PostgreSQL
Administration
Travaux pratiques

PGRE-SQL - TP - Rév 9

m2iformation.fr

SOMMAIRE

1. EXERCICES

1. Installation via les sources (TP du chapitre 2)

1/ Copier le fichier source (…tar.gz) récupéré sur Internet dans le répertoire **/usr/local/src**.

2/ Se positionner sous **/usr/local/src** et extraire les fichiers sources de l'archive.

3/ Configurer les sources sans option particulière (**./configure**).

4/ Compiler les sources (**gmake**).

5/ Installer les fichiers (**gmake install**).

6/ Créer l'utilisateur **postgres**.

7/ Configurer les variables d'environnement du fichier **.bash_profile** de l'utilisateur **postgres**.

8/ Effectuer les tests de régression (**gmake check**) avec l'utilisateur **postgres**.

Si vous avez du temps :

9/ Parcourir l'arborescence source et visualiser les fichiers **INSTALL**, **README** et **HISTORY**. Parcourir l'arborescence d'installation des fichiers (par défaut /usr/local/pgsql).

10/ Visiter le site Internet **www.postgresql.org.**

2. Mise en œuvre d'un serveur de données PostgreSQL (TP du chapitre 3)

ENVIRONNEMENT SERVEUR

1/ Initialiser un serveur de bases de données (Database Cluster) dans /home/postgres/data. Visualiser l'arborescence créée.

2/ Positionner, vérifier votre variable d'environnement PGDATA. Démarrer votre serveur avec commande pg_ctl en utilisant un fichier de log. Vérifier l 'état de votre serveur (pg_ctl status). Visualiser le contenu du fichier de log du serveur.

3/ Visualiser votre fichier postgresql.conf. Modifier les paramètres : 300 MB de shared buffers, 10 MB de work_mem, un checkpoint toutes les 7 minutes et accepter les connexions distantes. Arrêter et redémarrer votre serveur. Vérifier les valeurs des paramètres dans la vue pg_settings.

4/ Visualiser la valeur courante du paramètre work_mem (show work_mem;). Avec la commande ALTER SYSTEM, positionner la valeur de work_mem à 15MB. Visualiser de nouveau la valeur courante du paramètre work_mem. Effectuer un reload du serveur et visualiser de nouveau la valeur courante du paramètre work_mem. Afficher le contenu (sans le modifier) du fichier postgresql.auto.conf.

5/ Visualiser les process background de votre serveur PostgreSQL avec la commande ps.

Si vous avez du temps :

6/ Utilisez la commande service sous root pour démarrer / arrêter votre serveur Postgresql.

3. Création d'une base de données (TP du chapitre 4)

1/ Vous connecter à la base template1 avec psql et visualiser les bases existantes.

2/ Créer une ou deux bases dans /home/postgres/data.

3/ Créer un tablespace tbsp1dans le répertoire /home/postgres/tbsp1.

4/ Créer une base de données dans ce nouveau tablespace. Visualiser les informations des tables pg_database, pg_tablespace et utiliser les commandes spécifiques \l et \db de psql. Visualiser le contenu du répertoire $PGDATA/pg_tblspc et du répertoire correspondant au nouveau tablespace (/home/postgres/tbsp1).

5/ Supprimer une base de données.

Si vous avez du temps :

6/ Connectez-vous à une de vos bases et vérifiez la valeur de son paramètre « enable_mergejoin » avec la commande show de psql.

7/ Avec la commande « alter database », positionner le enable_mergejoin de cette base à off et vérifier de nouveau son « enable_mergejoin » avec la commande show de psql.

8/ Déconnectez-vous de cette base et reconnectez-vous à cette base avec psql. Vérifiez de nouveau son « enable_mergejoin » avec la commande show de psql.

4. Authentification des clients (TP du chapitre 5)

1/ Créer un utilisateur possédant un mot de passe (ex : create role jean login password 'jean').

2/ Ajouter un mot de passe au super-utilisateur postgres du serveur PostgreSQL (alter role postgres password 'mot_de_passe').

3/ Configurer votre fichier pg_hba.conf pour :

- autoriser les connexions locales traversant les couches TCP/IP (c'est-à-dire en utilisant l'adresse 127.0.0.1) sur une des bases de données que vous avez créée dans le module précédent en utilisant l'utilisateur créé en 1 et en demandant un mot de passe crypté au format md5.

