Export et Import d'une base de données sous Postgre...

Par Guillaume DI FRANCESCO • Publié le 21/01/2017 à 10:57:35 • Noter cet article: ★★★★★ (1 votes)

Avis favorable du comité de lecture

Guillaume DI FRANCESCO

Depuis quelques années, derrière les systèmes de bases de données SQL et Oracle, un troisième système - et non des moindres - est a dénoter : PostgreSQL. Réputé plus performant pour les gros volumes de données, celui-ci est assez commun en entreprise et des connaissances sur cette technologie sont donc de plus en plus nécessaires.
En ce sens, par cet article, nous allons montrer comment utiliser les commandes fournies par le logiciel PostgreSQL pour, d'une part, exporter une base de données d'un serveur 1 et, d'autre part, l'exporter sur un serveur 2.
Pour cela, nous utiliserons deux serveurs distincts sous Debian 8 :

- PGSQL01, qui sera notre serveur historique, hébergeant nos bases de données.
- PGSQL02, qui sera notre nouveau serveur, sur lequel nous allons importer notre base de données.

Export de la base de données.

Sur PGSQL01 :
Tout d'abord, regardons les bases de données présentes sur notre serveur.

```
###Connexion au compte d'administration postgre.
root@pgsql01:~# su - postgres

###Connexion au service de base de données.
postgres@pgsql01:~$ psql
psql (9.4.10)
Saisissez « help » pour l'aide.

###On liste les bases de données.
postgres=# \list
                                  Liste des bases de données
    Nom    | Propriétaire | Encodage | Collationnement | Type caract. |    Droits d'accès
-----------+--------------+----------+-----------------+--------------+-----------------------
 postgres  | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  |
 template0 | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  | =c/postgres          +
           |              |          |                 |              | postgres=CTc/postgres
 template1 | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  | =c/postgres          +
           |              |          |                 |              | postgres=CTc/postgres
 test      | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  |
(4 lignes)

postgres=#
```

Ici, nous allons exporter la base de données test. Pour cela, nous allons utiliser la commande pg_dump, fournie directement par le logiciel PostgreSQL.

```
####Connexion au compte d'administration postgre.
root@pgsql01:~# su - postgres

###Export de notre base de données.
postgres@pgsql01:~$  pg_dump test > /var/tmp/test_dump.bak
```

Par ces lignes de commandes, nous avons sauvegardé notre base de données "test" dans le fichier "test_dump.bak" situé dans le dossier /var/tmp.
Nous pouvons vérifier le contenu de notre fichier en utilisant la commande "cat".

```
root@pgsql01:/var/tmp# cat test_dump.bak
--
-- PostgreSQL database dump
--
SET statement_timeout = 0;
SET lock_timeout = 0;
```

```
SET client_encoding = 'UTF8';
SET standard_conforming_strings = on;
SET check_function_bodies = false;
SET client_min_messages = warning;
--
-- Name: plpgsql; Type: EXTENSION; Schema: -; Owner:
--
CREATE EXTENSION IF NOT EXISTS plpgsql WITH SCHEMA pg_catalog;
--
-- Name: EXTENSION plpgsql; Type: COMMENT; Schema: -; Owner:
--
COMMENT ON EXTENSION plpgsql IS 'PL/pgSQL procedural language';
SET search_path = public, pg_catalog;
SET default_tablespace = '';
SET default_with_oids = false;
--
-- Name: test; Type: TABLE; Schema: public; Owner: postgres; Tablespace:
--
CREATE TABLE test (
    id integer NOT NULL,
    test character varying(100) NOT NULL
);
ALTER TABLE test OWNER TO postgres;
--
-- Name: test_id_seq; Type: SEQUENCE; Schema: public; Owner: postgres
--
CREATE SEQUENCE test_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;
ALTER TABLE test_id_seq OWNER TO postgres;
--
-- Name: test_id_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres
--
ALTER SEQUENCE test_id_seq OWNED BY test.id;
--
-- Name: id; Type: DEFAULT; Schema: public; Owner: postgres
--
ALTER TABLE ONLY test ALTER COLUMN id SET DEFAULT nextval('test_id_seq'::regclass);
--
-- Data for Name: test; Type: TABLE DATA; Schema: public; Owner: postgres
--
COPY test (id, test) FROM stdin;
1       Hello
\.
--
-- Name: test_id_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres
--
SELECT pg_catalog.setval('test_id_seq', 1, true);
--
-- Name: test_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres; Tablespace:
--
ALTER TABLE ONLY test
    ADD CONSTRAINT test_pkey PRIMARY KEY (id);
--
-- Name: public; Type: ACL; Schema: -; Owner: postgres
--
REVOKE ALL ON SCHEMA public FROM PUBLIC;
REVOKE ALL ON SCHEMA public FROM postgres;
GRANT ALL ON SCHEMA public TO postgres;
GRANT ALL ON SCHEMA public TO PUBLIC;
--
-- PostgreSQL database dump complete
--
root@pgsql01:/var/tmp#
```

