     | 2024-03-17 09:00:00  | 2024-03-17 09:04:15  |      -      |
+---------------+----------+------------+--------------+-----------------------------+--------------+----------+----------------------+----------------------+-------------+
|   MF2B3R7T    | Завершён |  1.24 MiB  |      -       | Инкрементальный (Потоковый) |  0/6000028   | MF2A1K9Q | 2024-03-17 18:00:00  | 2024-03-17 18:00:08  |      -      |
+---------------+----------+------------+--------------+-----------------------------+--------------+----------+----------------------+----------------------+-------------+
```

Обратите внимание на столбец `родитель`: инкрементальная копия `MF2B3R7T` ссылается на полную `MF2A1K9Q`. Размер инкремента — 1.24 МБ против 45.80 МБ для полной копии.

Восстановление из последнего сохранения

Смоделируем аварию и восстановим базу из последней инкрементальной копии — она содержит все 8 организаций.

```
psql -U qhb
```

```
DROP DATABASE city_directory;
```

Важно.
Перед очисткой убедитесь, что `$PGDATA` указывает на нужный каталог. Если переменная не задана, пустая строка превратит команду в `rm -rf /*` — это уничтожит все данные на сервере. Узнать путь: `ps aux | grep "qhb -D"`.

```
# Укажите ваш путь к каталогу данных (пример – путь при установке по умолчанию)
PGDATA=/usr/local/qhb/data

qhb_ctl stop -D $PGDATA -m fast
cp $PGDATA/qhb.conf /tmp/qhb.conf.bak
cp $PGDATA/qhb_hba.conf /tmp/qhb_hba.conf.bak
rm -rf $PGDATA/*
```

Запустите восстановление, указав ID **инкрементальной** копии — qbackup автоматически найдёт и применит всю цепочку:

```
qbackup restore \
    -B /opt/qhb_backups \
    -D $PGDATA \
    -i MF2B3R7T
```

```
cp /tmp/qhb.conf.bak $PGDATA/qhb.conf
cp /tmp/qhb_hba.conf.bak $PGDATA/qhb_hba.conf
qhb_ctl start -D $PGDATA
```

Проверьте результат:

```
psql -U qhb
```

```
\c city_directory
SELECT name, category FROM organizations ORDER BY category, name;
```

```
         name          |  category
-----------------------+-------------
 Городская больница №1 | больница
 Городская больница №2 | больница
 Детский сад №2        | детский сад
 Детский сад №8        | детский сад
 Поликлиника №3        | поликлиника
 Поликлиника №7        | поликлиника
 Школа №12             | школа
 Школа №5              | школа
(8 rows)
```

Все 8 организаций вернулись.

Восстановление из промежуточного инкремента

Иногда нужно восстановить данные не до последней копии, а до более ранней точки — например, если после инкремента были внесены ошибочные изменения.

Восстановите базу из полной копии `MF2A1K9Q` — в ней только 4 медицинских организации, без школ:

Важно.
Перед очисткой убедитесь, что `$PGDATA` указывает на нужный каталог. Если переменная не задана, пустая строка превратит команду в `rm -rf /*` — это уничтожит все данные на сервере. Узнать путь: `ps aux | grep "qhb -D"`.

```
# Укажите ваш путь к каталогу данных (пример – путь при установке по умолчанию)
PGDATA=/usr/local/qhb/data

qhb_ctl stop -D $PGDATA -m fast
cp $PGDATA/qhb.conf /tmp/qhb.conf.bak
cp $PGDATA/qhb_hba.conf /tmp/qhb_hba.conf.bak
rm -rf $PGDATA/*

qbackup restore \
    -B /opt/qhb_backups \
    -D $PGDATA \
    -i MF2A1K9Q

cp /tmp/qhb.conf.bak $PGDATA/qhb.conf
cp /tmp/qhb_hba.conf.bak $PGDATA/qhb_hba.conf
qhb_ctl start -D $PGDATA
```

