

Heroku PGBackups

English — 日本語に切り替える (/ja/articles/heroku-postgres-backups)

🕒 Last updated May 11, 2021

☰ Table of Contents

- Creating a backup
- Scheduling backups
- Downloading your backups
- Checking backup status
- Deleting backups
- Restoring backups
- Direct database-to-database copies
- Visibility
- Data Residency

Backups of your data are crucial to any application, not just for recovering in case of a catastrophe, but also for testing, setting up staging environments, and migrating your data. Every professional tier Heroku Postgres database comes with a behind-the-scenes Continuous Protection (<https://devcenter.heroku.com/articles/heroku-postgres-data-safety-and-continuous-protection>) mechanism for disaster recovery, as well as optional “logical” backups under your direct control.

This article explains how to take manual and scheduled backups, show your existing backups, restore backups, and transfer data directly between two databases.



For all production Heroku Postgres plans, using **continuous protection** (<https://devcenter.heroku.com/articles/heroku-postgres-data-safety-and-continuous-protection>) is the recommended path for data backup and recovery.



PGBackups are intended for moderately loaded databases up to 20 GB in size. Larger (or heavily loaded) databases, or database with large amounts of schemas or large objects may not be able to capture a logical backup before the backup process times out. Read more about **the performance impact of logical backups** (<https://devcenter.heroku.com/articles/heroku-postgres-logical-backups#the-performance-impact-of-logical-backups>).

Creating a backup

By default, `pg:backups` operates against your primary database, identified by the `DATABASE_URL` config var:

```
$ heroku pg:backups:capture --app sushi
Hit Ctrl-C at any time to stop watching progress; the backup will
continue running. Stop a running backup with heroku pg:backups:cancel.

HEROKU_POSTGRESQL_BLACK (DATABASE_URL) ----backup---> b251

Running... done
```

If you have multiple databases on your application, you can choose which one to backup by specifying the database name:

```
$ heroku pg:backups:capture HEROKU_POSTGRESQL_PINK
Hit Ctrl-C at any time to stop watching progress; the backup will
continue running. Stop a running backup with heroku pg:backups:cancel.

HEROKU_POSTGRESQL_PINK ----backup---> b252

Running... done
```

You can use the flag `--verbose` to see logs as your backup progresses. If you need to stop a backup for any reason, use the `cancel` command:

```
$ heroku pg:backups:cancel

Canceled backup b252
```

Manual backup retention limits

There is a limit to the number of manual backups that you can retain. That number is based on your database plan.

Plan	Backups Retained
Hobby-Dev	2
Hobby-Basic	5
Standard-*	25
Premium-*	50
Enterprise (https://devcenter.heroku.com/articles/heroku-postgres-and-private-spaces)	50

If you've reached this limit and need to take additional backups, the capture command will automatically expire the oldest manual backup before capturing a new one.



Capturing a backup adds some load on your database for the duration of the backup. How this impacts your application will vary with the size of your database and the nature of the app. Consider taking backups on a follower if there is a significant impact from running them on the master.

Scheduling backups

In addition to manually triggered backups, you can schedule regular automatic backups. These will run daily against the specified database.

```
$ heroku pg:backups:schedule DATABASE_URL --at '02:00 America/Los_Angeles' --app sushi
```

The `--at` option uses a 24 hour clock to indicate the hour of the day that you want the backup to be taken. The `--at` option also accepts a timezone in either the full TZ format (America/Los_Angeles) or the abbreviation (PST) but we recommend you use the full TZ format (http://en.wikipedia.org/wiki/List_of_tz_database_time_zones).



The `--at` option is required when scheduling a backup. For Windows please enclose the `--at` time and timezone in double quotes (") for example, "02:00 America/Los_Angeles" .



If you upgrade the database from hobby tier to production tier, the schedule will be lost.



When a backup is restored to a new database, the schedule from the original database does not get restored, a new schedule needs to be created.



Updating your Heroku Postgres plan with a **[follower changeover](https://devcenter.heroku.com/articles/updating-heroku-postgres-databases#updating-with-follower-changeover)** (<https://devcenter.heroku.com/articles/updating-heroku-postgres-databases#updating-with-follower-changeover>), the schedule from the original database remains associated to the original database, a new schedule needs to be created for the promoted database if one does not exist.

To stop regular backups, use `unschedule` :

```
$ heroku pg:backups:unschedule DATABASE_URL --app sushi
```


To view current schedules for your app, use:

```
$ heroku pg:backups:schedules --app sushi
Current backup schedules:
RED: daily at 2:00 (America/Los_Angeles)
```

Scheduled backups retention limits


Scheduled backups have a different retention policy as compared to the manual backups. This policy is also based on the database plan. For all plans, a daily backup is retained for the last 7 days. That means that 7 backups will exist, one for each of the last 7 days. A weekly backup means that only one backup is saved over a 7 day period. A monthly backup means that only 1 backup is saved over the course of a month. Based on current limits, for example, a Premium-0 would have 12 monthly backups, one for each of the last 12 months.

Plan	Weekly Backups Retained	Monthly Backup Retained

 We delete all backups for deprovisioned add-ons 30 days after the time of deprovisioning. If you wish to retain backups for a database that has been deprovisioned, we recommend **downloading your backup (<https://devcenter.heroku.com/articles/heroku-postgres-backups#downloading-your-backups>)** and storing in an external data storage service.