Notre fichier étant bien créé, nous allons maintenant envoyer celui-ci sur notre second serveur. Pour cela, nous utilisons la commande SCP (Secure CoPy).

```
###Envoi de l'export de la base test.
postgres@pgsql01:~$ scp /var/tmp/test_dump.bak root@192.168.1.65:/var/tmp/test_dump.bak
The authenticity of host '192.168.1.65 (192.168.1.65)' can't be established.
ECDSA key fingerprint is 2e:98:42:5f:b3:61:44:b4:14:88:f3:a6:9d:7f:81:ce.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.65' (ECDSA) to the list of known hosts.
root@192.168.1.65's password:
test_dump.bak                                 100% 2698KB   2.6MB/s   00:00
postgres@pgsql01:~$
```

Vérifions que nos fichiers sont bien copiés sur notre second serveur.

```
root@pgsql02:~# ls -rtl /var/tmp
total 2700
-rw-r--r-- 1 root root 2762791 janv. 20 12:35 test_dump.bak
root@pgsql02:~#
```

Passons désormais à l'import de nos bases de données sur notre second serveur, PGSQL02.

Import de la base de données.

Sur PGSQL02 :
Regardons sur notre second serveur la configuration de nos bases de données. Pour cela, nous allons nous connecter à notre service de gestion de base de données puis nous listerons les bases. :

```
###Connexion au compte d'administration de Postgre.
root@pgsql02:~# su - postgres

###Connexion au service de base de données.
postgres@pgsql02:~$ psql
psql (9.4.10)
Saisissez « help » pour l'aide.

###Listing des bases de données.
postgres=# \list
                                  Liste des bases de données
    Nom    | Propriétaire | Encodage | Collationnement | Type caract. |    Droits d'accès
-----------+--------------+----------+-----------------+--------------+-----------------------
 postgres  | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  |
 template0 | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  | =c/postgres          +
           |              |          |                 |              | postgres=CTc/postgres
 template1 | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  | =c/postgres          +
           |              |          |                 |              | postgres=CTc/postgres
(3 lignes)

postgres=#
```

Désormais, avant d'importer notre base et ses données, nous allons créer une nouvelle base sur notre serveur, ayant le même nom.

```
###Connexion au compte d'administration.
root@pgsql02:~# su - postgres

###Connexion au service de base de données.
postgres@pgsql02:~$ psql
psql (9.4.10)
Saisissez « help » pour l'aide.

###Creation de la base de données.
postgres=# CREATE DATABASE test;
CREATE DATABASE

###Listing des bases de données.
postgres=# \list
                                  Liste des bases de données
    Nom    | Propriétaire | Encodage | Collationnement | Type caract. |    Droits d'accès
-----------+--------------+----------+-----------------+--------------+-----------------------
 postgres  | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  |
 template0 | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  | =c/postgres          +
           |              |          |                 |              | postgres=CTc/postgres
 template1 | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  | =c/postgres          +
           |              |          |                 |              | postgres=CTc/postgres
 test      | postgres     | UTF8     | fr_FR.UTF-8     | fr_FR.UTF-8  |
(4 lignes)

postgres=#
```

Nous pouvons désormais passer à l'import de notre base. Pour cela, nous utilisons directement la commande psql.

```
###Connexion au compte d'administration.
root@pgsql02:~# su - postgres

###Import de notre fichier dans la base de données test.
postgres@pgsql02:~$ psql -U postgres -d test -f /var/tmp/test_dump.bak
SET
SET
SET
SET
SET
SET
CREATE EXTENSION
COMMENT
SET
SET
SET
CREATE TABLE
ALTER TABLE
CREATE SEQUENCE
ALTER TABLE
ALTER SEQUENCE
ALTER TABLE
COPY 1
 setval
--------
      1
(1 ligne)

ALTER TABLE
REVOKE
REVOKE
GRANT
GRANT
postgres@pgsql02:~$
```

Sur cette dernière ligne de commande, nous pouvons noter :

- -U : Le compte utilisateur à utiliser pour l'import de la base de données.
- -d : La base de données dans laquelle effectuer l'import.
- -f : Le fichier a utiliser pour l'import de notre base de données ; dans notre cas /var/tmp/test_dump.bak.

Une fois ceci fait, vérifions que nos données ont été importées correctement.

```
###Connexion à la base de données "test".
postgres@pgsql02:~$ psql test

psql (9.4.10)
Saisissez « help » pour l'aide.

###Vérification des données présentes dans notre base de données.
test=# SELECT * FROM test;
 id | test
----+-------
  1 | Hello
(1 ligne)

test=#
```

Nos données sont bien présentes ; l'import a donc réussi et notre base de données est maintenant opérationnelle.