Laboratorio

En este laboratorio se identificará los procesos y se verán la estructura de la base de datos, de forma gráfica y mediante la consola de comandos.

Nota:

Las tareas se realizará sobre una máquina virtual VMWare y un sistema operativo Ubuntu.

Práctica 1: Estructura y arquitectura de PostgreSQL

Objetivos

El objetivo de la práctica es aprender la estructura y arquitectura de PostgreSQL que tenemos en el cluster

Notas

Para facilitar las prácticas, se ofrece una máquina virtual ya creada.
La máquina virtual descargada tiene la siguiente información:

- Usuario: postgresql96
- Contraseña: postgresql

Para utilizar la máquina virtual puede utilizar el software gratuito VMWare Player.

Todas las tareas deberán ser ejecutadas en la consola de Linux (Aplicación Terminal).

Tareas

1.- Crea un cluster con el comando initdb donde el directorio PGDATA se encuentre en /curso/postgresql/ y la codificación sea UTF-8 e inícialo con el comando pg_ctl. Indica los comandos resultantes utilizados.

2.- Identifica los procesos PostgreSQL que hay en el sistema con el comando ps -aux, Indica el comando utilizado y el nombre de todos los procesos que encuentres relacionados con PostgreSQL y el propietario. ¿Para qué sirven cada proceso?

3.- Crea una variable del sistema llamada $PGDATA. Incluye la ruta del cluster anteriormente creado en la variable y exportarla.

4.- Conecta a la base de datos postgres. Posteriormente abre otro terminal e indica si hay algún proceso nuevo e indica para qué sirve.

5.- Conecta a la base de datos postgres esta vez utilizando la opción -h localhost. Posteriormente abre otro terminal e indica si hay algún proceso nuevo e indica qué diferencia hay con el anterior.

6.- Accede al directorio $PGDATA/base e indica los directorios que encuentras. En psql, ejecuta la consulta "**select oid, datname from pg_database;**" e índica qué relación hay entre el nombre de los directorios y el resultado de la consulta.

7.- Vamos a crear una nueva tabla con el siguiente comando: "**create table pruebas (col integer);**" . Localiza la nueva tabla dentro del directorio base.

8.- Crea una nueva tabla llamada prueba2 con el siguiente comando: "**create tabla pruebas2(col1 integer, col2 char(150));**" . Posteriormente ejecuta el comando: "**insert into pruebas2 select generate_series(1,10) AS col1, md5(random()::text) AS col2;**" . Localiza la tabla en el directorio base e indica el tamaño que tiene la nueva tabla.

9.- Ejecuta 4 veces más el comando "**insert into pruebas2 select generate_series(1,1000000) AS col1, md5(random()::text) AS col2;**". ¿Qué ha ocurrido con el fichero de la tabla? ¿Por qué?.

1.- Soluciones

1.- Crea un cluster con el comando initdb donde el directorio PGDATA se encuentre en /curso/postgresql/ y la codificación sea UTF-8 e inícialo con el comando pg_ctl. Indica los comandos resultantes utilizados.

```
sudo mkdir /curso/postgresql/arquitectura/
sudo chown postgresql96 /curso/postgresql/arquitectura/
initdb -D /curso/postgresql/arquitectura/ -E UTF8
pg_ctl -D /curso/postgresql/arquitectura/ start
```

2.- Identifica los procesos PostgreSQL que hay en el sistema con el comando ps -aux, Indica el comando utilizado y el nombre de todos los procesos que encuentres relacionados con PostgreSQL y el propietario. ¿Para qué sirven cada proceso?

```
ps -aux | grep postgres
```

postgres: checkpointer process: Proceso encargado de limpiar todos los bloques "sucios" llevándolos a disco.

postgres: writer process: Proceso encargado de coger todas las páginas modificadas desde shared buffer.

postgres: wal writer process : Proceso encargado de escribir y sincronizar los ficheros WAL hacia disco cuando ocurra una transacción.

postgres: autovacuum launcher process: Proceso encargado de automatizar comandos VACUUM y ANALYZE.

postgres: stats collector process : Recolecta información acerca de la actividad del cluster como puede ser el número de accesos a una tabla o índice, número de tuplas de una tabla e información acerca de las acciones realizadas por ANALYZE y VACUUM por cada tabla.