```
psql -U qhb
```

```
\c city_directory
SELECT count(*) FROM organizations;
```

```
 count
-------
     4
(1 row)
```

Только больницы и поликлиники — данные на момент полной копии.

Итоги

Вы научились:

- создавать роль `backup` с правами репликации;
- включать CDC (`qcdc_registry_size`) для отслеживания изменений;
- делать инкрементальные бэкапы флагом `--incremental`;
- читать цепочку копий в `qbackup list` по столбцу `родитель`;
- восстанавливать как из последнего инкремента, так и из любой промежуточной точки.

Инкрементальные копии существенно экономят место и время. Следующий шаг — научиться восстанавливать базу не до момента бэкапа, а до конкретной секунды. Это возможно с WAL-архивированием — тема следующего рецепта.

Рецепт 7. Восстановление на момент времени (PITR)

Задача

Авария произошла не в момент последнего бэкапа — кто-то удалил таблицу десять минут назад, и нужно вернуть состояние базы именно до этого момента, а не на час назад. Обычное восстановление из резервной копии тут не поможет.

Решение

Настроить WAL-архивирование с помощью `qbackup backup-wal`, а затем при восстановлении указать параметр `--target-time`. qbackup восстановит базу из бэкапа и воспроизведёт WAL-файлы ровно до указанной секунды.

Как это сделать

Как работает PITR

При обычном восстановлении qbackup восстанавливает файлы кластера на момент бэкапа. Если включить WAL-архивирование, QHB будет копировать все сегменты WAL по мере их

заполнения — в тот же каталог бэкапов. При восстановлении с параметрами точки qbackup сначала восстанавливает базовую копию, а затем последовательно воспроизводит WAL-сегменты до указанной метки. Это позволяет восстановиться в любой момент между бэкапами.

Шаг 1. Включите WAL-архивирование в qhb.conf

Откройте `qhb.conf` и добавьте или измените следующие параметры:

```
# qhb.conf
wal_level = 'replica'
archive_mode = on
archive_command = '/usr/bin/qbackup backup-wal -B /opt/qhb_backups -f %f -p %p'
```

Примечание.
Параметр `archive_mode` и `archive_command` требуют **перезапуска** сервера. Убедитесь, что пути в `archive_command` соответствуют вашей системе. Если qbackup установлен в другое место, укажите полный путь к исполняемому файлу.

Перезапустите сервер:

```
qhb_ctl restart -D $PGDATA
```

После перезапуска QHB начнёт архивировать WAL-сегменты в подкаталог `wal/` внутри `/opt/qhb_backups`. Подкаталог создаётся автоматически.

Шаг 2. Сделайте базовую резервную копию

После включения WAL-архивирования необходимо сделать новую полную резервную копию — она станет точкой отсчёта для PITR:

```
su - qhb
qbackup backup \
    -B /opt/qhb_backups \
    -h localhost \
    -p 5432 \
    -u backup \
    --db-name qhb
```

Примечание.
Флаг `--db-name qhb` задаёт базу для начального SQL-соединения qbackup. Он необходим, потому что базы данных с именем `backup` не существует — без этого флага команда завершится ошибкой `FATAL: database "backup" does not exist`. На состав резервной копии флаг не влияет.

Запишите ID новой копии:

```
qbackup list -B /opt/qhb_backups
```

```
+---------------+---------+-----------+---------------+---------------------------+---------------+----------+----------------------+----------------------+-------------+
| идентификатор |  статус |   размер  | сжатый размер |            тип            | стартовый lsn | родитель |        начало        |        конец         | комментарий |
+---------------+---------+-----------+---------------+---------------------------+---------------+----------+----------------------+----------------------+-------------+
|   NP3K7M2A    | Завершён | 46.12 MiB |       -       |    Полный (Потоковый)     |  0/8000028    |    -     | 2024-03-18 09:00:00  | 2024-03-18 09:04:20  |      -      |
+---------------+---------+-----------+---------------+---------------------------+---------------+----------+----------------------+----------------------+-------------+
```

Шаг 3. Зафиксируйте время перед «аварией»

Работаем в нашей учебной базе `city_directory`. Сначала посмотрим текущее время — это будет наша точка восстановления:

