Importing and Exporting Heroku Postgres Databases

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

Last updated May 03, 2021

Table of Contents

On the surface, PG Backups (https://devcenter.heroku.com/articles/heroku-postgres-backups) provides a way to capture regular backups of your Heroku Postgres (https://devcenter.heroku.com/articles/heroku-postgresql) database. However, because of its general-purpose architecture and use of standard PostgreSQL utilities, it is also a useful tool capable of exporting to or importing from external PostgreSQL databases.

An alternative to using the dump and restore method of import/export if you have a Postgres instance on your local machine is to use the **pg:push (https://devcenter.heroku.com/articles/heroku-postgresql#pg-push-and-pg-pull)** and **pg:pull (https://devcenter.heroku.com/articles/heroku-postgresql#pg-push-and-pg-pull)** CLI commands to automate the process.

Export

PG Backups uses the native `pg_dump` PostgreSQL tool to create its backup files, making it trivial to export to other PostgreSQL installations. Note that the resulting backup file uses the custom format option in `pg_dump`. As compared to the plain-text format, the custom format options results in backup files that can be much smaller.

In general, PGBackups are intended for moderately loaded databases up to 20 GB. Contention for the I/O, memory and CPU needed for backing up a larger database becomes prohibitive at a moderate load and the longer run time increases the chance of an error that will end your backup capture prematurely. For databases that are larger than 20 GB, see Capturing Logical Backups on Larger Databases (https://devcenter.heroku.com/articles/heroku-postgres-logical-backups#capturing-logical-backups-on-larger-databases).

Download backup

To export the data from your Heroku Postgres database, create a new backup and download it.

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

Restore to local database

Load the dump into your local database using the pg_restore (http://www.postgresql.org/docs/current/static/app-pgrestore.html) tool. If objects exist in a local copy of the database already, you might run into inconsistencies when doing a `pg_restore`.

This will usually generate some warnings, due to differences between your Heroku database and a local database, but they are generally safe to ignore.

```
$ pg_restore --verbose --clean --no-acl --no-owner -h localhost -U myuser -d mydb latest.
```

If you are using an old version of `pg_restore`, you may see an error such as `pg_restore: [archiver] unsupported version (1.13) in file header` when you try to run `pg_restore`. Please make sure that you are using `pg_restore` version 11, 10.3, 9.6.8, 9.5.12, 9.4.17, and 9.3.22 or up. You can check the version of your `pg_restore` by running `pg_restore --version`.

You can optionally use the `--jobs <number of jobs>` flag with `pg_restore` to parallelise the restore of the dump. Only the custom and directory archive formats are supported with this option. More on this can be found in the **Postgres documentation (https://www.postgresql.org/docs/current/app-pgrestore.html)**.

Import

PG Backups can be used as a convenient tool to import database dumps from other sources into your Heroku Postgres database.

If you are importing data as part of the initialization of a new application you will need to first create and configure the app on Heroku before performing the import.

Create dump file

Dump your local database in compressed format using the open source pg_dump (http://www.postgresql.org/docs/current/interactive/backup-dump.html) tool:

```
$ PGPASSWORD=mypassword pg_dump -Fc --no-acl --no-owner -h localhost -U myuser mydb > myd
```

Import to Heroku Postgres

In order for PG Backups to access and import your dump file you will need to upload it somewhere with an HTTP-accessible URL. We recommend using Amazon S3 (https://devcenter.heroku.com/articles/s3) with a signed url (https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-signed-urls.html).

Note that the `pg:backups restore` command drops any tables and other database objects before recreating them.

Generate a signed URL using the aws console:

```
$ aws s3 presign s3://your-bucket-address/your-object
```

Use the raw file URL in the `pg:backups restore` command:

```
$ heroku pg:backups:restore '<SIGNED URL>' DATABASE_URL
```

`DATABASE_URL` represents the `HEROKU_POSTGRESQL_COLOR_URL` of the database you wish to restore to. You must specify a database configuration variable to restore the database.

If you’re using a Unix-like operating system be sure to use single quotes around the temporary S3 URL, because it might contain ampersands and other characters that will confuse your shell. If you’re running Windows, you must use double-quotes.

When you have completed the import process, delete the dump file from its storage location if it’s no longer needed.