2
Fichiers

Comme tout serveur de bases de données, PostgreSQL gère un certain nombre de fichiers qu'il enregistre dans différents répertoires suivant leur type et leur utilité. Quelques-uns sont journalisés, c'est-à-dire que les modifications dont ils font l'objet sont d'abord enregistrées dans un fichier intermédiaire avant d'être réellement effectuées sur le fichier cible.

Le répertoire principal est appelé le *répertoire des données* (*data directory* en anglais). Ce dernier contient quelques fichiers (comme les fichiers de configuration) ainsi que des répertoires.

Des répertoires situés ailleurs que dans le répertoire principal peuvent être utilisés :

- les tablespaces, qui sont des répertoires définis par l'administrateur, contenant des fichiers de données ;
- les sous-répertoires du répertoire principal, déplacés ailleurs mais accessibles via des liens symboliques (cette méthode est surtout utilisée pour déplacer le répertoire des journaux de transactions dans leur propre système disque).

Note *› L'astuce des liens symboliques est intéressante dans certains cas, comme celui des journaux de transactions parce qu'il n'existe aucun autre moyen de les stocker ailleurs. Par contre, elle est fortement déconseillée dans le cas des fichiers de données. Pour ces derniers, il est préférable de passer par les tablespaces car ils sont entièrement gérés par PostgreSQL, et donc bien plus fiables et pratiques.*

Certains outils (comme `initdb` et `pg_controldata`) livrés avec PostgreSQL peuvent utiliser une variable d'environnement appelée `PGDATA`. Elle pointe vers l'emplacement du répertoire de données principal.

2.1. Répertoire de données principal

Le répertoire de données principal est créé à l'origine par un outil nommé `initdb`. Cet outil s'appuie sur un système appelé BKI (pour *BacKend Interface*) pour savoir comment créer les différents répertoires et fichiers nécessaires au démarrage de PostgreSQL.

La commande `initdb` nécessite de connaître l'emplacement du répertoire des données. Il peut ne pas exister mais, dans ce cas, il doit pouvoir être créé par l'utilisateur qui

exécute la commande. S'il existe déjà, il doit être vide et l'utilisateur qui exécute la commande doit avoir les droits pour y ajouter des fichiers et répertoires. Il doit aussi pouvoir modifier les droits d'accès à ce répertoire, PostgreSQL nécessitant un accès exclusif à ce répertoire (droits 700).

Il est possible d'autoriser l'accès aux utilisateurs membres du groupe `postgres`. Cela se fait lors de l'exécution d'`initdb` avec l'option `--allow-group-access`. Dans ce cas, les droits sur ce répertoire ainsi que tous les sous-répertoires sont 0750, et 0740 pour tous les fichiers. Il est également possible d'autoriser ou de restreindre l'accès une fois l'instance initialisée. Cela doit se faire l'instance arrêtée, et les droits nécessaires doivent être positionnés sur chacun des répertoires et fichiers.

Voici le retour de cette commande si l'utilisateur a respecté les prérequis :

```
$ initdb --data /opt/postgresql/datas/livre --data-checksums ❶
The files belonging to this database system will be owned by user
 "postgres".
This user must also own the server process.

The database cluster will be initialized with locale "en_US.UTF-8".
The default database encoding has accordingly been set to "UTF8".
The default text search configuration will be set to "english".

Data page checksums are enabled.

creating directory /opt/postgresql/datas/livre ... ok  ❷
creating subdirectories ... ok
selecting dynamic shared memory implementation ... posix
selecting default max_connections ... 100
selecting default shared_buffers ... 128MB
selecting default time zone ... Europe/Paris
creating configuration files ... ok
running bootstrap script ... ok
performing post-bootstrap initialization ... ok
syncing data to disk ... ok

initdb: warning: enabling "trust" authentication for local connections
initdb: hint: You can change this by editing pg_hba.conf or using the option
 -A, or --auth-local and --auth-host, the next time you run initdb.

Success. You can now start the database server using:  ❸

    pg_ctl -D /opt/postgresql/datas/livre -l logfile start
```

Une fois la commande exécutée, le répertoire des données contient un bon nombre de fichiers et de répertoires :

```
$ ls -l /opt/postgresql/datas/livre
total 120
drwx------. 5 postgres postgres  4096 Sep 18 11:35 base
drwx------. 2 postgres postgres  4096 Sep 18 11:35 global
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_commit_ts
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_dynshmem
-rw-------. 1 postgres postgres  5711 Sep 18 11:35 pg_hba.conf
-rw-------. 1 postgres postgres  2640 Sep 18 11:35 pg_ident.conf
drwx------. 4 postgres postgres  4096 Sep 18 11:35 pg_logical
drwx------. 4 postgres postgres  4096 Sep 18 11:35 pg_multixact
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_notify
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_replslot
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_serial
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_snapshots
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_stat
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_stat_tmp
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_subtrans
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_tblspc
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_twophase
-rw-------. 1 postgres postgres     3 Sep 18 11:35 PG_VERSION
drwx------. 3 postgres postgres  4096 Sep 18 11:35 pg_wal
drwx------. 2 postgres postgres  4096 Sep 18 11:35 pg_xact
-rw-------. 1 postgres postgres    88 Sep 18 11:35 postgresql.auto.conf
-rw-------. 1 postgres postgres 29706 Sep 18 11:35 postgresql.conf
```

❶ La commande `initdb` commence par donner quelques informations : le répertoire principal des données, le propriétaire des fichiers (qui devra donc être l'utilisateur qui lance le serveur), la locale du serveur, la configuration de la recherche plein texte, ainsi que l'activation ou non des sommes de contrôle sur les fichiers de données.

❷ Ensuite, elle entame son travail. Elle crée le répertoire des données, s'assure des droits sur ce répertoire et crée les sous- répertoires. Elle lance le serveur plusieurs fois pour déterminer au mieux certains paramètres (comme `max_connections` et `shared_buffers`). Elle commence par les valeurs hautes et descend ces valeurs tant que le serveur ne peut pas démarrer. Une fois arrivée aux valeurs avec lesquelles le serveur peut fonctionner, elle arrête et conserve ces valeurs, qu'elle stocke dans le fichier de configuration. Elle crée la première base de données (nommée `template1`, d'identifiant 1). Elle lui ajoute tous les objets systèmes nécessaires à son bon fonctionnement. Elle exécute un `VACUUM` avec l'option `FREEZE` sur cette base. Enfin, elle copie la base `template1` pour créer les bases `template0` et `postgres`. Elle termine son travail en synchronisant les données sur disque.

❸ Elle finit en indiquant le succès de l'opération et en fournissant les lignes de commandes permettant de démarrer le serveur.

En cas d'erreur, la commande `initdb` nettoie le répertoire. Il peut devenir difficile de comprendre pourquoi l'erreur s'est produite et comment la corriger. Ce comportement est désactivable avec l'option `--no-clean`.

Tous ces répertoires et fichiers ont un rôle particulier et connaître leur fonctionnement permet de mieux configurer, administrer et superviser PostgreSQL. La Figure 2.1 récapitule les plus importants en partant du répertoire principal des données.

Figure 2.1 : *Liens entre répertoires et fichiers*