```
psql -U qhb
```

```
\c city_directory
SELECT now();
```

```
              now
-------------------------------
 2024-03-18 10:15:32.441+03
(1 row)
```

Запишите это время. В нашем примере это `2024-03-18 10:15:32+03`.

Шаг 4. Смоделируйте аварию

Удалите таблицу `organizations` — «случайная» авария:

```
DROP TABLE organizations;
\dt
```

```
          List of relations
 Schema |   Name    | Type  | Owner
--------+-----------+-------+-------
 public | districts | table | qhb
 public | streets   | table | qhb
(2 rows)
```

Таблица `organizations` исчезла. Зафиксируем время «аварии»:

```
SELECT now();
```

```
              now
-------------------------------
 2024-03-18 10:17:05.882+03
(1 row)
```

Шаг 5. Восстановите базу до момента времени

Остановите сервер и очистите каталог данных:

Важно.
Перед очисткой убедитесь, что `$PGDATA` указывает на нужный каталог. Если переменная не задана, пустая строка превратит команду в `rm -rf /*` — это уничтожит все данные на сервере. Узнать путь: `ps aux | grep "qhb -D"`.

```
# Укажите ваш путь к каталогу данных (пример — путь при установке по умолчанию)
PGDATA=/usr/local/qhb/data

qhb_ctl stop -D $PGDATA -m fast
cp $PGDATA/qhb.conf /tmp/qhb.conf.bak
cp $PGDATA/qhb_hba.conf /tmp/qhb_hba.conf.bak
rm -rf $PGDATA/*
```

Запустите восстановление с указанием времени — чуть раньше момента аварии:

```
qbackup restore \
    -B /opt/qhb_backups \
    -D $PGDATA \
    -i NP3K7M2A \
    --target-time='2024-03-18 10:15:45+03'
```

Важно.
В `--target-time` укажите время **после** `SELECT now()` но **до** команды `DROP TABLE`. В примере авария произошла в 10:17, поэтому любое время в диапазоне 10:15–10:16 подходит.

Верните конфиги и запустите сервер:

```
cp /tmp/qhb.conf.bak $PGDATA/qhb.conf
cp /tmp/qhb_hba.conf.bak $PGDATA/qhb_hba.conf
qhb_ctl start -D $PGDATA
```

Шаг 6. Проверьте результат

```
psql -U qhb
```

```
\c city_directory
\dt
```

```
           List of relations
 Schema |     Name      | Type  | Owner
--------+---------------+-------+-------
 public | districts     | table | qhb
 public | organizations | table | qhb
 public | streets       | table | qhb
(3 rows)
```

```
SELECT count(*) FROM organizations;
```

```
 count
-------
     8
(1 row)
```

Таблица `organizations` вернулась со всеми 8 записями — данные восстановлены на момент **до** аварии.

Другие варианты точки восстановления

Помимо `--target-time`, qbackup поддерживает несколько способов задать точку восстановления:

По LSN — восстановление до конкретной позиции в WAL:

```
qbackup restore -B /opt/qhb_backups -D $PGDATA \
    -i NP3K7M2A \
    --target-lsn=0/B374D848
```

До последнего LSN в архиве — максимально возможное восстановление:

```
qbackup restore -B /opt/qhb_backups -D $PGDATA \
    -i NP3K7M2A \
    --target-lsn=latest
```

По идентификатору транзакции:

```
qbackup restore -B /opt/qhb_backups -D $PGDATA \
    -i NP3K7M2A \
    --target-xid=1542
```

По именованной точке восстановления, созданной через `SELECT pg_create_restore_point('before_migration')`:

```
qbackup restore -B /opt/qhb_backups -D $PGDATA \
    -i NP3K7M2A \
    --target-name='before_migration'
```

Параметр –action

После достижения точки восстановления qbackup может вести себя тремя способами — в зависимости от значения `--action`:

- `--action=promote` — сервер завершает восстановление и сразу принимает подключения. **Рекомендуется** для большинства сценариев.

- `--action=pause` — сервер приостанавливается и принимает только запросы на чтение. Это позволяет проверить состояние базы прежде, чем разрешить пользователям подключаться. Для снятия с паузы выполните:

```
SELECT pg_wal_replay_resume();
```

- `--action=shutdown` — сервер останавливается сразу после достижения точки.

Примечание.
По умолчанию используется `--action=pause`. Однако `pg_wal_replay_resume()` требует наличия в архиве следующего WAL-сегмента после точки восстановления — иначе сервер зависнет в ожидании. Чтобы избежать этой ситуации, передавайте `--action=promote` явно:

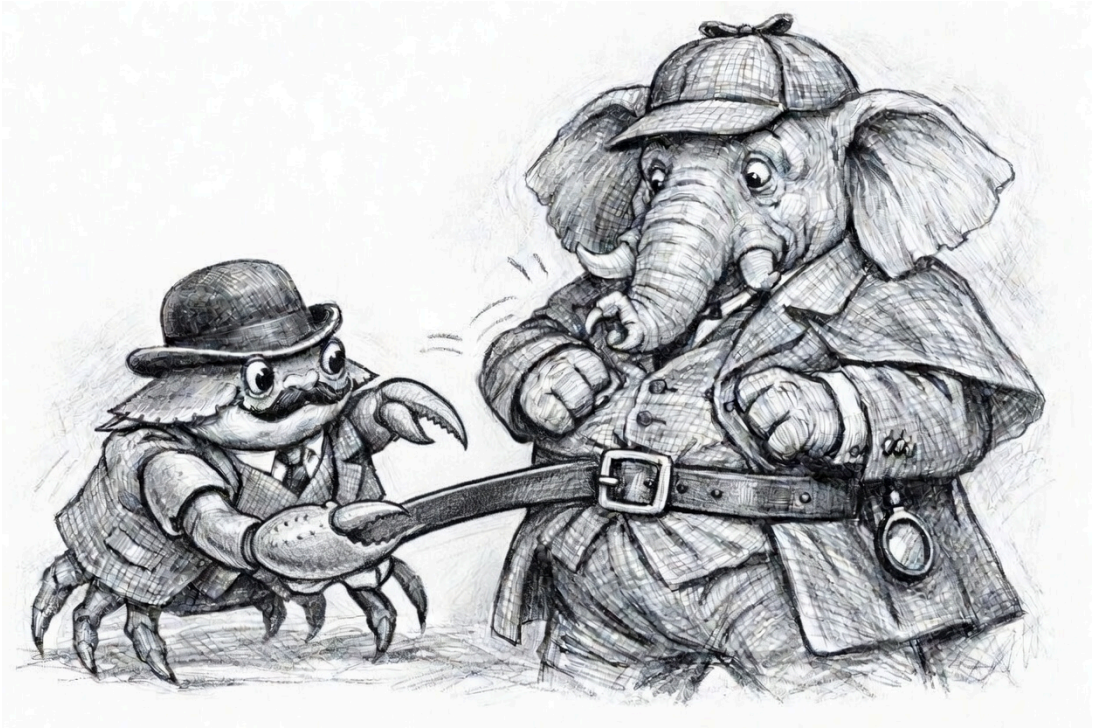
```
qbackup restore \
    -B /opt/qhb_backups \
    -D $PGDATA \
    -i NP3K7M2A \
    --target-time='2024-03-18 10:15:45+03' \
    --action=promote
```

Итоги

Вы научились:

- настраивать непрерывное WAL-архивирование через `archive_command = 'qbackup backup-wal ...'`;
- восстанавливать базу до конкретного момента времени с помощью `--target-time`;
- использовать альтернативные точки восстановления: `--target-lsn`, `--target-xid`, `--target-name`;
- управлять поведением сервера после восстановления через `--action`.

PITR — мощный инструмент, но он требует непрерывно работающего WAL-архива. Следующий рецепт покажет, как убедиться, что ваши бэкапы целостны и пригодны к восстановлению — до того, как это понадобится по-настоящему.



Рецепт 8. Управление каталогом резервных копий

Задача

Каталог резервных копий растёт с каждым бэкапом. Через несколько месяцев накопятся десятки копий, займут гигабайты места, а поиск нужной копии станет неудобным. Нужен порядок.

Решение

Использовать команду `qbackup remove` с флагами `--hold`, `--single`, `--dry-run` для удаления устаревших копий, а также `qbackup remove-wal` для очистки архивных WAL-сегментов. Дополнительно — форматы `--raw` и `--json` для интеграции со скриптами мониторинга.

Как это сделать

Форматы вывода qbackup list

По умолчанию `qbackup list` выводит таблицу в человекочитаемом виде. Но для скриптов и автоматизации удобнее другие форматы.

Стандартный вывод (таблица):

