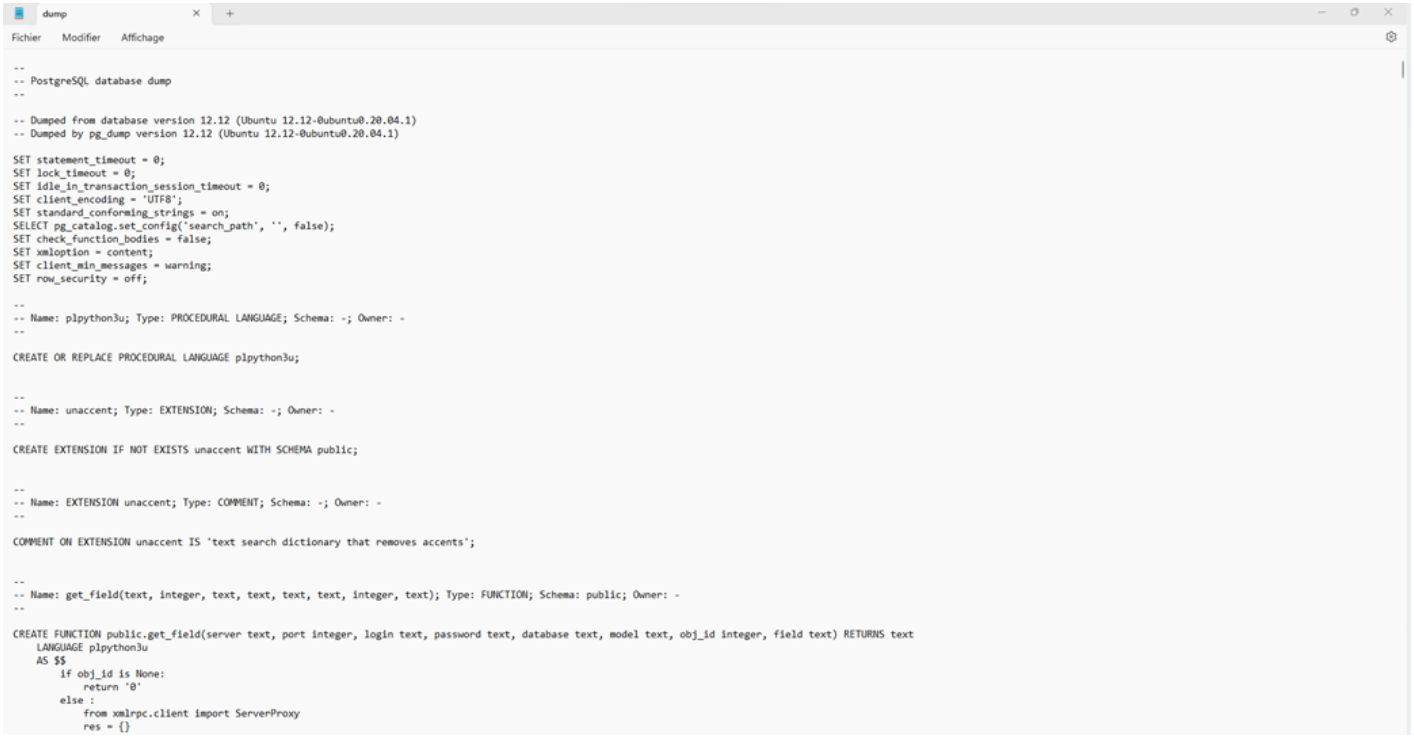
Manipulation des bases de données

- Présentation
- Restauration d'une base de test
- Impact d'une montée de version (sql-update)
- Déterminer la volumétrie d’une base de données
- Echec de restauration de Base de données

Présentation

Sous PostgreSQL, la sauvegarde et la restauration de bases s'effectuent par deux commandes distinctes : `pg_dump` et `pg_restore`. Chacune de ces deux commandes exploitent des fichiers « dump », l'équivalent sous SQL Server du « .bak ».