3.- Crea una variable del sistema llamada $PGDATA. Incluye la ruta del cluster anteriormente creado en la variable y exportarla.

```
PGDATA="/curso/postgresql/arquitectura/"
export PGDATA
```

4.- Conecta a la base de datos postgres. Posteriormente abre otro terminal e indica si hay algún proceso nuevo e indica para qué sirve.

```
psql postgres
```

Abrimos nuevo terminal

```
ps -aux | grep postgres
```

Aparece un nuevo proceso:
postgres: postgresql96 postgres [local] idle : Proceso hijo de Postmaster que se encarga de autentificar peticiones de conexión, gestionar consultas y mandar los resultados a las aplicaciones clientes. El usuario conectado es postgresql96 y la base de datos es postgres.

5.- Conecta a la base de datos postgres esta vez utilizando la opción -h localhost. Posteriormente abre otro terminal e indica si hay algún proceso nuevo e indica qué diferencia hay con el anterior.

```
psql -h localhost postgres
```

Abrimos nuevo terminal

```
ps -aux | grep postgres
```

Aparece un nuevo proceso:

postgres: curso postgres 127.0.0.1(33316) idle : La diferencia es que el anterior está conectado por socket y en este caso por TCP. El puerto abierto para la comunicación del proceso postgres con el cliente es el 33316.

6.- Accede al directorio $PGDATA/base e indica los directorios que encuentras. En psql, ejecuta la consulta "select oid, datname from pg_database;" e índica qué relación hay entre el nombre de los directorios y el resultado de la consulta.

Directorio $PGDATA/base:

```
postgresql96@ubuntu:/curso/postgresql$ cd config/
postgresql96@ubuntu:/curso/postgresql/config$ cd base/
postgresql96@ubuntu:/curso/postgresql/config/base$ ls
1  12438  12439
```

La consulta realiza la siguiente salida.

```
postgres=# select oid, datname from pg_database ;
  oid  |  datname
-------+-----------
 12439 | postgres
     1 | template1
 12438 | template0
(3 rows)
```

El identificador de objeto (OID) nos proporciona el mismo número que el nombre del directorio en el directorio base. Así pues, en el directorio 1 está todo lo relacionado con template1 y en el directorio 12439 está todo lo relacionado con la base de datos postgres.

7.- Vamos a crear una nueva tabla con el siguiente comando: "create table pruebas (col integer);" . Localiza la nueva tabla dentro del directorio base.

Una forma de saber como se llama el fichero creado podría ser ordenando por fecha el directorio relacionado con la base de datos postgres, podría ser algo así:

```
postgresql96@ubuntu:/curso/postgresql/config/base/12439$ ls -lt
total 303492
-rw------- 1 postgresql96 postgresql96 303415296 May 15 11:30 16387
-rw------- 1 postgresql96 postgresql96     98304 May 15 11:30 16387_fsm
-rw------- 1 postgresql96 postgresql96     16384 May 15 11:27 2696
-rw------- 1 postgresql96 postgresql96     24576 May 15 11:27 2840
-rw------- 1 postgresql96 postgresql96      8192 May 15 11:27 2840_vm
-rw------- 1 postgresql96 postgresql96     16384 May 15 11:27 2841
-rw------- 1 postgresql96 postgresql96     90112 May 15 11:27 1259
-rw------- 1 postgresql96 postgresql96    131072 May 15 11:27 2619
-rw------- 1 postgresql96 postgresql96      8192 May 15 11:27 2619_vm
-rw------- 1 postgresql96 postgresql96    368640 May 15 11:24 1249
-rw------- 1 postgresql96 postgresql96     32768 May 15 11:22 3455
-rw------- 1 postgresql96 postgresql96     40960 May 15 11:22 2704
-rw------- 1 postgresql96 postgresql96     16384 May 15 11:22 2703
-rw------- 1 postgresql96 postgresql96    344064 May 15 11:22 2674
-rw------- 1 postgresql96 postgresql96    327680 May 15 11:22 2673
-rw------- 1 postgresql96 postgresql96     40960 May 15 11:22 2663
-rw------- 1 postgresql96 postgresql96     32768 May 15 11:22 2662
-rw------- 1 postgresql96 postgresql96     73728 May 15 11:22 2659
-rw------- 1 postgresql96 postgresql96    106496 May 15 11:22 2658
-rw------- 1 postgresql96 postgresql96    450560 May 15 11:22 2608
-rw------- 1 postgresql96 postgresql96     73728 May 15 11:22 1247
-rw------- 1 postgresql96 postgresql96      8192 May 15 11:12 2608_vm
-rw------- 1 postgresql96 postgresql96      8192 May 15 11:12 1259_vm
-rw------- 1 postgresql96 postgresql96      8192 May 15 11:12 1249_vm
-rw------- 1 postgresql96 postgresql96      8192 May 15 11:12 1247_vm
-rw------- 1 postgresql96 postgresql96         0 May 15 11:09 16384
```