```
qbackup list -B /opt/qhb_backups
```

```
+----------+--------+----------+-----------------+---------------------+----------+----------+--------------------------+--------------------------+---------+
|    ID    | status |   size   | compressed size |        kind         | start lsn |  parent  |        start time        |         end time         | comment |
+----------+--------+----------+-----------------+---------------------+----------+----------+--------------------------+--------------------------+---------+
| LE4D8X48 |  Done  | 43.26MiB |        –        |    Full (Stream)    | 0/2000028 |    –     | 2024-03-16 12:00:00 +03:00 | 2024-03-16 12:04:32 +03:00 |    –    |
+----------+--------+----------+-----------------+---------------------+----------+----------+--------------------------+--------------------------+---------+
| MF2A1K9Q |  Done  | 45.80MiB |        –        |    Full (Stream)    | 0/4000028 |    –     | 2024-03-17 09:00:00 +03:00 | 2024-03-17 09:04:15 +03:00 |    –    |
+----------+--------+----------+-----------------+---------------------+----------+----------+--------------------------+--------------------------+---------+
| MF2B3R7T |  Done  |  1.24MiB |        –        | Incremental (Stream) | 0/6000028 | MF2A1K9Q | 2024-03-17 18:00:00 +03:00 | 2024-03-17 18:00:08 +03:00 |    –    |
+----------+--------+----------+-----------------+---------------------+----------+----------+--------------------------+--------------------------+---------+
| NP3K7M2A |  Done  | 46.12MiB |        –        |    Full (Stream)    | 0/8000028 |    –     | 2024-03-18 09:00:00 +03:00 | 2024-03-18 09:04:20 +03:00 |    –    |
+----------+--------+----------+-----------------+---------------------+----------+----------+--------------------------+--------------------------+---------+
```

Формат –raw — разделители пробелами, удобен для обработки через `awk`:

```
qbackup list -B /opt/qhb_backups --raw
```

```
id        status  size       compressed_size  kind                start_lsn
parent    start_time                  end_time
LE4D8X48  done    45364828   -                full stream         0/2000028   -
2024-03-16 12:00:00 +03:00  2024-03-16 12:04:32 +03:00
MF2A1K9Q  done    48025600   -                full stream         0/4000028   -
2024-03-17 09:00:00 +03:00  2024-03-17 09:04:15 +03:00
MF2B3R7T  done    1300480    -                incremental stream  0/6000028
MF2A1K9Q  2024-03-17 18:00:00 +03:00  2024-03-17 18:00:08 +03:00
NP3K7M2A  done    48367616   -                full stream         0/8000028   -
2024-03-18 09:00:00 +03:00  2024-03-18 09:04:20 +03:00
```

Формат –json — для интеграции с системами мониторинга и скриптами:

```
qbackup list -B /opt/qhb_backups --json
```

```
[
  {
    "identifier": "NP3K7M2A",
    "status": "Done",
    "size": 48367616,
    "compressed_size": null,
    "kind": { "Full": "Stream" },
    "start_lsn": "0/8000028",
    "start_time": "2024-03-18T09:00:00+03:00",
    "end_time": "2024-03-18T09:04:20+03:00",
    "parent": null,
    "comment": null
  }
]
```

Шаг 1. Посмотрите, что будет удалено, без реального удаления

Прежде чем удалять, используйте флаг `--dry-run` — он покажет, какие копии будут затронуты, но ничего не удалит:

```
qbackup remove \
    -B /opt/qhb_backups \
    -i LE4D8X48 \
    --dry-run
```

```
The following backups will be removed:
  LE4D8X48  Full (Stream)  2024-03-16 12:00:00
```

Шаг 2. Удалите старую полную копию вместе с зависимыми инкрементами

Команда `qbackup remove` по умолчанию удаляет указанную копию и все инкрементальные копии, которые от неё зависят. Перед удалением она выводит список и просит подтверждение.