- autoriser les connexions locales traversant les couches TCP/IP (c'est-à-dire en utilisant l'adresse 127.0.0.1) sur toutes les bases de données de votre serveur PostgreSQL en utilisant l'utilisateur postgres et en demandant un mot de passe crypté au format md5.

- refuser les connexions locales ne traversant pas les couches TCP/IP (c'est-à-dire l'entrée de type local du fichier pg_hba.conf).

4/ Tester votre fichier pg_hba.conf en utilisant psql pour vous connecter.

5/ Revalider les connexions locales ne traversant pas les couches TCP/IP (remettre l'entrée « local all all trust ») pour faciliter les travaux pratiques des modules suivants.

5. Gestion de la sécurité (TP du chapitre 6)

1/ Créer l'utilisateur **admpaye** qui sera propriétaire des objets d'une application gérant la paye.

2/ Créer la base de données **prod** (dans laquelle on stockera les objets de l'application gérant la paye).

3/ Connectez-vous à la nouvelle base de données **prod** sous l'identité de l'utilisateur **admpaye**. Créer les tables de l'application paye en exécutant le script **demotab.txt**.

4/ Créer un groupe **PAYE**.

5/ Créer deux ou trois utilisateurs rattachés à ce groupe.

6/ Donner les privilèges SELECT, INSERT, UPDATE sur la table EMP appartenant à l'utilisateur **admpaye** au groupe **PAYE**. Donner le privilège DELETE sur la table EMP appartenant à l'utilisateur **admpaye** à un des utilisateurs du groupe **PAYE**. Faire l'état des lieux concernant les privilèges objets distribués.

7/ Tester les privilèges en prenant l'identité des utilisateurs du groupe **PAYE**.

6. Maintenance d'une base de données (TP du chapitre 7)

1/ Utiliser quelques fonctions systèmes d'administration livrées par PostgreSQL sur une de vos bases pour connaître la taille d'une table ou d'une base.

2/ Positionner le paramètre « **log_autovacuum_min_duration** » à 0 (toutes les actions de autovacuum seront loggées). Effectuer des mises à jour volumineuses (insert ou update ou delete) sur une table (plusieurs dizaines de milliers de lignes) et observer le déclenchement automatique des vacuum / analyze sur cette table (démarrage automatique du process « autovacuum worker ») dans le fichier de log du serveur.

3/ Désactiver **autovacuum** (autovacuum à off). Installer la contrib **oid2name** et utiliser **oid2name** pour visualiser le nom du répertoire correspondant à une de vos bases de données et repérer le nom du fichier O.S d'une table de cette base.

4/ Augmenter la taille de cette table en ajoutant des lignes (insert…select…) et observer la taille du fichier correspondant. Effectuer des delete dans cette table et observer si la taille du fichier O.S correspondant diminue ou pas (select pg_size_pretty(pg_relation_size('nomdelatable'));).

5/ Effectuer un **VACUUM ANALYZE** sur cette base de données et observer si la taille du fichier O.S de votre table diminue ou pas. Refaire la même opération avec un **VACUUM FULL ANALYZE**.

6/ Positionner les paramètres suivants :

```
log_destination = 'stderr'
logging_collector = on
log_directory = '/home/postgres/log'
log_filename = 'postgresql-%Y-%m-%d_%H%M%S.log'
log_min_duration_statement = 0
log_checkpoints = on
log_connections = on
log_disconnections = on
log_duration = on
log_line_prefix = '%r-%d-%u-%t'
log_statement = 'ddl'
log_hostname = on
```

Arrêter / démarrer votre serveur puis visualiser le contenu du fichier log de votre serveur.

<u>Si vous avez du temps</u> :

7/ Mettre en œuvre pgBadger

7. Psql et pgAdmin 4 (TP du chapitre 8)

psql

1/ Ouvrir une session avec psql et effectuer une mise à jour (insert,update, delete) sans la committer (utiliser la commande SQL BEGIN). Constater les modifications dans votre session (select). Ouvrir une deuxième session avec psql et constater que les modifications en cours ne sont pas visibles. Committer la première session et constater les modifications dans la deuxième session.