Si on regarde un peu plus en détail le contenu d'un fichier .dump, on constate que le .dump est une succession de commande SQL, que le moteur va « rejouer », récréant ainsi la structure de la base de données (ses extensions, ses fonctions, ses tables, etc..) et les enregistrements contenus dans chacun des fichiers au moment où la sauvegarde a été produite.

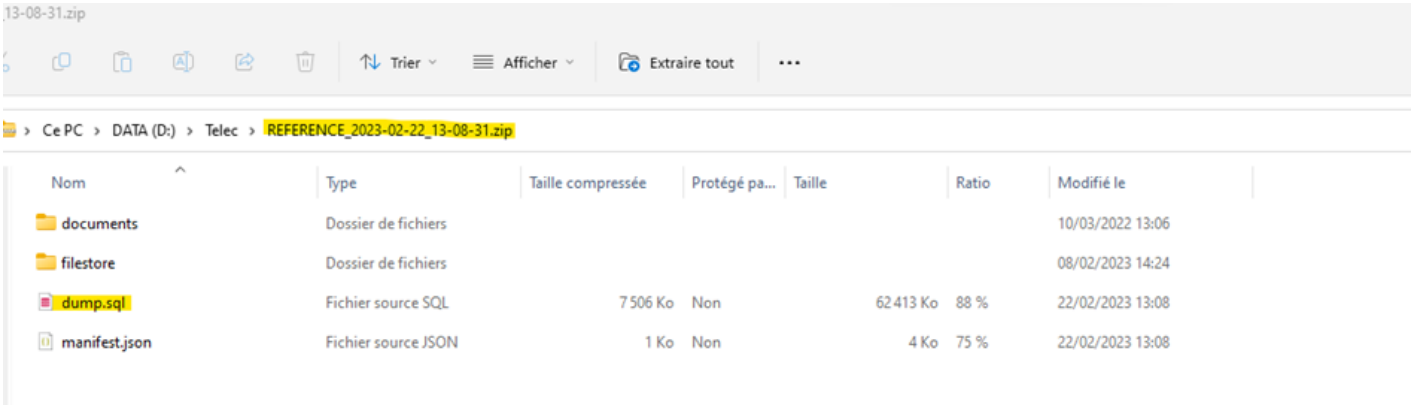


Sur Open-Prod, il existe deux type des sauvegardes : celle en .dump, et celle avec le filestore inclus.
Leur contenu est le suivant :

1. Si on produit une sauvegarde avec pg_dump, on obtiendra un fichier directement exploitable dans une version compatible pour tout moteur PostgreSQL de la même version :



2. Si on produit une sauvegarde avec le filestore inclus, le fichier sera produit au format zip et contiendra les fichiers suivants :



Le zip sera porteur du nom de la base sur le serveur d'origine et le .dump sera intégré dans ce dernier.