En un sistema con mucho movimiento, no sería fácil encontrarlo, para ello podemos utilizar la siguiente consulta:

```
postgres=# select relname, relfilenode from pg_class where relname='pruebas';
 relname | relfilenode
---------+-------------
 pruebas |       16384
(1 row)
```

8.- Crea una nueva tabla llamada prueba2 con el siguiente comando: "create tabla pruebas2(col1 integer, col2 char(150));" . Posteriormente ejecuta el comando: "insert into pruebas2 select generate_series(1,10) AS col1, md5(random()::text) AS col2;" . Localiza la tabla en el directorio base e indica el tamaño que tiene la nueva tabla.

```
postgres=# create table pruebas2(col1 integer, col2 char(250));
CREATE TABLE
postgres=# insert into pruebas2 select generate_series(1, 1000000) as col1, md5(
random()::text) as col2;
    Terminal    0000
postgres=# select relname, relfilenode from pg_class where relname='pruebas2';
 relname  | relfilenode
----------+-------------
 pruebas2 |       16387
(1 row)
```

```
postgresql96@ubuntu:/curso/postgresql/config/base/12439$ ls -lth 16387
-rw------- 1 postgresql96 postgresql96 290M May 15 11:30 16387
```

En este caso se ha creado una tabla y se ha insertado en la tabla una serie de datos desde el 1 hasta 1.000.000. El tamaño ocupado por esta tabla es de 290Mb.

9.- Ejecuta 4 veces más el comando "insert into pruebas2 select generate_series(1,1000000) AS col1, md5(random()::text) AS col2;". ¿Qué ha ocurrido con el fichero de la tabla? ¿Por qué?.

```
postgres=# insert into pruebas2 select generate_series(1, 1000000) as col1, md5(
random()::text) as col2;
INSERT 0 1000000
postgres=# insert into pruebas2 select generate_series(1, 1000000) as col1, md5(
random()::text) as col2;
INSERT 0 1000000
postgres=# insert into pruebas2 select generate_series(1, 1000000) as col1, md5(
random()::text) as col2;
LOG:  checkpoints are occurring too frequently (22 seconds apart)
HINT:  Consider increasing the configuration parameter "max_wal_size".
INSERT 0 1000000
postgres=# insert into pruebas2 select generate_series(1, 1000000) as col1, md5(
random()::text) as col2;
INSERT 0 1000000
```

```
postgresql96@ubuntu:/curso/postgresql/config/base/12439$ ls -lth 16387*
-rw------- 1 postgresql96 postgresql96 423M May 15 11:49 16387.1
-rw------- 1 postgresql96 postgresql96 1.0G May 15 11:49 16387
-rw------- 1 postgresql96 postgresql96 384K May 15 11:48 16387_fsm
```

El tamaño máximo del fichero es de 1Gb, por ello, al llegar a este tamaño, se crea uno nuevo con el sufijo .1. Si llenáramos más este fichero, entonces aparecería un nuevo fichero con un sufijo .2 . El fichero terminado en fsm es del mapa de espacio libre de la tabla.
EL log indica que se están acumulando los checkpoints con demasiada frecuencia.