2/ Utiliser quelques commandes spécifiques de **psql** que vous n'avez pas encore utiliser (\o, \t, \timing, \set, \H ….).

3/ Utiliser une option de lancement de psql (-d, -U, -f …).

Si vous avez du temps :

pgAdmin 4

4/ Installer pgAdmin 4.

5/ Afficher les bases de données de votre serveur, les tablespaces, les rôles, les tables de la base prod .... Visualiser sur la partie droite de pgadmin 3, les propriétés, les statistiques et le ddl d'une table de la base prod.

6/ Démarrer l'éditeur de requêtes SQL. Effectuer une requête sur les tables de votre base prod et récupérer le plan d'exécution.

8. Sauvegardes et restaurations (TP du chapitre 10)

1/ Effectuer un **pg_dump** d'une de vos base de données dans un fichier texte incluant le create database au début de la sauvegarde. Visualiser le contenu du fichier texte produit par pg_dump.

2/ Supprimer la base de données (drop database) précédemment sauvegardée et la recréer en utilisant **psql** et le fichier texte produit par pg_dump.

3/ Effectuer une sauvegarde avec pg_dump d'une autre base de données dans un fichier en format tar. Supprimer la base de données (drop database) précédemment sauvegardée et la recréer en utilisant **pg_restore**.

4/ Configurer votre serveur PostgreSQL en mode archive. Générer de l'activité sur votre serveur et vérifier la production des archives.

5/ Arrêter votre serveur PostgreSQL. Sauvegarder (tar) le répertoire PGDATA et le répertoire du tablespace de votre serveur PostgreSQL.

2. SOLUTIONS DES TRAVAUX PRATIQUES

1. Installation via les sources

1/ cp postgresql-9.1.n.tar.gz /usr/local/src

2/ cd /usr/local/src
tar –xzvf postgresql-9.1.n.tar.gz

3/ cd /usr/local/src/postgresql-9.1.n
./configure

4/ gmake

5/ gmake install

6/ adduser postgres

7/ su – postgres
vi .bash_profile
……
PATH=$PATH:$HOME/bin:/usr/local/pgsql/bin
export PATH
……

8/ su – postgres
cd /usr/local/src/postgresql-9.1.n
gmake check

2. Mise en œuvre d'un serveur de données PostgreSQL

1/ su – postgres

initdb –D /home/postgres/data

2/ vi .bash_profile

……

PGDATA=/home/postgres/data

export PATH MANPATH PGDATA

……

su - postgres

pg_ctl start –l serverlog

pg_ctl status

cat serverlog

3/ vi $PGDATA/postgresql.conf

…..

listen_addresses='*'

shared_buffers=300MB

work_mem=10MB

checkpoint_segments=10

…..

pg_ctl restart –l serverlog

psql template1

template1=# select * from pg_settings;

4/ ps -ef|grep postgres

3. Création d'une base de données

1/ psql template1

template1=# \l+

2/ template1=# create database test;

template1=# create database test1with template=test;

3/ mkdir /home/postgres/tbsp1

psql template1

template1=# create tablespace tbsp1

template1-# location '/home/postgres/tbsp1';

4/ template1=# create database test2 tablespace=tbsp1;

template1=# select * from pg_database;

…….

template1=# select * from pg_tablespace;

…….

template1=# \l+

…….

template1=# \db

…….

template1=# \db+

…….

template1=# \q

ls –l $PGDATA/pg_tblspc

…….

ls –l /home/postgres/tbsp1

…….

5/ dropdb test2

6/ template1=# \c test

test=# show enable_mergejoin;

7/ test=# alter database test set enable_mergejoin to off;

test=# show enable_mergejoin;

8/ test=# \q

psql test

test=# show enable_mergejoin;

4. Authentification des clients

1/ psql template1

template1=# create role jean login password ‘jean’;

2/ template1=# alter role postgres password 'postgres';

3/ vi $PGDATA/pg_hba.conf

……..

host	test	jean	127.0.0.1/32	md5
host	all	postgres	127.0.0.1/32	md5
local	all	all		reject

……..