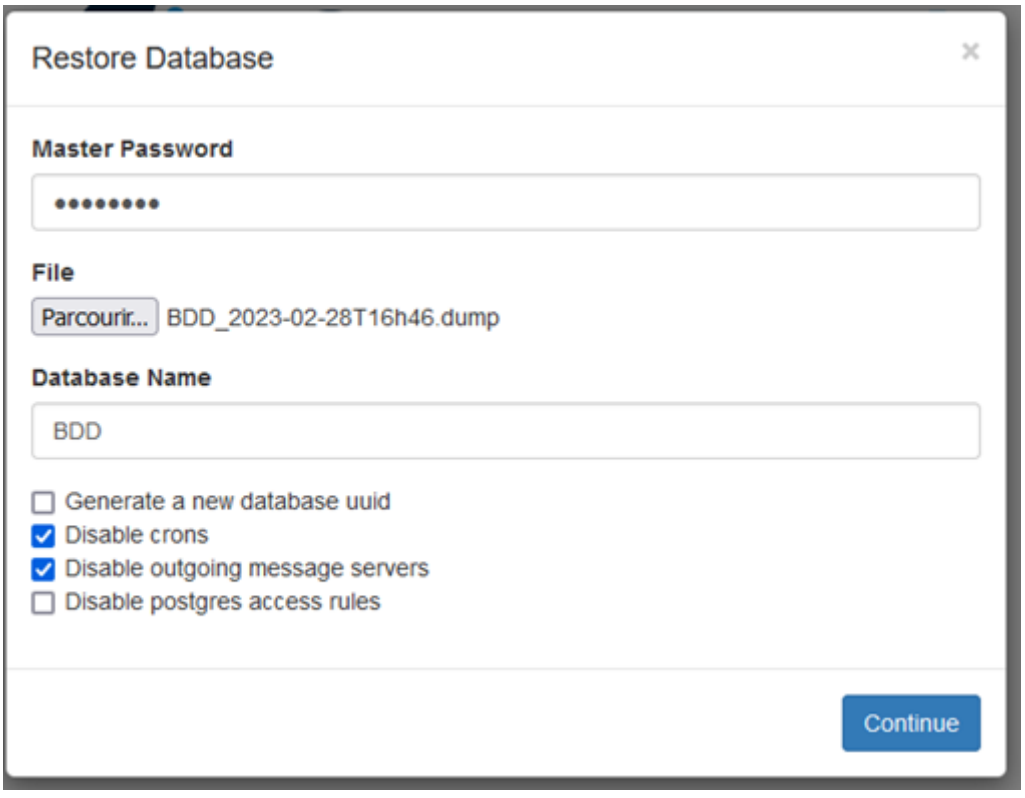
> **Attention ! la volumétrie du .dump n'a rien a voir avec l'espace disque qu'utilisera la base de données une fois restaurée. En effet, le .dump est un fichier « txt » qui ne contient que les commandes permettant de réalimenter la base. Avec des champs « vides » par exemple occuperont nécessairement un espace dans un SGBD mais pas dans le dump.**

Restauration d'une base de test

La première étape est de vérifier l'espace disque sur le serveur :

```
openprod@ubuntu18-jmp:~ $ df -H
Filesystem                         Size  Used Avail Use% Mounted on
udev                               4,1G     0  4,1G   0% /dev
tmpfs                              817M  938k  816M   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv   67G   13G   51G  21% /
tmpfs                              4,1G   17k  4,1G   1% /dev/shm
tmpfs                              5,3M     0  5,3M   0% /run/lock
tmpfs                              4,1G     0  4,1G   0% /sys/fs/cgroup
/dev/sda2                          1,1G  160M  791M  17% /boot
tmpfs                              817M     0  817M   0% /run/user/1000
openprod@ubuntu18-jmp:~ $
```

Nous constatons ici que le disque dur a une taille de 67 Go utiles et que nous avons une disponibilité de 51 Go.
Nous allons donc restaurer notre fichier dump cité plus haut sur ce disque dur au travers de l'interface d'Open-Prod.



Avec la commande `tail -f /var/log/openprod/openprod-server.log`, il est possible de "suivre" les traitements réalisés par Open-Prod sur le moment. Nous trouvons l'information suivante :

Le serveur exécute la command `pg_restore` sur le fichier « /tmp/tmp5SX1l3 », préalablement téléchargé via le navigateur. Une fois restauré, on contrôle à nouveau l'espace disque :

```
openprod@ubuntu18-jmp:~ $ df -H
Filesystem                      Size  Used Avail Use% Mounted on
udev                            4,1G     0  4,1G   0% /dev
tmpfs                           817M  2,0M  815M   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv  67G   25G   39G  40% /
tmpfs                           4,1G   17k  4,1G   1% /dev/shm
tmpfs                           5,3M     0  5,3M   0% /run/lock
tmpfs                           4,1G     0  4,1G   0% /sys/fs/cgroup
/dev/sda2                       1,1G  160M  791M  17% /boot
tmpfs                           817M     0  817M   0% /run/user/1000
openprod@ubuntu18-jmp:~ $
```