You can create a publicly accessible backup URL with the `pg:backups:url` command. When the command is specified without a backup id, the latest available backup URL will be returned. This is useful for exporting your data outside of Heroku Postgres:

```
$ heroku pg:backups:url b001 --app sushi
The following URL will expire at 2015-04-07 18:35:50 +0000:
"http://s3.amazonaws.com/xkpgbackups/app1234567@heroku.com/b004.dump?AWSAccessKeyId=ABCD"
```

 The URL is public but is not easily guessable. It expires after 60 minutes.

```
$ heroku pg:backups:url --app sushi | cat
http://s3.amazonaws.com/xkpgbackups/app1234567@heroku.com/b004.dump?AWSAccessKeyId=ABCD12
```

```
$ heroku pg:backups:download
```

❗ Please see our **documentation** (<https://devcenter.heroku.com/articles/heroku-postgres-import-export>) on importing and exporting Heroku Postgres databases with PGBackups for more information.

Checking backup status

To list your backups, you can run the following:

```
$ heroku pg:backups --app sushi
=== Backups
ID      Backup Time                Status                                Size  Database
----      -
b013    2015-03-18 19:03:16 +0000    Running                                IVORY
b011    2015-02-18 17:55:38 +0000    Finished 2015-02-18 17:55:39 +0000    1.9GB  IVORY
b010    2015-02-17 19:14:43 +0000    Finished 2015-02-17 19:14:48 +0000    1.9GB  IVORY
b004    2015-02-11 19:00:55 +0000    Finished 2015-02-17 19:14:48 +0000    1.9GB  IVORY

==== Restores
ID      Restore Time                Status                                Size  Database
----      -
r002    2015-03-16 17:33:19 +0000    Finished 2015-03-16 17:33:19 +0000    1.9GB  IVORY
r001    2015-03-15 12:13:44 +0000    Failed 2015-03-15 12:13:47 +0000    1.7GB  IVORY
```

To get more detailed information about a given backup, use the info command:

```
$ heroku pg:backups:info b017 --app sushi
=== Backup info: b017
Database: HEROKU_POSTGRESQL_IVORY
Started: 2013-06-24 17:11.28 UTC
Status:   Running
Type:     Manual
Size:     1.2GB
Schema:   0bff3ac
```

With the optional `--verbose` flag, you can also see the logs of the backup in question. If the backup is still running, the command will print the ongoing logs until the backup finishes or you cancel the command by typing `CTRL+C`.



PG Backups stores backups in a **compressed binary format** (<http://www.postgresql.org/docs/current/static/app-pgdump.html>), and the backups include commands to recreate indexes instead of storing the indexes themselves. As a result, backups will be much smaller in size than your database's current size on disk from `pg:info`.

Deleting backups

You can manually delete a backup to remove obsolete backups or make room for new captures. Use the backup ID to specify which backup to remove.

```
$ heroku pg:backups:delete b101 --app foo
Deleting b101... done
```

Restoring backups

To restore a backup, use the restore command:

```
$ heroku pg:backups:restore b101 DATABASE_URL --app sushi
```

This will restore backup id `b101` to the specified database URL in the app `sushi` . Note: you can omit the backup id and the target database to restore the latest backup to `DATABASE_URL` , otherwise both backup id and target database must be provided.

You can also restore from a backup on another app (from `sushi` app to `sushi-staging` app):

```
$ heroku pg:backups:restore sushi::b101 DATABASE_URL --app sushi-staging
```

Or from a publicly accessible URL:

```
$ heroku pg:backups:restore 'https://s3.amazonaws.com/me/items/mydb.dump' DATABASE_URL --app sushi
```



To ensure your data is encrypted while in transit, always use an HTTPS URL. Restoring over HTTP is unsupported.

Over time, a database will grow on disk due to table bloat (https://wiki.postgresql.org/wiki/Show_database_bloat) and unused index data. As a result, restoring a backup into a new database will almost always result in a reduction in overall disk size as reported by `pg:info` .



Unlike with the previous `pgbackups` commands, you cannot restore a partial backup into an existing database. When you run `pg:backups:restore` , **all data is deleted from the target database** before restoring the backup.

Direct database-to-database copies

In addition to backups and restores, `pg:backups` also provides the ability to transfer data directly between databases.

To perform such a transfer, run the following:

```
$ heroku pg:copy COBALT GREEN --app sushi
```

This would copy all data from the `COBALT` database to the `GREEN` database in the `sushi` app.

You can also transfer directly from a database on another app:

```
$ heroku pg:copy sushi::ORANGE GREEN --app sushi-staging
```

This would copy data from the `ORANGE` database of the `sushi` app to the `GREEN` database in `sushi-staging` . This could be used to copy production data into a staging app for testing purposes.

Visibility

Heroku Postgres needs to connect to your database using your credentials in order to capture a backup. This connection counts against your plan connection limit. You can identify connections made by Heroku Postgres by running the following command from your terminal:

```
$ heroku pg:psql -c "select * from pg_stat_activity where application_name = 'Heroku Postgres'"
```

Data Residency

All backups captured via `pg:backups` are stored in the U.S. (<https://devcenter.heroku.com/articles/heroku-postgresql#data-residency>) regardless of where your database is located.