4/ pg_ctl reload

psql test –U jean refusé

psql test –U jean –h localhost

password for user jean:

test=# connexion ok

test=# \q

psql template1 -h localhost

password for user postgres:

template1=# connexion ok

template1=# \q

psql template1 refusé

5/ vi $PGDATA/pg_hba.conf

……..

local	all	all	trust

……..

pg_ctl reload

5. Gestion de la sécurité

1/ psql template1

template1=# create role admpaye login password ‘admpaye’;

2/ template1=# create database prod;

3/ template1=# \c prod admpaye

prod=# \i demotab.txt

4/ prod=# \c template1 postgres

template1=# create role paye;

5/ template1=# create role pierre login password ‘pierre’;

template1=# create role paul login password ‘paul’;

template1=# create role anne login password;

template1=# grant paye to pierre,paul,anne;

6/ template1=# \c prod admpaye

prod=# grant select,insert,update on emp to paye;

prod=# grant delete on emp to anne;

prod=# \dp

7/ prod=# \c prod pierre

prod=# select * from emp;

prod=# \c prod anne

prod=# delete from emp where deptno=10;

6. Maintenance d'une base de données

1/ psql prod

prod=# select pg_relation_size('emp');

prod=# select pg_database_size('prod');

2/ cat /home/postgres/data/postgresql.conf

.....

.....

log_autovacuum_min_duration = 0

.....

psql prod

insert into emp select * from emp; plusieurs fois puis

delete from emp where deptno=20;

cat /home/postgres/serverlog

3/ cat /home/postgres/data/postgresql.conf

.....

.....

autovacuum = off

.....

su – root

cd /usr/local/src/postgresql-9.1.n/contrib/oid2name/

make

make install

su – postgres

oid2name

oid2name –s

oid2name –d prod –x

oid2name –d prod –t emp –x

.....

4/

```
[postgres@linux55 ~]$ cd $PGDATA/base/16427
[postgres@servoracle 16427]$ ls -l|grep 16467
-rw-------    1 postgres postgres 1073741824 mar 24 11:56 16467
-rw-------    1 postgres postgres 205611008 mar 24 11:56 16467.1
[postgres@linux55 ~]$ psql prod
psql (9.1.2)
Type "help" for help.
prod=# select count(*) from emp;
  count
----------
 14680064
(1 row)
prod=# delete from emp where deptno=10;
DELETE 3145728
prod=# \!ls -l|grep 16467
-rw-------    1 postgres postgres 1073741824 mar 24 12:07 16467
-rw-------    1 postgres postgres 205611008 mar 24 12:08 16467.1
```

5/

```
prod=# vacuum verbose analyze emp;
INFO:  vacuuming "public.emp"
INFO:  index "emp_ename" now contains 11534336 row versions in 55209 pages
……
CPU 11.01s/16.92u sec elapsed 395.66 sec.
INFO:  analyzing "public.emp"
INFO:  "emp": scanned 3000 of 156171 pages, containing 221583 live rows and 0 dead rows; 3000 rows in sample,
11534946 estimated total rows
VACUUM
prod=# \!ls -l|grep 16467
-rw-------    1 postgres postgres 1073741824 mar 24 12:16 16467
-rw-------    1 postgres postgres 205611008 mar 24 12:19 16467.1
prod=# delete from emp where deptno=20;
DELETE 5242880
prod=# \!ls -l|grep 16467
-rw-------    1 postgres postgres 1073741824 mar 24 12:22 16467
-rw-------    1 postgres postgres 205611008 mar 24 12:23 16467.1
prod=# select count(*) from emp;
  count
---------
 6291456
(1 row)
```

```
prod=# vacuum full verbose analyze emp;
INFO:  vacuuming "public.emp"
INFO:  "emp": found 5242880 removable, 6291456 nonremovable row versions in 156171
pages
……
……
DETAIL:  3460296 index row versions were removed.
33102 index pages have been deleted, 20000 are currently reusable.
CPU 5.38s/3.71u sec elapsed 1315.87 sec.
INFO:  analyzing "public.emp"
INFO:  "emp": scanned 3000 of 70277 pages, containing 268604 live rows and 0 dead
rows; 3000 rows in sample, 6292228 estimated total rows
VACUUM
prod=# \!ls -l|grep 16467
-rw-------    1 postgres postgres 575709184 mar 24 13:08 16467
```