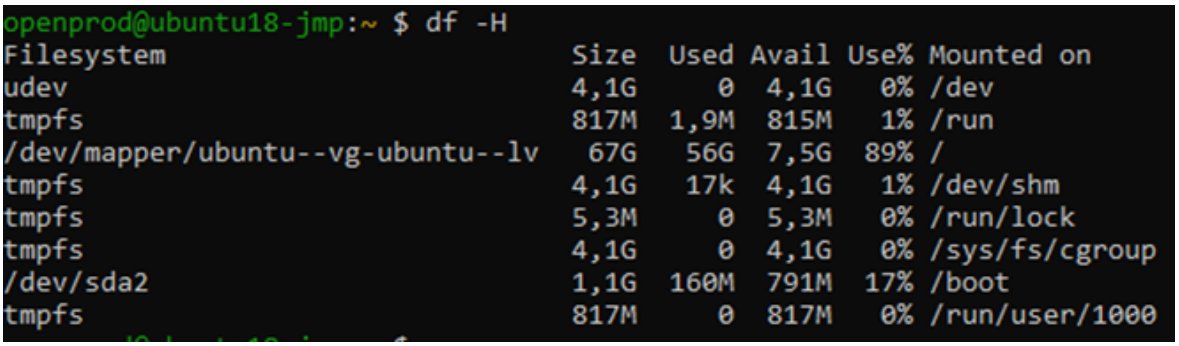
> **Nous constatons que le fichier dump de 1.13 Go, une fois restauré occupe +/- 12 Go d'espace disque.**

Impact d'une montée de version (sql-update)

De nouveau, une montée de version d’Open-Prod peut fortement impacter l'espace disque de l'environnement. Par exemple, si de nouveaux champs fonctionnels sont mis en place par l’éditeur (ou vos modules) dans un grand nombre d’enregistrements, l’occupation définitive du disque peut arriver à saturation.

Naturellement, PostgreSQL va créer des fichiers d’échange temporaire sur disque et il convient que de l’espace disque soit disponible en quantité suffisante lors de l’opération.

Si nous poursuivons sur l'exemple précédent, voici l'espace disque suite à un `sql-update` sur notre base de donnée :

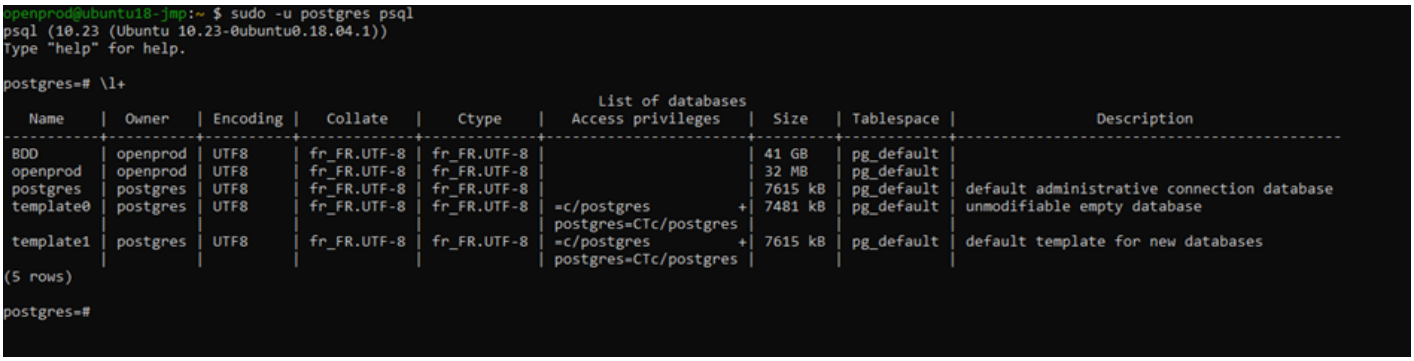


> **Après la montée de version de notre base dans un format ancien, notre espace disque disponible est tombé à 7.5 Go, la taille de la base a donc augmenté de 32 Go ! Prenez soin d’avoir toujours le maximum d’espace disque disponible lors d’une montée de version et de tester la taille définitive de la future base sur un serveur de test.**

Déterminer la volumétrie d’une base de données

Il existe de nombreux moyen pour déterminer la taille d’une base de données spécifique.