```
qbackup remove \
    -B /opt/qhb_backups \
    -i LE4D8X48
```

```
The following backups will be removed:
  LE4D8X48  Full (Stream)  2024-03-16 12:00:00
Proceed? [y/N] y
Backup LE4D8X48 removed.
```

Чтобы пропустить интерактивный вопрос (например, в скриптах), добавьте флаг `-y`:

```
qbackup remove -B /opt/qhb_backups -i LE4D8X48 -y
```

Шаг 3. Удалите только один инкремент, не трогая цепочку

Иногда нужно удалить конкретный инкремент, не удаляя те, что зависят от него. Флаг `--single` удаляет только указанную копию:

```
qbackup remove \
    -B /opt/qhb_backups \
    -i MF2B3R7T \
    --single
```

Внимание.
После удаления копии из цепочки все последующие инкрементальные копии, которые на неё опираются, становятся недействительными. Убедитесь, что они вам больше не нужны, или используйте `--dry-run` для предварительной проверки.

Шаг 4. Настройте политику удержания

Вместо ручного удаления по ID используйте политику удержания — она автоматически удалит все копии старше указанного числа дней. Для нашего примера оставим копии за последние 7 дней:

```
qbackup remove \
    -B /opt/qhb_backups \
    --hold 7
```

qbackup удалит все копии, созданные до `текущая_дата – 7 дней`. Если старая копия является частью цепочки с более новой — она не будет удалена до тех пор, пока вся цепочка не устареет.

Рекомендуемая стратегия: запускать эту команду в cron ежедневно, например:

```
# /etc/cron.d/qbackup-cleanup
0 3 * * * qhb /usr/bin/qbackup remove -B /opt/qhb_backups --hold 30 -y
```

Шаг 5. Удалите устаревшие WAL-сегменты

После удаления старых бэкапов их WAL-сегменты могут оставаться в каталоге `wal/` и занимать место. Очистите их командой `remove-wal`:

```
qbackup remove-wal \
    -B /opt/qhb_backups
```

Команда удалит WAL-сегменты, созданные до последней успешной полной копии. Перед удалением выводится список кандидатов и запрашивается подтверждение.

Для предварительного просмотра без удаления:

```
qbackup remove-wal \
    -B /opt/qhb_backups \
    --dry-run
```

Для удаления без интерактивного подтверждения:

```
qbackup remove-wal -B /opt/qhb_backups -y
```

Рекомендации по стратегии хранения

Универсального рецепта нет — стратегия зависит от требований к RPO (допустимая потеря данных) и доступного дискового пространства. Общие рекомендации:

- Держите минимум одну полную копию, которой ещё не было на момент последней аварии.
- Для PITR необходима непрерывная цепочка WAL-сегментов. Не удаляйте WAL раньше, чем устареет самая старая нужная вам полная копия.
- Храните несколько поколений бэкапов на случай постепенного повреждения данных, которое не сразу обнаруживается.
- Автоматизируйте проверку (`qbackup validate`) и очистку (`qbackup remove --hold`) через планировщик задач.

Итоги

Вы научились:

- просматривать каталог копий в форматах таблицы, `--raw` и `--json`;
- удалять копии по ID командой `qbackup remove`;
- использовать `--dry-run` для предварительного просмотра, `-y` для автоматического подтверждения;
- удалять только один элемент цепочки с помощью `--single`;
- настраивать политику удержания через `--hold N`;
- очищать устаревшие WAL-сегменты командой `qbackup remove-wal`.

Последний рецепт покажет, как ускорить резервное копирование крупных баз и уменьшить размер копий с помощью параллельного выполнения и сжатия.