6/ `cat /home/postgres/data/postgresql.conf`

```
.....
log_destination = ‘stderr’
logging_collector = on
log_directory = ‘/home/postgres/log’
log_filename = 'postgresql-%Y-%m-%d_%H%M%S.log‘
log_min_duration_statement = 0
log_checkpoints = on
log_connections = on
log_disconnections = on
log_duration = on
log_line_prefix = ‘%r-%d-%u-%t’
log_statement = ‘ddl’
log_hostname = on
....
```

```
pg_ctl stop
pg_ctl start
vi /home/postgres/log/postgresql-nnnn-nn-nn_nnnnnn.log
```

7/

```
[root@linux55 ~]# rpm -qa|grep php
php-cli-5.1.6-27.el5
php-5.1.6-27.el5
php-ldap-5.1.6-27.el5
php-common-5.1.6-27.el5
[root@linux55 ~]# ls -l|grep pgfouine
-rw-r--r-- 1 root root   598345 jan 30 14:02 pgfouine-1.2-1.noarch.rpm
[root@linux55 ~]# rpm -ivh pgfouine-1.2-1.noarch.rpm
PrÃ©paration...              ########################################### [100%]
   1:pgfouine               ########################################### [100%]
[root@linux55 ~]#
```

```
cat /home/postgres/data/postgresql.conf
                      .....
log_destination = 'stderr'
logging_collector = on
log_directory = '/home/postgres/log'
log_filename = 'postgresql-%d-%m-%Y_%H%M%S.log'
log_min_duration_statement = 0
log_duration = off
log_statement = 'none'
log_line_prefix = '%t [%p]: [%l-1] '
lc_messages = 'en_US.UTF-8'
                      ....
```

```
[postgres@linux55 ~]$ pgfouine.php -logtype stderr -file
/home/postgres/log/postgresql-30-01-2012_162712.log >
/home/postgres/www/rapport3.html
```

Ouvrir ensuite le fichier /home/postgres/www/rapport3.html avec un navigateur

8. Sauvegardes et restaurations

1/

```
[postgres@servoracle postgres]$ pg_dump prod -f proddump -C -v
pg_dump: reading schemas
pg_dump: reading user-defined functions
pg_dump: reading user-defined types
……
[postgres@servoracle postgres]$ ls -l|grep prod
-rw-rw-r--    1 postgres postgres   230237 Mar 31 10:40 proddump
[postgres@servoracle postgres]$ less proddump
--
-- PostgreSQL database dump
--
-- Started on 2006-03-31 10:40:51 CEST
SET client_encoding = 'LATIN9';
SET check_function_bodies = false;
SET client_min_messages = warning;
……
```

2/

```
[postgres@servoracle postgres]$ dropdb prod
DROP DATABASE
[postgres@servoracle postgres]$ psql template1 -f proddump
SET
SET
SET
CREATE DATABASE
ALTER DATABASE
……
```

3/

```
[postgres@servoracle postgres]$ pg_dump test -f testdump.tar -F t -C
[postgres@servoracle postgres]$ dropdb test
DROP DATABASE
[postgres@servoracle postgres]$ pg_restore -d template1 -C testdump.tar -v
pg_restore: connecting to database for restore
pg_restore: creating DATABASE test
pg_restore: connecting to new database "test"
pg_restore: connecting to database "test" as user "postgres"
pg_restore: creating SCHEMA public
……
```