Via la commande `sudo -u postgres psql` puis `\l+` :

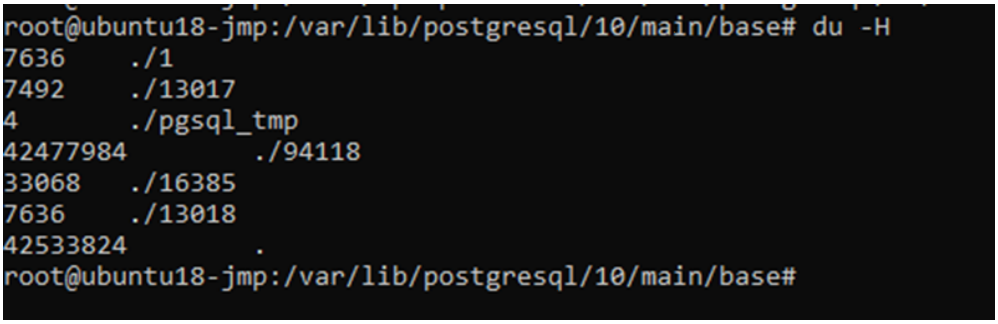


```
openprod@ubuntu18-jmp:~ $ sudo -u postgres psql
psql (10.23 (Ubuntu 10.23-0ubuntu0.18.04.1))
Type "help" for help.

postgres=# \l+
                                                                    List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges   |  Size   | Tablespace |                Description
-----------+----------+----------+-------------+-------------+-----------------------+---------+------------+--------------------------------------------
 BDD       | openprod | UTF8     | fr_FR.UTF-8 | fr_FR.UTF-8 |                       | 41 GB   | pg_default |
 openprod  | openprod | UTF8     | fr_FR.UTF-8 | fr_FR.UTF-8 |                       | 32 MB   | pg_default |
 postgres  | postgres | UTF8     | fr_FR.UTF-8 | fr_FR.UTF-8 |                       | 7615 kB | pg_default | default administrative connection database
 template0 | postgres | UTF8     | fr_FR.UTF-8 | fr_FR.UTF-8 | =c/postgres          +| 7481 kB | pg_default | unmodifiable empty database
           |          |          |             |             | postgres=CTc/postgres |         |            |
 template1 | postgres | UTF8     | fr_FR.UTF-8 | fr_FR.UTF-8 | =c/postgres          +| 7615 kB | pg_default | default template for new databases
           |          |          |             |             | postgres=CTc/postgres |         |            |
(5 rows)

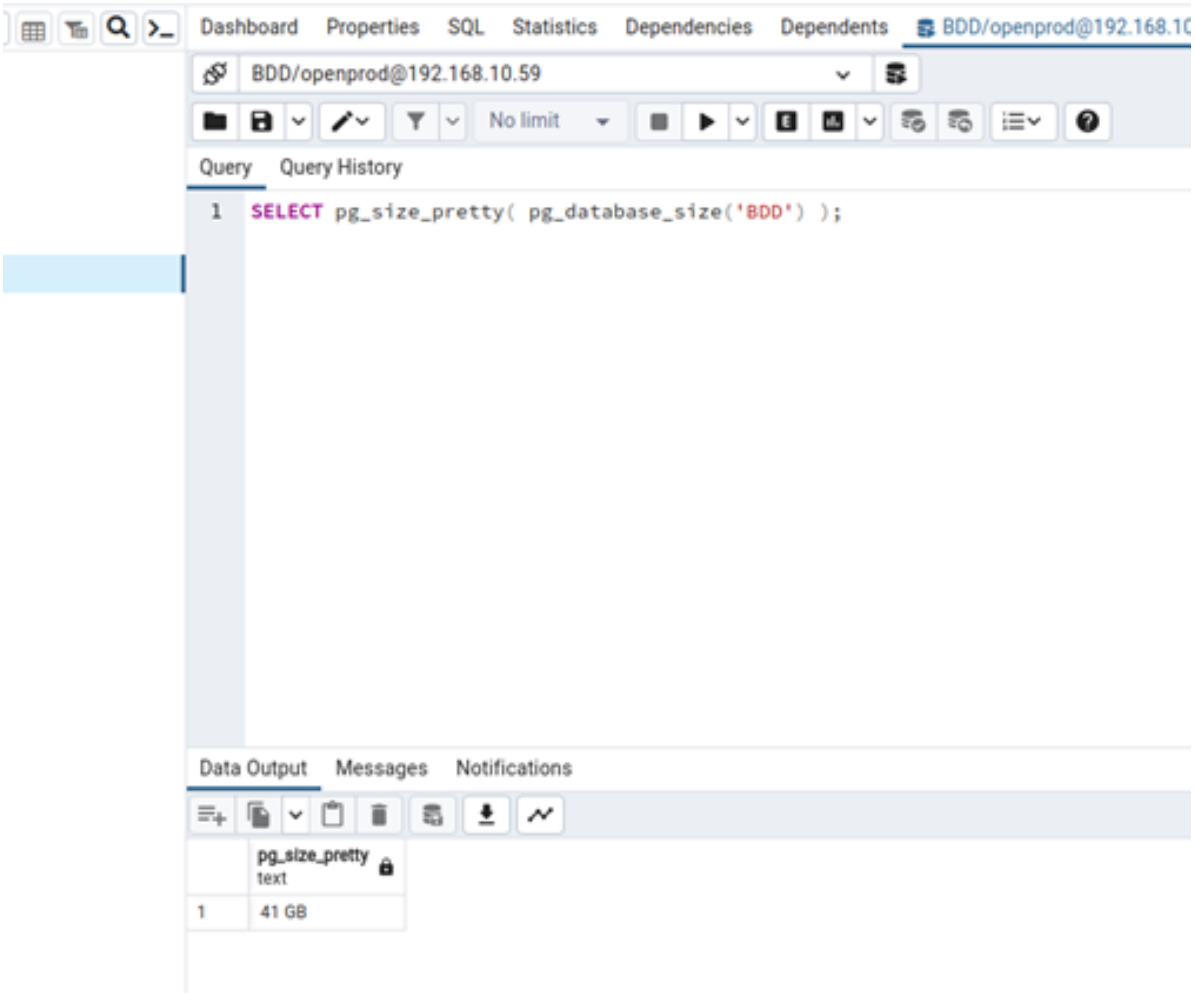
postgres=#
```

Via la commande `du` si on connait le lieu de stockage de nos bases. Ici la base « BDD » correspond au répertoire 94118, soit 42477984 Mo :



```
root@ubuntu18-jmp:/var/lib/postgresql/10/main/base# du -H
7636    ./1
7492    ./13017
4       ./pgsql_tmp
42477984        ./94118
33068   ./16385
7636    ./13018
42533824        .
root@ubuntu18-jmp:/var/lib/postgresql/10/main/base#
```

Depuis l'interface de pgadmin4 et v **SELECT pg_size_pretty(pg_database_size('BDD'))**


Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
BDD/openprod@192.168.10.59
No limit
Query
Query History
1 SELECT pg_size_pretty(pg_database_size('BDD'));
Data Output
Messages
Notifications
pg_size_pretty
text
1
41 GB

Echec de restauration de Base de données

Saturer l’espace disque d’un serveur de production va fatalement interrompre/impacter un grand nombre de services. Veiller toujours à avoir de l’espace en adéquation avec la manipulation à réaliser. Par exemple, si on veut dupliquer une base par la production d’un fichier zip, le serveur va occuper l’espace de la base de données initiale plus la base de données compressée (dans le .zip uploadé) et la taille de la base de données de destination.

Voici un exemple d'erreur qui peut être rencontrée :

> Database restore error: Postgres subprocess ('/usr/bin/pg_restore', u'--dbname=BDD_TEST', '--no-privileges', '--role=openprod', '--no-owner', '/tmp/tmpfz2jX9') error 1

Et le log joint :



> **Si vous obtenez ce type de message lors d’une restauration, contrôler rapidement l’espace à disposition sur le serveur et effectuer le correctif nécessaire.**