4/

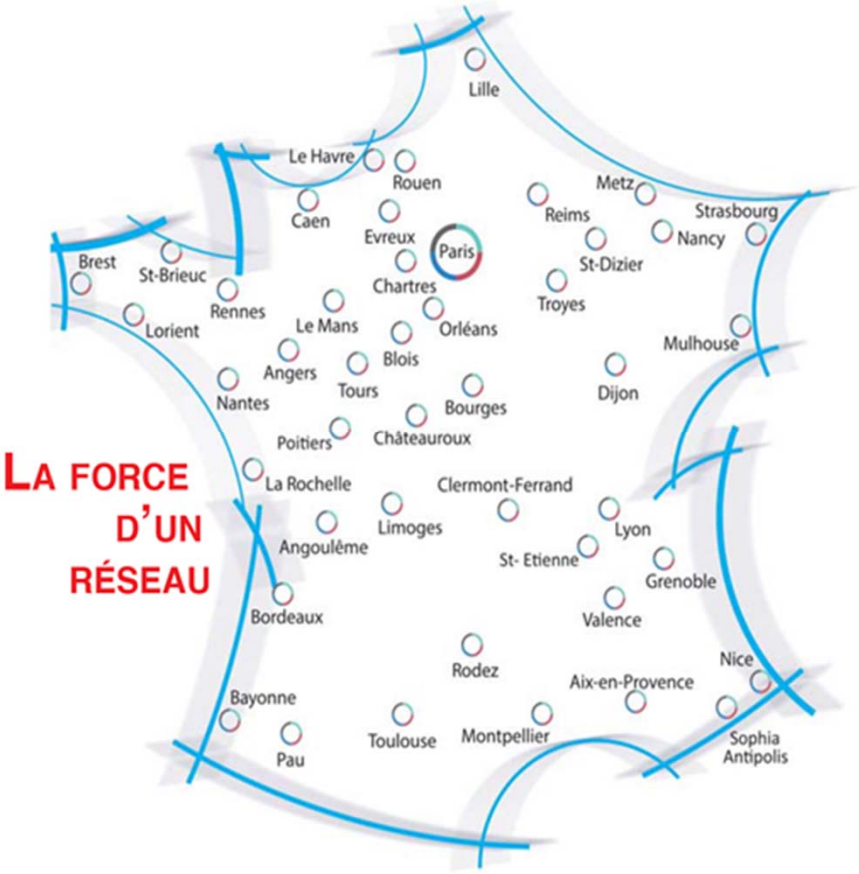
```
[postgres@linux55 ~]$ cat /home/postgres/data/postgresql.conf
# -----------------------------
# PostgreSQL configuration file
# -----------------------------
…/…
#------------------------------------------------------------------------------
# WRITE AHEAD LOG
#------------------------------------------------------------------------------
# - Settings -
wal_level = archive
#wal_level = minimal                    # minimal, archive, or hot_standby
…/…
# - Archiving -
archive_mode = on
#archive_mode = off             # allows archiving to be done
                                # (change requires restart)
archive_command = 'cp -i %p /home/postgres/arch/%f </dev/null'
#archive_command = ''           # command to use to archive a logfile segment
…/…
[postgres@linux55 ~]$ mkdir /home/postgres/arch
[postgres@linux55 ~]$ pg_ctl -D /home/postgres/data -l serverlog start
server starting
[postgres@linux55 ~]$ pg_ctl status
pg_ctl: server is running (PID: 16922)
/usr/local/pgsql/bin/postgres "-D" "/home/postgres/data
[postgres@linux55 ~]$ mkdir /home/postgres/arch
[postgres@linux55 ~]$ psql mabase
psql (9.1.2)
Type "help" for help.


mabase=# CREATE TABLE t1 as select * from pg_class, pg_attribute;
SELECT 639557
```

```
mabase=# \! ls -l /home/postgres/arch
total 213252
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:24 000000010000000000000009
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:24 00000001000000000000000A
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:24 00000001000000000000000B
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:24 00000001000000000000000C
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:24 00000001000000000000000D
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:24 00000001000000000000000E
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:24 00000001000000000000000F
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:25 000000010000000000000010
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:25 000000010000000000000011
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:25 000000010000000000000012
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:25 000000010000000000000013
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:25 000000010000000000000014
-rw------- 1 postgres postgres 16777216 fÃ©v  7 14:25 000000010000000000000015
mabase=#
```

5/

```
[postgres@linux55 ~]$ pg_ctl stop
waiting for postmaster to shut down.... done
postmaster stopped
[postgres@linux55 ~]$ tar -cf backup_data.tar /home/postgres/data/
[postgres@linux55 ~]$ tar -cf backup_data.tar /home/postgres/tbsp1/
```



Formation

N°Azur 0 810 007 689

PRIX D'UN APPEL LOCAL DEPUIS UN POSTE FIXE

Découvrez également l'ensemble des stages à votre disposition sur notre site

http://www.m2iformation.fr